SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 1–9, February 1992
© 1992 Society for Industrial and Applied Mathematics

A LINEAR TIME APPROACH TO THE SET MAXIMA PROBLEM*

AMOTZ BAR-NOY†, RAJEEV MOTWANI†, AND JOSEPH NAOR†

Abstract. The set maxima problem is as follows: given a family of subsets $\mathcal{S}$ of a totally ordered set $X = \{x_1, \cdots, x_n\}$, find the maximum in each subset. The computational model is the comparison tree. One possible solution is to sort the set X, which requires $O(n \log n)$ comparisons. The open question is whether set maxima is easier than sorting.

Here, a solution is presented that requires a linear number of comparisons for the following two cases:

- The sets are hyperplanes in a d-dimensional projective geometry $PG(d, q)$. In particular, the interesting case is $PG(2, q)$, when the intersection of any two subsets is exactly one.
- The sets are chosen randomly with probability $p(n)$ for each element to be in a set. The random choices are mutually independent and the number of comparisons needed is linear with probability approaching 1 asymptotically.

Key words. set maxima, comparison model, projective geometry

AMS(MOS) subject classifications. 68P10, 51A05

1. Introduction. Let $X = \{x_1, \cdots, x_n\}$ be a set of n distinct elements on which a total order is defined, and let $\mathcal{S} = \{S_1, \cdots, S_n\}$ be a family of n nonempty subsets of the base set X. The *set maxima* problem is as follows: compute a *maxima vector* $\langle y_1, \cdots, y_n \rangle$ such that $y_j = \max_i \{x_i \in S_j\}$. The model of computation is the comparison tree model in which the cost of the computation is the number of comparisons between the base elements, while any other computation is free.

This question was first posed by Graham, Yao, and Yao [GYY] in the context of verifying partial orders. A partial order P is given and its validity is to be verified; i.e., for all $x \in P$ test that x is the maximum among all elements smaller than it in P. This question is a special case of set maxima and the reader is referred to [KK] for details.

A trivial solution to the set maxima problem is to sort the elements in the set X, which requires $O(n \log n)$ comparisons. The problem is to determine whether there exists a more efficient solution. Deterministically, this question still remains open. A randomized algorithm that makes only $O(n \log^{1/3} n)$ comparisons was presented in [KK]. Recently, Goddard, King, and Schulman [GKS] gave a randomized algorithm for the graphical local sorting problem, which was used to devise a randomized algorithm for the set maxima problem that makes only $O(n)$ comparisons.

Komlós [Ko] considered the special case of the set maxima problem, which is to verify the solution to a minimum spanning tree problem. Verifying a solution to this problem is equivalent to checking whether every nontree edge has the maximum weight in the simple cycle that it forms with tree edges. Komlós presented a linear time algorithm for this special case of the set maxima problem.

There are no nontrivial lower bounds known for this problem. Let a vector Y be called *feasible* if there exists a permutation of the elements X for which Y is a maxima vector. The difficulty in proving a lower bound seems to stem from Fredman's proof [GYY] that the number of distinct feasible vectors can be at most 2^{2n}. Therefore, the standard information theoretic lower-bound argument cannot be applied here.

* Received by the editors September 2, 1989; accepted for publication (in revised form) November 16, 1990.

† Computer Science Department, Stanford University, Stanford, California 94305. Respectively, the authors were supported in part by a Weizmann fellowship and by Office of Naval Research contract N00014-88-K-0166, in part by National Science Foundation grant CCR-9010517, and by Office of Naval Research contract N00014-88-K-0166.

In this paper we present deterministic solutions for the following two cases. Both algorithms require a linear number of comparisons.

1. *Projective geometry*. The set X corresponds to the points of the d-dimensional projective geometry, $PG(d, q)$, and where each hyperplane is a subset in the family $\mathcal{S}$. In particular, the interesting case is when the dimension is two, and any two subsets intersect in exactly one element.

2. *Probabilistic setting*. The family $\mathcal{S}$ is chosen at random. For every pair $x_i \in X$ and $S_j \in \mathcal{S}$, Prob $[x_i \in S_j] = p$, where $p = p(n)$ is any arbitrary function of n; also, all the random choices are mutually independent. We show that, with probability tending to one, a linear number of comparisons suffice.

It was conjectured by several researchers that the set system induced by projective planes may be a hard instance of the set maxima problem. Intuitively, the set maxima problem becomes harder as the intersections between subsets get smaller. In a projective plane, any two subsets intersect in exactly one element and, hence, not much "information" is gained for other subsets from the comparisons performed within one subset. To obtain a linear-time solution, we exploit the expansion properties for projective geometries that were established by Alon [Al].

In general, under the comparison tree model, most problems seem to have the same complexity (within constant factors) in the deterministic worst-case, randomized case and in the case where the input is drawn from some natural probability distribution. This is especially true for problems involving sorting and selection. Thus, the randomized algorithm of [KK], [GKS] and our probabilistic analysis seem to indicate that the set maxima problem is easier than sorting.

It is interesting to note that both the probabilistic case and the projective geometry are solved by the same generic algorithm. It seems possible that some version of our generic algorithm might lead to an efficient solution for the general case.

The rest of the paper is organized as follows. In § 2 we give some preliminaries, § 3 contains the generic algorithm, and the solutions for the projective geometry and for randomly chosen sets are described in §§ 4 and 5, respectively.

2. Preliminaries. Let W_x be the set of elements in X that are greater than or equal to x. The *rank* of an element $x \in X$ is defined to be the cardinality of the set W_x. It is well known [AHU, pp. 97–102] that the k-selection computation, i.e., finding the element whose rank is k, requires a linear number of comparisons. It is implicit that the output of k-selection is (i) an element x whose rank is k, and (ii) a partition of the remaining elements of X into two sets where one set contains all elements whose rank is greater than k and the other set contains all elements whose rank is smaller than k.

A finite plane geometry is an incidence system containing two types of elements, *points* and *lines*. The incidence relation is $p \in l$; i.e., point p is on line l. A projective geometry $PG(2, q)$ is characterized by the following properties:

1. There is a unique line containing any two distinct points.
2. Any two lines intersect and their intersection contains exactly one point.
3. The number of points in the plane is $n = q^2 + q + 1$, where q is a prime power.
4. Each line contains exactly $q + 1$ points and each point is incident on exactly $q + 1$ lines.

Higher-dimension projective geometries, $PG(d, q)$, are defined recursively. The incidence relation is $p \in h$; i.e., point p is on hyperplane h. The equivalent properties are:

1. There is a unique hyperplane containing any d distinct points.
2. Any d hyperplanes intersect and their intersection contains exactly one point.

3. The number of points is $n = q^d + q^{d-1} + \cdots + 1$, where q is a prime power.

4. Each hyperplane contains exactly $q^{d-1} + \cdots + 1$ points and each point is incident on exactly $q^{d-1} + \cdots + 1$ lines.

For more details on projective geometries, the reader is referred to [Ba].

3. The generic algorithm. In this section a generic algorithm for computing the set maxima is described. Let R be an integer sequence $n \geqq r_1 \geqq r_2 \geqq \cdots \geqq r_k \geqq 1$; the entries of the sequence will vary in the different applications.

1. Compute the sequence Z, $z_1 \leqq \cdots \leqq z_k$ such that z_i is the element in X whose rank is r_i. Let
   (a) $Z_0 = \{x \in X \mid x \leqq z_1\}$;
   (b) $Z_i = \{x \in X \mid z_i < x \leqq z_{i+1}\}$, for $1 \leqq i \leqq k-1$;
   (c) $Z_k = \{x \in X \mid z_k < x\}$.
2. For all $1 \leqq j \leqq n$, let $S'_j = S_j \cap Z_{\ell(S_j)}$ where $\ell(S_j)$ is the largest index i such that $S_j \cap Z_i \neq \varnothing$. Compute $y_j = \max\{S'_j\}$.

The set S'_j is the *reduced set* of S_j, and $\ell(S_j)$ denotes its level. Let us denote the cardinality of the set $Z_{\ell(S_j)}$ by $h(S_j)$.

The number of comparisons performed in the generic algorithm is

$$C(n) = f_R(n) + \sum_{j=1}^{n} (|S'_j| - 1),$$

where $f_R(n)$ is the number of comparisons needed to compute the sequence Z. The maximum is computed in each S'_j by brute force using $|S'_j| - 1$ comparisons.

The following theorem gives a generic bound on the complexity of the *simultaneous selection* problem, viz., the problem of computing the sequence Z for a given rank sequence R.

THEOREM 3.1. *For any given sequence R, the comparison complexity of computing the sequence Z is*

$$f_R(n) = \Theta\left(n\left(1 - \sum_{i=0}^{k} \frac{r_{i-1} - r_i}{n} \log \frac{r_{i-1} - r_i}{n}\right)\right),$$

where $r_0 = n$ and $r_{k+1} = 0$.

Proof. This theorem is a well-known "folklore" fact and we will omit the proof. Suffice it to say that the lower bound is obtained by a straightforward leaf-counting information theory lower bound. As for the upper bound, it is obtained by a simple recursion: choose j so that $r_j \leqq n/2 \leqq r_{j-1}$ and find z_j and z_{j-1} using any linear-time selection algorithm [AHU]. Let $X^- = \{x \mid x < x_{j-1}\}$ and $X^+ = \{x \mid x > x_j\}$, and the remaining sequence can be obtained by recursing on X^- and X^+. □

4. The projective geometry. First, suppose that the base set X is represented by points in a projective plane $PG(2, q)$ and each line in the plane is a subset in the family $\mathcal{S}$.

To implement the generic algorithm for the projective plane set family, we should first specify the entries of the sequence R. The requirements are that (i) R is computable in linear time (Lemma 4.1), and (ii) The maxima of all the reduced sets is also computable in linear time (Lemma 4.5). To find such a sequence we exploit the expansion properties of projective geometries [Al] (Lemma 4.3). Intuitively, these properties entail that the number of lines that are incident with a set of points Z_i and not incident with any other set Z_j, for $j > i$, is "balanced." This bounds the number of reduced sets contained in each set Z_i. The next task is to bound the sum of the cardinalities of the reduced sets.

This is done in Lemma 4.2, which takes advantage of the block design properties of projective planes.

The sequence R is defined as follows: for all i, $1 \leq i \leq k$, $r_i = \lfloor n/b_i \rfloor$, where $b_i = i^{1-\varepsilon}$, $k = \lceil n^{1/(1-\varepsilon)} \rceil$ and $0 < \varepsilon < 1$. For this sequence R, it can be shown using Theorem 3.1 that the complexity of computing the sequence Z is $O(n)$. However, for the sake of completeness, the following lemma directly proves this assertion for this special case.

LEMMA 4.1. *The number of comparisons required for computing* $z_1, \cdots, z_k$ *is* $O(n)$.

Proof. The sequence Z can be computed recursively. First, compute z_2, the element whose rank is $\lceil n/2^{1-\varepsilon} \rceil$. Recursively, compute the sequence Z', $z'_1 \leq \cdots \leq z'_{k'}$, where Z' is defined analogously to Z but over the base set W_{z_2}. For all i, $1 \leq i \leq k'$, the rank of z'_i in X is

$$\left\lfloor \frac{n}{(2i)^{1-\varepsilon}} \right\rfloor - 1 \leq \left\lfloor \frac{\lfloor n/2^{1-\varepsilon} \rfloor}{i^{1-\varepsilon}} \right\rfloor \leq \left\lfloor \frac{n}{(2i)^{1-\varepsilon}} \right\rfloor.$$

Hence, we either computed z_{2i} or the element whose rank is r_{2i-1}. In the latter case, z_{2i} is the minimum among elements in $[z'_i \cdots z'_{i+1}]$; the total number of additional comparisons is at most n.

Let the elements $z_i \in Z$ for which i is even (odd) be defined as having even (odd) ranks. Thus, all the elements in Z that have even ranks were computed. In between every two consecutive even-rank elements there is exactly one odd-rank element. Let d_i be the number of elements of X whose ranks are greater than z_{2i} and smaller than z_{2i+2}. The complexity of computing z_{2i+1} is $O(d_i)$. As $\sum d_i \leq n$, the complexity of computing all the odd-rank elements is linear. To summarize, we get the following recursive equation for $f_R(n)$:

$$f_R(n) = f_R\left(\frac{n}{2^{1-\varepsilon}}\right) + O(n),$$

and therefore $f_R(n) = O(n)$. □

To prove that our algorithm makes a linear number of comparisons, it remains to show that $\sum_{j=1}^{n} |S'_j| = O(n)$. Let us first prove a general lemma for projective planes.

Let $v_1, \cdots, v_s$ be a subset of the points and let $m_1, \cdots, m_t$ be a subset of the lines. Define the $t \times s$ binary matrix M such that $M_{ij} = 1$ if and only if $v_i \in m_j$.

LEMMA 4.2. *For* $1 \leq i \leq s$, *let* $c_i = \sum_{j=1}^{t} M_{ji}$ (*the number of nonzero entries in the* i*th column*). *Then*,

$$\sum_{i=1}^{s} c_i \leq t\sqrt{s} + s.$$

Proof. First assume that for all i, $c_i \geq 2$. We will prove that

$$\binom{c_1}{2} + \binom{c_2}{2} + \cdots + \binom{c_s}{2} \leq \binom{t}{2}. \tag{1}$$

The right-hand side of (1) is a count of the number of pairs of lines. We show that in the left-hand side, no pair is counted more than once. For each column i, $\binom{c_i}{2}$ is the number of pairs of lines that contain v_i. If $v_i \in m_j$, and also $v_i \in m_k$, then there cannot be any other point that belongs to both lines. Hence, the pair of lines, m_j and m_k, was counted at most once.

Let $C = \sum_{i=1}^{s} c_i$. The left-hand side of (1) is minimized when $c_i = C/s$. Then,

$$s\left(\frac{C}{s} - 1\right)^2 \leq \sum_{i=1}^{s} \binom{C/s}{2} \leq \sum_{i=1}^{s} \binom{c_i}{2} \leq \binom{t}{2} \leq \frac{t^2}{2},$$

which, when rearranged, yields $C \leq t\sqrt{s} + s$.

If for all i, $c_i < 2$, then obviously $C \leqq s$. Consequently, if there are exactly s', $0 \leqq s' \leqq s$, indices i such that $c_i \geqq 2$, then

$$C \leqq t\sqrt{s'} + (s - s') \leqq t\sqrt{s} + s. \qquad \square$$

We will use the previous lemma to estimate the size of the reduced sets S'_j, but first we need some more definitions.

DEFINITION 4.1. For all $0 \leqq i \leqq k$:

1. $s_i = |Z_i|$;
2. L_i—the set of lines that intersect Z_i but miss all the points in $\cup_{i<j\leqq k} Z_j$;
3. $t_i = |L_i|$;
4. M_i—the incidence matrix of Z_i and L_i (a $t_i \times s_i$ matrix).

The next proposition is easy to verify.

PROPOSITION 4.1. *The number of nonzero entries summed over all matrices M_i, $0 \leqq i \leqq k$, is $\sum_{j=1}^{n} |S'_j|$.*

It follows from the last proposition and Lemma 4.2 that $O(\sum_{i=0}^{k} t_i\sqrt{s_i} + \sum_{i=0}^{k} s_i)$ bounds the cost of the second stage of the algorithm. We have the following bound on s_i with respect to the sequence R:

$$s_i = \left\lfloor \frac{n}{i^{1-\varepsilon}} \right\rfloor - \left\lfloor \frac{n}{(i+1)^{1-\varepsilon}} \right\rfloor = O\left(n \frac{(i+1)^{1-\varepsilon} - i^{1-\varepsilon}}{((i+1)i)^{1-\varepsilon}} \right).$$

To bound t_i, we need the following lemma of Alon [Al].

LEMMA 4.3 (Alon). *Let Y be a nonempty set of points in $PG(d, q)$. The number of lines that Y misses is at most $O(nq/|Y|)$.*

COROLLARY 4.1. *If $|Y| = \lfloor n/i^{1-\varepsilon} \rfloor$, then Y misses at most $O(qi^{1-\varepsilon})$ lines.*

Let a and b be upper bounds on the number of lines missed by W_{z_i} and $W_{z_{i-1}}$, respectively, as implied by Alon's lemma. As the sequence s_i is decreasing, the term $\sum_{i=0}^{k} t_i\sqrt{s_i}$ is maximized when $t_i = b - a$. Hence, $t_i = O(q((i+1)^{1-\varepsilon} - i^{1-\varepsilon}))$. Combining the bounds for s_i and t_i we obtain

$$t_i\sqrt{s_i} = O\left(n \frac{((i+1)^{1-\varepsilon} - i^{1-\varepsilon})^{3/2}}{(((i+1)i)^{1-\varepsilon})^{1/2}} \right).$$

The following lemma bounds the term $(t_i\sqrt{s_i})/n$.

LEMMA 4.4. *For $\delta < \varepsilon/2$*

$$\frac{t_i\sqrt{s_i}}{n} = \frac{((i+1)^{1-\varepsilon} - i^{1-\varepsilon})^{3/2}}{(((i+1)i)^{1-\varepsilon})^{1/2}} < \frac{1}{i^{1+\delta}}.$$

Proof. First, we estimate $\alpha = (i+1)^{1-\varepsilon} - i^{1-\varepsilon}$,

$$\alpha = (i+1)^{1-\varepsilon} - i^{1-\varepsilon} = (i+1)^{1-\varepsilon}\left(1 - \left(1 - \frac{1}{i+1}\right)^{1-\varepsilon}\right).$$

Using the Taylor series expansion, we get

$$\alpha \leqq (i+1)^{1-\varepsilon}\left(1 - \left(1 - \frac{2(1-\varepsilon)}{i+1}\right)\right) = \frac{2(1-\varepsilon)}{(i+1)^{\varepsilon}} < \frac{2}{i^{\varepsilon}}.$$

Now,

$$\beta = (((i+1)i)^{1-\varepsilon})^{1/2} > i^{1-\varepsilon}.$$

Therefore,

$$\frac{((i+1)^{1-\varepsilon} - i^{1-\varepsilon})^{3/2}}{(((i+1)i)^{1-\varepsilon})^{1/2}} = \frac{\alpha^{3/2}}{\beta} < \frac{4}{i^{3\varepsilon/2} i^{1-\varepsilon}} = \frac{4}{i^{1+\varepsilon/2}} < \frac{1}{i^{1+\delta}}$$

as $\delta < \varepsilon/2$. $\square$

COROLLARY 4.2. *It holds that*

$$\sum_{i=1}^{\infty} \frac{((i+1)^{1-\varepsilon} - i^{1-\varepsilon})^{3/2}}{(((i+1)i)^{1-\varepsilon})^{1/2}} < \infty.$$

LEMMA 4.5. *It holds that*

$$\sum_{j=1}^{n} |S'_j| = O(n).$$

Proof. It follows from Corollary 4.2 that $\sum_{i=0}^{k} t_i \sqrt{s_i} = O(n)$. Also, since the sets Z_i are disjoint, it follows that $\sum_{i=0}^{k} s_i \leq n$. Applying Lemma 4.2 and Proposition 4.1 completes the proof. □

Lemmas 4.1 and 4.5 yield the following theorem.

THEOREM 4.1. *The algorithm for the projective plane case requires a linear number of comparisons.*

Now suppose the base set element X is represented by points in a high-dimension projective geometry $PG(d, q)$ $(d > 2)$, and each hyperplane is a subset in the family $\mathcal{S}$. We give only an outline of the analysis of the algorithm for this case, as it is very similar to the above one.

Here a simpler sequence R can be applied in the generic algorithm to obtain linear cost: for all i, $r_i = \lfloor n/i \rfloor$. It is easy to see that the sequence Z can be computed with a linear number of comparisons (this result is analogous to Lemma 4.1).

Let $v_1, \cdots, v_s$ be a subset of the points and let $m_1, \cdots, m_t$ be a subset of the hyperplanes. Define the $t \times s$ binary matrix M, where $M_{ij} = 1$ if and only if $v_i \in m_j$.

LEMMA 4.6. *For* $1 \leq i \leq s$, *let* $c_i = \sum_{j=1}^{t} M_{ji}$ (*the number of nonzero entries in the ith column*). *Then*,

$$\sum_{i=1}^{s} c_i = O(d \cdot t \cdot s^{(d-1)/d} + d \cdot s).$$

Proof. The proof of Lemma 4.2 can be easily emulated using the following inequality:

$$\binom{c_1}{d} + \binom{c_2}{d} + \cdots + \binom{c_s}{d} \leq \binom{t}{d}.$$ □

Note that for R, $s_i \leq n/i^2$ and, by Lemma 4.3, $i \cdot t_i \leq n^{1/d}$. Therefore,

$$d \cdot t_i \cdot s_i^{(d-1)/d} \leq \frac{d \cdot n}{i^{(2(d-1))/d}}.$$

Now, when $d > 2$ is fixed we get

$$\sum_{i=0}^{k} (d \cdot t \cdot s^{(d-1)/d} + d \cdot s) = O(n),$$

and this yields the next lemma.

LEMMA 4.7. *It holds that*

$$\sum_{j=1}^{n} |S'_j| = O(n).$$

The next theorem follows from the above discussion.

THEOREM 4.2. *The number of comparisons required by the algorithm for the projective geometry* $PG(d, q)$ *is linear in* n.

5. The probabilistic case. We now present an algorithm for the set maxima problem that, with high asymptotic probability, runs in linear time under the assumption that the set family is chosen at random. The probabilistic model assumes that each element x_i belongs to a set S_j with probability p, where all the random choices are made independently. The probability $p = p(n)$ may be any arbitrary function of n, except that we exclude the degenerate cases where $p(n)$ is 0 or 1.

The algorithm is an instance of the generic algorithm with $k = \lceil \log_2 n \rceil$ and, for $1 \leqq i \leqq k$, $r_i = \lceil n/2^i \rceil$. Assuming that $n = 2^m - 1$, for some positive integer m, the element z_1 is the median of the base set, and the element z_i is the median of the base elements that are larger than z_{i-1}.

LEMMA 5.1. *The rank sequence R can be computed using $O(n)$ comparisons.*

Proof. The idea is to compute the sequence $z_1, z_2, \cdots, z_k$ in that order. In general, to compute the element z_i, we need only perform a single selection from the elements that are larger than z_{i-1} and there are $r_{i-1} - 1$ such elements. Thus, the cost of the entire computation is the following number of comparisons:

$$f_R(n) = O(r_1) + O(r_2) + \cdots + O(r_k) = O(n). \qquad \square$$

The nontrivial part of the analysis is to show that $\sum_{i=1,n} \{ | S_i' | - 1 \}$ is linear in n with high asymptotic probability. This is hard to analyze directly, so we will first demonstrate that each term in the sum is stochastically dominated by a more well-behaved random variable.

Let $m(S_i)$ denote the number of elements in X that are larger than the maximum of the set S_i; in other words, $m(S_i)$ is one less than the rank of the largest element in S_i. The random variable of interest is $C = \sum_{i=1}^{n} \{ | S_i' | - 1 \}$, which represents the cost of finding the maxima of the sets during the second stage of the generic algorithm. Recall that $S_i' = S_i \cap Z_{\ell(S_i)}$. Define $C_i = | S_i' | - 1$ as the contribution of each set S_i to C; then $C = \sum_{i=1}^{n} C_i$. The following lemma shows that the random variable C_i is stochastically dominated by a random variable with the binomial distribution $B_{m(S_i),p}$.

LEMMA 5.2. *Let $\hat{C}_i$ be a random variable that is the sum of $m(S_i)$ independent and identically distributed random variables each taking the value 1 with probability p, and value 0 with probability $1 - p$. Then, C_i is stochastically dominated by $\hat{C}_i$.*

Proof. First, observe that even though $| S_i' |$ is the sum of $h(S_i)$ independent and identically distributed Bernoulli trials, its value has been conditioned by the event that the maximum element of S_i lies in it. Thus, we know that $| S_i' |$ is at least 1. To get around this conditioning, we consider the following probabilistic game.

Suppose we repeatedly flip a coin with $P[\text{head}] = p$. Let M be the number of tails we see before the first head appears. Define l to be such that $r_{l+1} \leqq M + 1 \leqq r_l - 1$. We now continue flipping the coin until we have had a total of $r_l - 1$ coin flips. The value of the game is the random variable V, which is the total number of heads seen during the course of the game. Consider now one who observes the outcome of the coin flips only until the first head appears. We wish to determine the distribution of V from the point of view of this observer. This is equivalent to determining the distribution of V conditioned by the event that $M = m$, for some $m > 0$. It is obvious that the conditional distribution of $V - 1$ is the sum of $(r_l - 1) - (m + 1)$ Bernoulli trials, where the probability of a 1 is p. Since $r_{l+1} \leqq m + 1 \leqq r_l - 1$, it follows that $(r_l - 1) - (m + 1) \leqq m$. Thus, V is one more than the sum of α Bernoulli trials, where $\alpha \leqq m$.

Let us now apply the above analysis to the problem at hand. Suppose we start choosing the elements of S_i, starting from the largest to the smallest. Assume that the elements of X are ordered as $x_n > x_{n-1} > \cdots > x_1$. We first flip a coin with $P[\text{head}] = p$, and include x_n in S_i if we see a heads. We then repeat this for x_{n-1}, x_{n-2}, and so on.

It is clear that $m(S_i)$ is precisely the number of tails we see before the first head appears. At this point we have no knowledge of the remaining $n - (m(S_i) + 1)$ coin flips. Suppose that $r_l < m(S_i) + 1 \leq r_{l+1}$, then l is exactly $\ell(S_i)$. Clearly, the size of S'_i is the number of heads we see during the first $r_l - 1$ coin flips. The correspondence between this process and the above game is obvious, where the value of the game V corresponds to $|S'_i| = C_i + 1$. We now conclude that the conditional distribution of C_i (conditioned by the event that the largest element in S_i has rank $m(S_i) + 1$) is the same as that of the sum of $\alpha \leq m(S_i)$ Bernoulli trials with parameter p. This implies the desired result. □

Define a new cost measure $\hat{C} = \sum_{i=1}^{n} \hat{C}_i$. Since each term in the sum $\hat{C}$ stochastically dominates the corresponding term in the sum C, it follows that C is stochastically dominated by $\hat{C}$. It now suffices to prove that $\hat{C}$ is linear in n with high probability. Also, note that each $\hat{C}_i$ is independent and distributed with the binomial distribution $B_{m(S_i),p}$. It then follows that $\hat{C}$ is also binomially distributed as $B_{M,p}$, where $M = \sum_{i=1}^{n} m(S_i)$. Thus, if we knew the value of each $m(S_i)$, and hence of M, the distributions of the new cost measures would be completely determined. Unfortunately, M itself is a random variable and this makes even the task of analyzing $\hat{C}$ nontrivial.

Let us first totally characterize the distribution of M using the following lemma. We assume below that $n \to \infty$.

LEMMA 5.3. *The random variables $m(S_i)$ are independent and identically distributed (i.i.d.) with a common distribution, which is geometric with parameter p. Moreover, the random variable M has the probability generating function (p.g.f.)*

$$g(z) = \left(\frac{p}{1 - qz}\right)^n,$$

where $q = 1 - p$.

Proof. The random choices of the elements of a set S_i are independent and identical, viz., Bernoulli trials with probability p. Clearly, for $1 \leq d \leq n$,

$$\Pr[m(S_i) = d] = q^d p.$$

Since we have assumed that $n \to \infty$, this is precisely the geometric distribution. Therefore, each random variable $m(S_i)$ has the probability generating function $p/(1 - qz)$. Since these are i.i.d. random variables, it follows that the p.g.f. of their sum M is the product of their p.g.f.'s [Fe, p. 264]. □

We can now derive some information about the distribution of $\hat{C}$. First, observe that $E[\hat{C}] = n \cdot q/p \cdot p = nq$ because the geometric distribution has the mean q/p. Since $q < 1$, it is clear that the algorithm makes a linear number of comparisons *on the average*. To obtain sharper results we will first determine the p.g.f. for $\hat{C}$.

LEMMA 5.4. *The probability generating function (p.g.f.) for the random variable $\hat{C}$ is*

$$h(z) = \left(\frac{p}{1 - q^2 - pqz}\right)^n.$$

Proof. $\hat{C}$ is the sum of M i.i.d. Bernoulli random variables that have the common generating function $b(z) = q + pz$. We have already obtained the probability generating function $g(z)$ of M in Lemma 5.3. From [Fe, p. 287] it follows that $\hat{C}$'s generating function $h(z)$ is the composition of $g(z)$ and $b(z)$. Thus,

$$h(z) = g(b(z)) = \left(\frac{p}{1 - q(q + pz)}\right)^n.$$

□

We now use the following tail inequality for probability generating functions (see, for example, [KMP]) to obtain a high probability result.

THEOREM 5.1 ([KMP]). *Let X be a positive integer-valued random variable with the probability generating function $F(z)$. Then, for any $z \geqq 1$,*

$$\Pr[X \geqq \lambda] \leqq \frac{F(z)}{z^{\lambda}}.$$

Using this tail inequality, we obtain the following.

LEMMA 5.5. *Let $\alpha > 1$ be any positive constant. Then, for any $\varepsilon > 0$,*

$$\Pr[C \geqq \alpha nq] \leqq e^{-(\alpha-1)\varepsilon n}.$$

Proof. It suffices to prove that the above statement holds for $\hat{C}$ since it stochastically dominates C. First, note that the p.g.f. for $\hat{C}$ can be written as

$$h(z) = \left(\frac{p}{1-q(q+pz)}\right)^n = \left(\frac{1}{1+q(1-z)}\right)^n.$$

We now use the tail inequality for p.g.f.'s, with $\lambda = \alpha nq$ and $z = 1 + \varepsilon/q$. This gives us

$$\Pr[\hat{C} \geqq \alpha nq] \leqq \left(1+\frac{\varepsilon}{q}\right)^{-\alpha nq}\left(\frac{1}{1-\varepsilon}\right)^n.$$

We now obtain the desired result,

$$\Pr[\hat{C} \geqq \alpha nq] \leqq e^{-\alpha\varepsilon n}e^{\varepsilon n} = e^{-(\alpha-1)\varepsilon n}. \qquad \square$$

The next theorem follows from Lemmas 5.1 and 5.5, and the definition of the generic algorithm.

THEOREM 5.2. *The algorithm for the probabilistic set families makes a linear number of comparisons, with high asymptotic probability.*

Acknowledgments. We thank Mauricio Karchmer, Valerie King, Nati Linial, Tomasz Radzik, and Ilan Vardi for helpful discussions. The authors also thank the anonymous referees for several insightful comments that have greatly improved the presentation.

REFERENCES

[AHU] A. V. AHO, J. E. HOPCROFT, AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[Al] N. ALON, *Eigenvalues, geometric expanders, sorting in rounds, and Ramsey theory*, Combinatorica, 6, 3 (1986), pp. 207–219.

[Ba] L. M. BATTEN, *Combinatorics of Finite Geometries*, Cambridge University Press, Cambridge, UK, 1986.

[GKS] W. GODDARD, V. KING, AND L. SCHULMAN, *Optimal randomized algorithms for local sorting and set maxima*, Proc. 22nd Annual Symposium on Theory of Computing, 1990, pp. 45–53.

[GYY] R. L. GRAHAM, A. C. YAO, AND F. F. YAO, *Information bounds are weak in the shortest distance problem*, J. Assoc. Comput. Mach., 27 (1980), pp. 428–444.

[Fe] W. FELLER, *An Introduction to Probability Theory and Its Applications*, Vol. I, John Wiley, New York, 1968.

[Ko] J. KOMLÓS, *Linear verification for spanning trees*, Combinatorica, 5, 1 (1985), pp. 57–65.

[KK] C. KENYON-MATHIEU AND V. KING, *Verifying partial orders*, Proc. 21st Annual ACM Symposium on Theory of Computing, 1989, pp. 367–374.

[KMP] D. E. KNUTH, R. MOTWANI, AND B. PITTEL, *Stable husbands*, Random Structures and Algorithms, 1, 1 (1990), pp. 1–14.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 10–24, February 1992

© 1992 Society for Industrial and Applied Mathematics
002

BROADCASTING IN BOUNDED DEGREE GRAPHS*

JEAN-CLAUDE BERMOND†, PAVOL HELL‡, ARTHUR L. LIESTMAN‡, AND JOSEPH G. PETERS‡

Abstract. Broadcasting is an information dissemination process in which a message is to be sent from a single originator to all members of a network by placing calls over the communication lines of the network. Several previous papers have investigated methods to construct sparse graphs (networks) in which this process can be completed in minimum time from any originator. The graphs produced by these methods contain high degree vertices. [Liestman and Peters, *SIAM Journal on Discrete Mathematics*, 1 (1988), pp. 531–540] and [Bermond and Peyrat, *Proceedings of the 19th SE Conference on Combinatorics, Graph Theory and Computing*, Congressus Numerantium, 1988, pp. 283–292] began an investigation of graphs with fixed maximum degree in which broadcasting can be completed in near minimum time. This investigation is continued in this paper by giving lower bounds and constructing bounded degree graphs that allow rapid broadcasting. The constructions use ideas developed by Jerrum and Skyum [*IEEE Transactions on Computers*, C-33(2), 1984, pp. 190–194], which allow passing from a graph with good average case behaviour to one with good worst case behaviour. In addition, de Bruijn digraphs [de Bruijn, *Koninklijke Nederlandse Akademie Van Wetenschappen, Indagationes Mathematicae, Series A*, 49 (1946), pp. 758–764], minimum broadcast graphs, and sparse broadcast graphs [Bermond, Hell, Liestman, and Peters, *Discrete Applied Mathematics*, to appear] are used. The resulting graphs yield the best broadcasting time known for bounded degree graphs. Also obtained are asymptotic upper and lower bounds for broadcasting time, as the maximum degree increases.

Key words. broadcasting, graphs, networks, bounded degree graphs, de Bruijn digraphs

AMS(MOS) subject classifications. 05C99, 68M10, 68R10, 94A05

1. Introduction.

Broadcasting refers to the process of message dissemination in a communication network whereby a message, originated by one member, is transmitted to all members of the network. Broadcasting is accomplished by placing a series of calls over the communication lines of the network. This is to be completed as quickly as possible subject to the constraints that each call involves only two vertices, each call requires one unit of time, a vertex can participate in only one call per unit of time, and a vertex can only call a vertex to which it is adjacent.

Given a connected graph G and a message originator, vertex u, the *broadcast time of vertex* u $b(u)$ is the minimum number of time units required to complete broadcasting from vertex u. It is easy to see that for any vertex u in a connected graph G with n vertices, $b(u) \geqq \lceil \log_2 n \rceil$, since the number of informed vertices can at most double during each time unit. The *broadcast time of a graph* G $b(G)$ is defined as the maximum broadcast time of any vertex u in G, i.e., $b(G) = \max \{ b(u) \mid u \in V(G) \}$. For the complete graph K_n with $n \geqq 2$ vertices, $b(K_n) = \lceil \log_2 n \rceil$, yet K_n is not minimal with respect to this property for any $n \geqq 3$. That is, we can remove edges from K_n and still have a graph G with n vertices such that $b(G) = \lceil \log_2 n \rceil$.

The *broadcast function*, $B(n)$, is the minimum number of edges in any graph on n vertices such that each vertex in the graph can broadcast in minimum time, that is, in time $\lceil \log_2 n \rceil$. A *minimum broadcast graph* (mbg) is a graph G on n vertices having $B(n)$

* Received by the editors August 17, 1988; accepted for publication (in revised form) January 10, 1991. This research was supported by Natural Sciences and Engineering Research Council of Canada grants A-5057, A-1734, and A-0322.

† I3S, CNRS, Bât 4, rue A. Einstein, Sophia-Antipolis, 06560 Valbonne, France. This article was written while the author was visiting the School of Computing Science, Simon Fraser University, and supported by the Advanced Systems Institute and Simon Fraser University.

‡ School of Computing Science, Simon Fraser University, Burnaby, British Columbia V5A 1S6, Canada.

edges and $b(G) = \lceil \log_2 n \rceil$. From an applications perspective, mbg's represent the cheapest possible communication networks (having the fewest communication lines) in which broadcasting can be accomplished from any vertex as fast as is theoretically possible.

For a survey of results on broadcasting and related problems, see Hedetniemi, Hedetniemi, and Liestman [14]. Slater, Cockayne, and Hedetniemi [24] showed that given an arbitrary graph G, vertex v, and $k \geq 4$ as input, deciding whether $b(v) \geq k$ is NP-complete. In [11] Farley, Hedetniemi, Mitchell, and Proskurowski studied $B(n)$. In particular, they determined the values of $B(n)$ for $n \leq 15$ and noted that $B(2^k) = k2^{k-1}$ (the k-cube is an mbg on $n = 2^k$ vertices). Mitchell and Hedetniemi [22] determined the value for $B(17)$; Wang [25] found the value of $B(18)$; and Bermond, Hell, Liestman, and Peters [3] found the values of $B(19)$, $B(30)$, and $B(31)$. These studies suggest that mbg's are extremely difficult to find; in fact, no mbg with n vertices is known for any value of $n > 32$, except for the easy values of $n = 2^k$, where the k-cube can be used and the recently discovered family of graphs with $n = 2^k - 2$ [8].

Since mbg's seem to be difficult to find, several authors have devised methods to construct sparse graphs that allow minimum time broadcasting from each vertex. We use the term *sparse broadcast graph* (sbg) to denote a graph G on n vertices with "close to" $B(n)$ edges such that $b(G) = \lceil \log_2 n \rceil$. In [10] Farley designed several techniques for constructing sparse broadcast graphs with n vertices and approximately $n/2 \log_2 n$ edges, for arbitrary values of n. Chau and Liestman [5] presented constructions based on Farley's techniques, which yield somewhat sparser graphs for most values of n. In [13] Grigni and Peleg showed that $B(n) \in \Theta(L(n)n)$ for $n \geq 1$, where $L(n)$ denotes the exact number of consecutive leading 1's in the binary representation of $n - 1$. Recently, Gargano and Vaccaro [12] gave constructions that produce the best of the known graphs for some large values of n. Asymptotically, Grigni and Peleg's construction (which establishes their upper bound) produces the best of the known graphs for most values of n.

So far, the emphasis in this research has been on obtaining sparse graphs in which each vertex can broadcast in minimum time. If these graphs are to be used in the design of actual networks, other considerations may override the need for minimum time broadcasting. In particular, the constructions of Farley, of Chau and Liestman, and of Gargano and Vaccaro result in graphs with n vertices and average degree $O(\log_2 n)$, while the construction of Grigni and Peleg yields n vertex graphs with some vertices of degree $\log_2 \log_2 n + L(n)$. It may be more realistic to use a graph with fixed maximum degree (see [1], [2], and [15]) in which every vertex can broadcast "quickly." We will use the term *bounded degree broadcast graph* (bdbg) to describe a graph G on n vertices with maximum degree Δ such that $b(G)$ is "close to" $b(n, \Delta) = \min \{ b(H) \mid H \text{ has } n \text{ vertices and max degree } \Delta \}$. (Questions related to broadcasting in slightly more than minimum time have previously been addressed by Liestman [19] and by Grigni and Peleg [13].)

In a recent paper, Liestman and Peters [20] investigated bounded degree broadcast graphs with maximum degrees 3 and 4. They gave lower bounds on the time required to broadcast in such graphs and presented several constructions that produce good bounded degree broadcast graphs. Liestman and Peters showed that $b(n, 3) \geq 1.440 \log_2 n - 1.769$ and that if n is a power of 2, then $b(n, 3) \leq 2 \log_2 n + 1$. The upper bound is achieved by constructing folded-shuffle-exchange graphs [6]. They also showed that $b(n, 4) \geq 1.137 \log_2 n - 0.637$ and that if n is a power of 4, then $b(n, 4) \leq 1.625 \log_2 n + 2.25$. The upper bound in this case is achieved by constructing folded-4-shuffle-exchange graphs. More recently, Bermond and Peyrat [4] considered broadcasting in de Bruijn and Kautz graphs. They were able to improve on the upper bounds of Liestman and Peters, showing, in particular, that $b(n, 4) \leq 1.5 \log_2 n + 1$ when n is either a power of 2 or 3 times a power of 2.

In this paper, we improve on the results of Liestman and Peters [20] and of Bermond and Peyrat [4]. In § 2 we present general lower bounds on the time required to broadcast in bounded degree graphs. In § 3 we give the definition of de Bruijn digraphs and summarize some of the work of Bermond and Peyrat [4], which will be useful in later sections. In § 4 we give some examples of constructions that motivate the formal definitions of compound graphs in § 5. In § 6 we describe how to broadcast in compound graphs, generalizing the work of Bermond and Peyrat [4] reported in § 3. Finally, in § 7 we report the best of the known upper bounds on the time required to broadcast in bounded degree graphs. All of these bounds are achieved by using compound graphs. In particular, it will follow from our results that $b(n, 3) \leqq 1.875 \log_2 n + 2.903$ when n is 6 times a power of 4 and that $b(n, 4) \leqq 1.417 \log_2 n + 4$ when n is a power of 8.

2. Lower bounds. We wish to prove a lower bound on $b(n, \Delta)$, the minimum $b(G)$ for any graph G with n vertices and maximum degree Δ. It will be more convenient to first consider the quantity a_t^Δ, which denotes the maximum number of vertices in a graph of maximum degree Δ in which each vertex can inform all others in time t. An upper bound on a_t^Δ will clearly translate to a lower bound on $b(n, \Delta)$. In any broadcasting scheme that achieves this maximum a_t^Δ we may assume that a vertex does not remain idle if it has already been informed and it still has uninformed neighbours. Therefore, if there is one informed vertex at time $t = 0$, then after time $t + \Delta - 1$, $t \geqq 1$, all the vertices that were informed by time t have informed all of their neighbours and must become idle. Hence, $a_{t+\Delta}^\Delta \leqq a_{t+\Delta-1}^\Delta + (a_{t+\Delta-1}^\Delta - a_t^\Delta) = 2a_{t+\Delta-1}^\Delta - a_t^\Delta$. It is also clear that $a_s^\Delta \leqq 2^s$ because at each time a vertex can inform at most one other vertex. Thus an upper bound on a_t^Δ is the solution to the recurrence

$$(1) \qquad b_s^\Delta = 2^s \text{ for } s = 0, 1, \cdots, \Delta, \qquad b_{t+\Delta}^\Delta = 2b_{t+\Delta-1}^\Delta - b_t^\Delta \quad \text{for all } t \geqq 1.$$

Note that $a_s^\Delta = 2^s$ for $s = 0, 1, \cdots, \Delta$ since the s-cube, with $s \leqq \Delta$, is of degree at most Δ and has 2^s vertices. It is possible that $a_t^\Delta = b_t^\Delta$ for all t.

For $\Delta = 3$, we know that $a_s^3 = b_s^3 = 2^s$, for $0 \leqq s \leqq 3$. Furthermore, $a_4^3 = b_4^3 = 14$, as the Heawood graph (Fig. 7.1(a)) is a cubic graph on 14 vertices with broadcast time 4. Similarly, $a_5^3 = b_5^3 = 24$, as we presented in [3] a cubic graph on 24 vertices with broadcast time 5. Also, $a_6^3 = b_6^3 = 40$, as we presented a cubic graph on 40 vertices with broadcast time 6 in [3]. The next value in the sequence is $b_7^3 = 66$. We do not know whether there is a cubic graph on 66 vertices with broadcast time 7, i.e., if $a_7^3 = 66$.

For $\Delta = 4$, we know that $a_s^4 = b_s^4 = 2^s$, for $0 \leqq s \leqq 4$. For $s = 5$, $a_5^4 = b_5^4 = 30$, as we presented (three) 4-regular graphs on 30 vertices with broadcast time 5 in [3]. Moreover, $a_6^4 = b_6^4 = 56$, as we presented a 4-regular graph on 56 vertices with broadcast time 6 in [3]. The next value in the sequence is $b_7^4 = 104$ and we do not know whether $a_7^4 = 104$.

For $\Delta = 5$, we know that $a_6^5 = b_6^5 = 62$ since there is a 5-regular graph on 62 vertices with broadcast time 6 [23]. This graph is constructed by adding chords to a cycle of length 62. In particular, when vertices are numbered consecutively around the cycle, chords are added from each even-numbered vertex x to vertices $(x + 5) \bmod 62$, $(x - 7) \bmod 62$, and $(x + 19) \bmod 62$.

Very recently, Dinneen, Fellows, and Faber [8] solved the conjecture that $a_t^\Delta = b_t^\Delta$ for $t = \Delta + 1$ (see [3]) by constructing a Δ-regular broadcast graph with $2^{\Delta+1} - 2$ vertices for each $\Delta \geqq 3$. It is easy to show that these graphs are minimum broadcast graphs. At present, we do not know other values of $t > \Delta$ for which $a_t^\Delta = b_t^\Delta$.

We suspect that $a_t^\Delta = b_t^\Delta$ for all $t \geqq \Delta$, or at least that the values are very close. At present, this seems difficult to prove. For example, for $\Delta = 3$ this would mean that

$b(n, 3) \approx 1.440 \log_2 n$. However, the best construction for n-vertex cubic graphs with small diameter (see [16] or [6]) gives graphs with diameter $1.472 \log_2 n$. Since the diameter of a graph is an obvious lower bound for its broadcast time, this means that if we could show that $b(n, 3) \approx 1.440 \log_2 n$, then the cubic graphs that achieve this bound would also improve the results for the diameter problem.

Note that $b(n, \Delta)$ is at least as large as the minimum t for which $a_t^\Delta \geqq n$. The best lower bound on $b(n, \Delta)$ that can be obtained from (1) is $L_\Delta(n) = \min\{t : b_t^\Delta \geqq n\}$. If $a_t^\Delta = b_t^\Delta$, for all t, then $L_\Delta(n)$ would be the true optimum time for broadcasting in graphs with n vertices and maximum degree Δ. In [20], the recurrence (1) was derived and used to determine that $b(n, 3) \geqq L_3(n) \approx 1.440 \log_2 n - 1.769$ and $b(n, 4) \geqq L_4(n) \approx 1.137 \log_2 n - 0.637$. Iterated numerical techniques can be used to find other values of c_Δ such that $L_\Delta(n) \approx c_\Delta \log_2 n$. Table 2.1 lists values of c_Δ for $3 \leqq \Delta \leqq 16$ that were obtained this way. In contrast, the following careful analysis of the recurrence relation (1) gives the asymptotic behaviour of $L_\Delta(n)$ and, therefore, the best general lower bound for $b(n, \Delta)$ obtainable from (1).

THEOREM 1. *For every $\varepsilon > 0$ and all sufficiently large n and Δ,*

$$\left(1 + \frac{\log_2 e}{2^\Delta}\right) \log_2 n + O(1) \leqq L_\Delta(n) < \left[1 + \frac{(1+\varepsilon)\log_2 e}{2^\Delta}\right] \log_2 n.$$

Remark. We interpret Theorem 1 to say that $c_\Delta \approx 1 + (\log_2 e)/2^\Delta$. We have listed the values of $1 + (\log_2 e)/2^\Delta$ in Table 2.1.

COROLLARY 2. *$b(n, \Delta) \geqq (1 + (\log_2 e)/2^\Delta) \log_2 n + O(1)$ for all Δ and all sufficiently large n.*

Corollary 2 follows from the proof of Theorem 1.

Proof of Theorem 1. For simplicity of notation, let $\Delta = d + 1$. To solve (1), consider its characteristic equation $x^{d+1} - 2x^d + 1 = 0$. It is known (see Miles [21]) that this equation has $d + 1$ distinct roots $r_0, r_1, \cdots, r_d$; that two of these roots are real, say $r_0 = 1$ and $r_1 = r$, with $1 < r < 2$; and that the remaining $d - 1$ roots are complex and lie inside the unit disk of the complex plane. (In fact, Miles studied the equation $x^d - x^{d-1} - \cdots - 1 = 0$. To apply his results we need only note that $x^{d+1} - 2x^d + 1 = (x - 1)(x^d - x^{d-1} - \cdots - 1)$.) By a standard technique [21], there exist complex numbers $s_0, s_1, \cdots, s_d$ so that $b_t^\Delta = s_0 r_0^t + s_1 r_1^t + \cdots + s_d r_d^t$. Since $|r_i| \leqq 1$ for $i \neq 1$

TABLE 2.1
Lower bounds.

Δ	c_Δ	$1 + (\log_2 e)/2^\Delta$
3	1.440420	1.180337
4	1.137467	1.090168
5	1.056215	1.045084
6	1.025404	1.022542
7	1.012034	1.011271
8	1.005842	1.005636
9	1.002874	1.002818
10	1.001424	1.001409
11	1.000709	1.000704
12	1.000353	1.000352
13	1.000176	1.000176
14	1.000088	1.000088
15	1.000044	1.000044
16	1.000022	1.000022

and $r_1 = r > 1$, we have $b_t^\Delta = \Theta(r^t)$. Thus it is important to estimate r. (We have just discovered that a result similar to Lemma 3 is described by Knuth [18].)

LEMMA 3. $2 - (1+\varepsilon)/2^d < r < 2 - 1/2^d$ *for all* $\varepsilon > 0$ *and all sufficiently large* d.

Proof. Let $r = 2 - \delta$. Since r is the root of $x^{d+1} - 2x^d + 1 = 0$, we have $\delta(2-\delta)^d = 1$. We will show that $1/2^d < \delta < (1+\varepsilon)/2^d$. Both of these inequalities follow from the fact that $(1/2^d)(2 - 1/2^d)^d < 1$ and $((1+\varepsilon)/2^d)(2-(1+\varepsilon)/2^d)^d > 1$. The first fact is obvious. The second fact is equivalent to $(1+\varepsilon)(1-(1+\varepsilon)/2^{d+1})^d > 1$. For $y = (1+\varepsilon)/2^{d+1}$, we can use the estimate $1 - y > e^{-2y}$ (which is valid for $y \in (0, 1/2]$) to obtain $(1-y)^d > e^{-2dy} = e^{-d}((1+\varepsilon)/2^d) > 1/(1+\varepsilon)$, when $d/2^d < (\ln(1+\varepsilon))/(1+\varepsilon)$. This certainly holds for any fixed ε and d large enough.

To complete the proof of Theorem 1 we need to bound $L_\Delta(n)$. Let Δ be fixed and let $\alpha = 2 - 1/2^d$ be the upper bound from Lemma 3. If $b_t^\Delta \geqq n$ and n is sufficiently large, then t will also be large enough so that $b_t^\Delta \leqq c_1\alpha^t$ (for some positive c_1). Thus $c_1\alpha^t \geqq n$ and $t \geqq (1/\log_2 \alpha)\log_2 n + O(1)$. This means that $L_\Delta(n) \geqq (1/\log_2 \alpha)\log_2 n + O(1)$. Now since $\log_2 \alpha = 1 + \log_2 e \ln(1 - 1/2^{d+1})$, we can use the fact that $1/(1 + \log_2 e \ln(1-y)) > 1 + (\log_2 e)y$ for $0 < y < 1$ to obtain $L_\Delta(n) \geqq (1 + \log_2 e/2^\Delta)\log_2 n + O(1)$ for any fixed Δ.

For given n and Δ, let t be such that $b_t^\Delta \geqq n$ and $b_{t-1}^\Delta < n$ and let $\beta = 2 - ((1+\varepsilon)/2^d)$ be the lower bound from Lemma 3. If n is sufficiently large, then t will be large enough so that $c_2\beta^{t-1} \leqq b_{t-1}^\Delta$ (for some $c_2 > 0$). Thus $c_2\beta^{t-1} < n$ and $t < (1/\log_2 \beta)\log_2 n + O(1)$. This means that $L_\Delta(n) < (1/\log_2 \beta)\log_2 n + O(1)$. Now since $\log_2 \beta = 1 + \log_2 e \ln(1 - ((1+\varepsilon)/2^{d+1}))$, we can use the fact that for any ε', $0 < \varepsilon' < 1$, there exists $f(\varepsilon')$ such that if $0 < y < f(\varepsilon')$, $1/(1 + \log_2 e \ln(1-y)) < 1 + (1+\varepsilon')(\log_2 e)y$ to obtain $L_\Delta(n) < (1 + ((1+\varepsilon)(1+\varepsilon')\log_2 e)/2^\Delta)\log_2 n + O(1)$ for Δ large enough. Therefore, for any ε'' such that $(1+\varepsilon'') > (1+\varepsilon)(1+\varepsilon')$, and for any Δ and n sufficiently large we get $L_\Delta(n) < (1 + ((1+\varepsilon'')\log_2 e)/2^\Delta)\log_2 n$. □

As mentioned above, it may be the case that $L_\Delta(n)$ is the optimum broadcast time in graphs with maximum degree Δ. However, at present, the best of our constructions only achieves $b(n, \Delta) \leqq (1 + c'/\Delta)\log_2 n$ asymptotically with $c' \approx .415$ (see § 7).

3. Broadcasting in de Bruijn digraphs. In the remainder of the paper, we show how to construct bounded degree graphs that allow rapid broadcasting. Our constructions make use of de Bruijn digraphs, which were defined in [7]. The *de Bruijn digraph* $B(d, D)$ with indegree and outdegree d and diameter D is the digraph whose d^D vertices are the words of length D on an alphabet of d letters (we will always use the alphabet $\{0, 1, \cdots, d-1\}$). There is an arc from a vertex x to a vertex y if and only if the last $D-1$ letters of x are the same as the first $D-1$ letters of y; that is, there are arcs from $(x_1, \cdots, x_D)$ to the vertices $(x_2, \cdots, x_D, \lambda)$ where λ is any letter of the alphabet. Note that the diameter of $B(d, D)$ is D because $(x_1, \cdots, x_D)$, $(x_2, \cdots, x_D, z_1)$, $(x_3, \cdots, x_D, z_1, z_2)$, $\cdots$, $(x_D, z_1, \cdots, z_{D-1})$, $(z_1, \cdots, z_D)$ is a directed path of length D joining any vertex $(x_1, \cdots, x_D)$ to any other vertex $(z_1, \cdots, z_D)$.

We use $UB(d, D)$ to denote the associated *undirected de Bruijn graph*. That is, $UB(d, D)$ is the graph whose vertices are the words of length D on an alphabet of d letters in which the vertex $(x_1, \cdots, x_D)$ is adjacent to the vertices $(x_2, \cdots, x_D, \lambda)$ (called the *right neighbours* of $(x_1, \cdots, x_D)$) and the vertices $(\lambda, x_1, \cdots, x_{D-1})$ (called the *left neighbours* of $(x_1, \cdots, x_D)$). It is clear that $UB(d, D)$ has maximum degree $\Delta = 2d$.

Bermond and Peyrat [4] recently investigated broadcasting in de Bruijn graphs and presented three different broadcasting schemes. One of their schemes proves that $b(UB(d, D)) = ((d+1)/2)D + d/2$. One of their other schemes (which achieves a slightly different bound) is useful in understanding the broadcasting scheme for

compound graphs, which follows in § 6. It is the following scheme in which each vertex (including the originator) only sends the message to its right neighbours. A vertex $(x_1, \cdots, x_D)$ informs its neighbours in an order that depends on $\delta \equiv [\sum_{i=1}^{D} x_i] \bmod d$, the d-arity of the label $(x_1, \cdots, x_D)$. Vertex $(x_1, \cdots, x_D)$ sends the message to its uninformed neighbours in the order $(x_2, \cdots, x_D, \delta)$, $(x_2, \cdots, x_D, (\delta + 1)), \cdots, (x_2, \cdots, x_D, (\delta - 1))$ (addition/subtraction modulo d). As shown in [4], the message will arrive at any vertex $(z_1, \cdots, z_D)$ from $(x_1, \cdots, x_D)$ after at most $((d + 1)/2)(D + 1)$ time units along one of the d paths $(x_1, \cdots, x_D)$, $(x_2, \cdots, x_D, \alpha)$, $(x_3, \cdots, x_D, \alpha, z_1), \cdots, (x_{i+2}, \cdots, x_D, \alpha, z_1, \cdots, z_i), \cdots$, $(\alpha, z_1, \cdots, z_{D-1}), (z_1, \cdots, z_D)$, where $0 \leqq \alpha \leqq d - 1$.

Strictly speaking, in the above scheme the message may arrive at some vertex more than once. By deleting redundant calls, a scheme can be obtained that completes the broadcast at the appropriate time.

4. Introduction to compound graphs. Our goal is to construct graphs with bounded degree and good (asymptotic) broadcasting time. To do this, we will use the notion of compound graphs (see [2]). In particular, we will use the ideas developed by Jerrum and Skyum [16] to construct bounded degree graphs with the best diameter currently known. (It appears that a bounded degree graph with smallest diameter does not necessarily give the best broadcasting time.) In this section, we describe some ad hoc constructions that give some insight into the details that are addressed more formally in later sections.

To construct a bounded degree graph that may have good broadcasting time, begin with a "good" digraph B, such as a de Bruijn digraph. Replace each vertex x of B by a copy G_x of a "suitable" graph G and join the copies using the arcs of B. (If there is an arc from x to y in B, then put an arc from G_x to G_y as described below.) The associated undirected graph $G[B]$ is called the *compound* of G in B. We will, however, find it useful to refer to the directions of the arcs when describing the construction of a specific compound graph or broadcast schemes for it.

For example, let $B = B(2, D)$, the de Bruijn digraph with indegree 2, outdegree 2, and diameter D, and let G be a single edge ($G = K_2$). Replace each vertex x in B by two new vertices $(x; 0)$ and $(x; 1)$, joined by an edge in the new graph $G[B]$. The arcs incident on x in B are redirected in $G[B]$ so that each vertex $(x; j)$ for $j = 0, 1$ has one incoming arc and one outgoing arc. (Figure 4.1(a) shows a vertex x in B and its incident edges. Figure 4.1(b) shows the corresponding vertices $(x; 0)$ and $(x; 1)$ of the new graph $G[B]$ and their incident edges.) The resulting graph is $H = K_2[B(2, D)]$.

Suppose we want to broadcast from vertex $(x; j)$ in this new graph H. At time 1, send the message to $(x; j + 1)$ (addition modulo 2), the other vertex in the originator's copy of K_2. At time 2, send the message to the two copies $G_{x'}$ and $G_{x''}$ where xx' and xx'' are arcs of B; that is, if $x = (x_1, \cdots, x_D)$, then $x' = (x_2, \cdots, x_D, 0)$ and $x'' =$

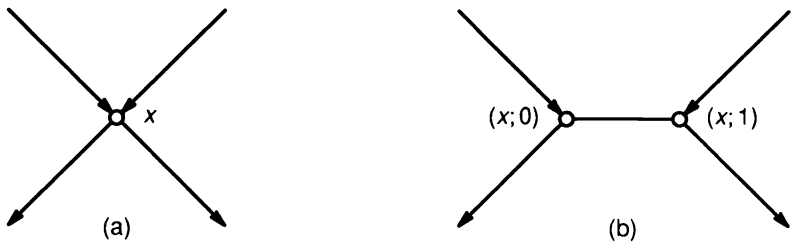


FIG. 4.1. *Replacing a vertex with an edge.*

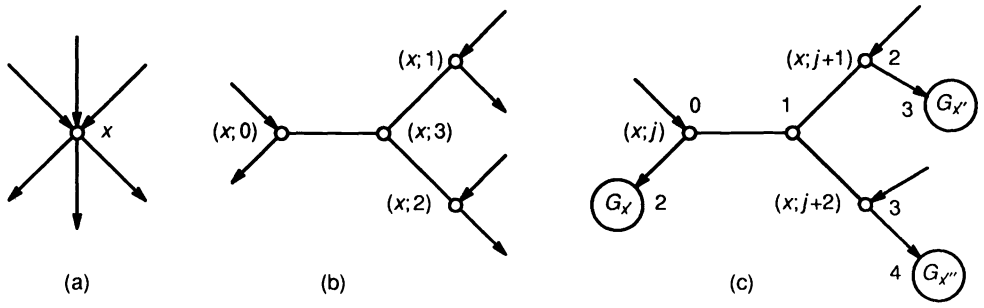


FIG. 4.2. *Replacing a vertex with a 3-star.*

$(x_2, \cdots, x_D, 1)$. If a message arrives at time t in some copy G_z, then at time $t + 1$ send it to the other vertex of G_z, and at time $t + 2$ send it to the two copies $G_{z'}$ and $G_{z''}$, where zz' and zz'' are arcs of B. Since the diameter of $B(2, D)$ is D, at time $2D$ the message will have reached all copies of G, and at time $2D + 1$ all the vertices of $K_2[B(2, D)]$ are informed.

We have constructed a graph H on $n = 2 \times 2^D = 2^{D+1}$ vertices that is 3-regular and has broadcast time $b(H) \leqq 2D + 1 = 2 \log_2 n - 1$. This simple construction allows us to match the result of Liestman and Peters [20] for graphs of maximum degree 3. As we will see, this technique will enable us to make further improvements.

Note that in the previous example we did not specify which of the two arcs leaving x will be associated (in the compound graph) with $(x; 0)$ and which with $(x; 1)$. In this case it does not matter as some vertex in both copies $G_{x'}$ and $G_{x''}$ will receive the message two units of time after it arrives in copy G_x. However, this need not always be the case, as is illustrated by the next example.

Let $B = B(3, D)$, the de Bruijn digraph with indegree 3, outdegree 3, and diameter D. Replace each vertex x in B with a copy of a 3-star G consisting of vertices $(x; j)$, where $j = 0, 1, 2, 3$ and edges $(x; 0)(x; 3)$, $(x; 1)(x; 3)$, and $(x; 2)(x; 3)$. The arcs incident on x in B are distributed among the vertices $(x; j)$ for $j = 0, 1, 2$ in $G[B]$ so that each vertex has one incoming arc and one outgoing arc (see Figs. 4.2(a) and 4.2(b)).

To broadcast in $G[B]$ we can use the following scheme. If a message arrives (or originates) at time t at vertex $(x; j)$ in some copy G_x, send it to $(x; 3)$ at time $t + 1$. At time $t + 2$, $(x; 3)$ sends it to $(x; j + 1)$ and $(x; j)$ sends it to the copy $G_{x'}$ joined to G_x by the arc going out from $(x; j)$. At time $t + 3$, $(x; 3)$ sends it to $(x; j + 2)$ while

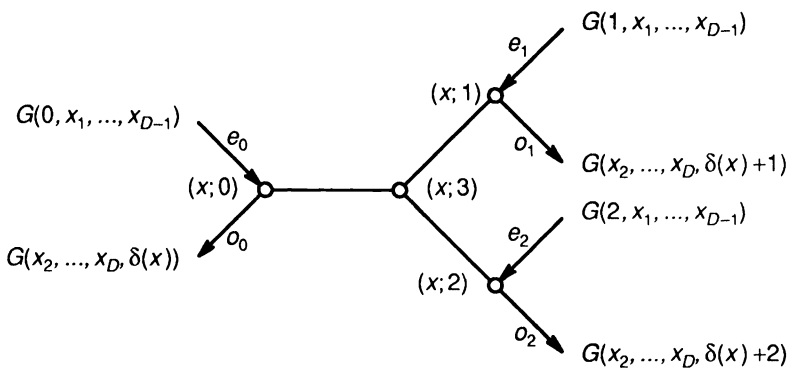


FIG. 4.3. *Connections for G_x with $x = (x_1, \cdots, x_D)$.*

$(x; j+1)$ sends it to the copy $G_{x''}$ joined to G_x by the arc going out from $(x; j+1)$. At time $t+4$, $(x; j+2)$ sends it to the copy $G_{x'''}$ joined to G_x by the arc going out from $(x; j+2)$. (All additions on vertex labels are performed modulo 3.) This is illustrated in Fig. 4.2(c), where the vertex labels denote the delay in transmitting the message from $(x; j)$. Even without specifying exactly how the arcs are connected, we can see that the constructed graph $H' = G[B]$ has broadcast time $b(H') \leqq 4D + 3$.

If we are more specific about the connections in H', the bound can be improved. We will use e_j to label the arc entering the vertex $(x; j)$ and o_j to label the arc leaving $(x; j)$, as shown in Fig. 4.3. (Of course, these labels are relative to a particular G_x.) For any $G_{(x_1,\cdots,x_D)}$, e_j will come from $G_{(j,x_1,\cdots,x_{D-1})}$. The arc o_j will join $(x; j)$ to $G_{(x_2,\cdots,x_D,\delta(x)+j)}$, where $\delta(x) \equiv [\sum_{i=1}^{D} x_i] (\mathrm{mod}\ 3)$ is the 3-arity of x. We will see in § 7 that the broadcast time $b(H')$ is at most $3D + 6$ with these connections.

5. Construction of the compound graph $G[B]$. We now formally describe the construction of $G[B]$, the compound of G in B. Let G be a graph on p vertices and let $B = B(d, D)$ be the de Bruijn digraph of outdegree d, indegree d, and diameter D. The vertices of $B(d, D)$ are labelled $(x_1, \cdots, x_D)$ with $x_i \in \{0, 1, \cdots, d-1\}$. $G[B]$ is obtained by replacing each vertex x of B by a copy of G (denoted G_x) and by associating with each arc xy of B an edge between G_x and G_y, as described below.

The vertices of $G[B]$ are labelled $(x; j)$, where $x = (x_1, \cdots, x_D)$ is a vertex of B, and j is a vertex of G. The set of vertices of the copy G_x is the set of all vertices $(x; j)$ with $j \in V(G)$. There are d arcs entering and d arcs leaving a given G_x. An arc entering G_x will be labelled e_i if it comes from $G_{(i,x_1,\cdots,x_{D-1})}$. Now we can assign the d in-arcs to the vertices of G_x in a uniform way for all copies of G by giving a mapping g of $\{0, 1, \cdots, d-1\}$ into $V(G)$ so that the arc e_i enters G_x by way of the vertex $(x; g(i))$.

We label the outgoing arcs o_i in each copy of G. In G_x, we let the arc o_i go to $G_{(x_2,\cdots,x_D,i+\delta(x))}$, where $\delta(x)$ is the d-arity of the vertex $x = (x_1, \cdots, x_D)$ (addition is modulo d). The d out-arcs are assigned to the vertices of each copy of G in a uniform way by giving a mapping f of $\{0, 1, \cdots, d-1\}$ into $V(G)$ so that the arc o_i starts at $(x; f(i))$.

In summary, we choose two mappings f and g from $\{0, 1, \cdots, d-1\}$ to $V(G)$. There is an edge between $(x; j)$ and $(y; k)$ if and only if either $x = y$ and j is adjacent to k in G or xy is an arc of B with $x = (x_1, \cdots, x_D)$, $y = (x_2, \cdots, x_D, x_{D+1})$, $j = f(x_{D+1} - \delta(x))$ and $k = g(x_1)$. Note that while f and g significantly influence the construction of $G[B]$, we will continue to use the simplified notation $G[B]$, and let f and g be determined by context.

Consider the following example, illustrated in Fig. 5.1. Let $G = C_4$, $d = 6$, and $f(0) = f(1) = 0, f(2) = 1, f(3) = f(4) = 2, f(5) = 3$, $g(0) = 0$, $g(1) = g(2) = 1$, $g(3) = 2$, $g(4) = g(5) = 3$. For example, e_3 will always enter $(x; 2)$, since $g(3) = 2$. Similarly, o_3 will always connect $(x; 2)$ to the copy $G_{(x_2,\cdots,x_D,\delta(x)+3)}$ since $f(3) = 2$.

The graph $G[B(d, D)]$ contains $|V(G)| \times |V(B(d, D))| = pd^D$ vertices. The degree of vertex $(x; j)$ is the sum of the degree of j in G and the number of arcs from $B(d, D)$ entering and leaving $(x; j)$, that is, $d(x; j) = d_G(j) + |f^{-1}(j)| + |g^{-1}(j)|$. (Note that this is independent of x.) The sum of the degrees of the vertices of any copy G_x is $2|E(G)| + 2d$. Note that we can always choose f and g in order to ensure a maximum degree Δ in $G[B]$ as long as $\Delta \geqq \max\{d_G(j) | j \in V(G)\}$ and $p\Delta \geqq 2|E(G)| + 2d$. Indeed, it suffices to choose f and g such that $d_G(j) + |f^{-1}(j)| + |g^{-1}(j)| \leqq \Delta$, $0 \leqq j \leqq p-1$.

In fact, for constructing bounded degree broadcast graphs, we do this backward. Given the degree Δ that we desire for the compound graph, we choose a graph

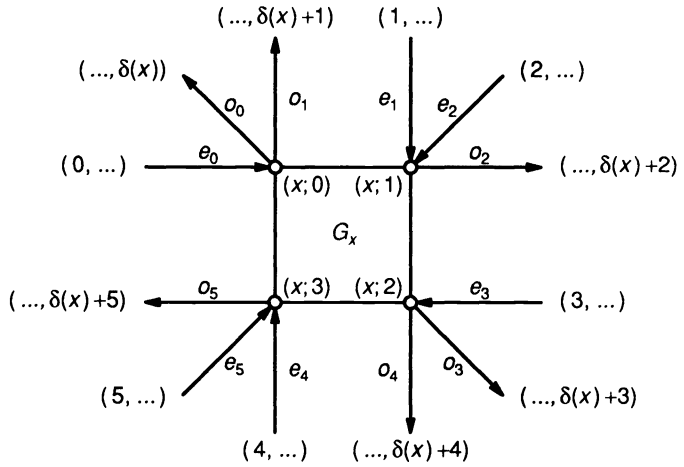


FIG. 5.1. *Replacing a vertex with a* C_4.

G with maximum degree at most Δ, and then determine d satisfying $2d \leqq p\Delta - 2|E(G)|$. To obtain the maximum possible number of vertices, we will choose $d = \lfloor (p\Delta - 2|E(G)|)/2 \rfloor$.

6. Broadcast time in $G[B]$. In the broadcast scheme for the de Bruijn digraph B described in § 3, when a message arrives at vertex $(x_1, \cdots, x_D)$, the vertex relays the message to its right neighbours $(x_2, \cdots, x_D, \lambda)$. The general idea for broadcasting in the compound graph $G[B]$ is to simulate this scheme. In $G[B]$, when a message first arrives at a vertex $((x_1, \cdots, x_D); j)$ (for some vertex j of G), we inform the other members of this copy of G and relay the message to the "out-neighbour copies" of G that replaced the neighbours of $(x_1, \cdots, x_D)$.

To determine the broadcast time of $G[B]$, we introduce a new parameter $\bar{b}(G)$, the smallest possible "average time" needed to transmit a message originated in a copy of G to its out-neighbour copies. Formally, let G be a graph to which d outgoing arcs $o_0, \cdots, o_{d-1}$ have been attached. (These arcs will correspond to the arcs o_i defined in $G[B]$.) Note that although $o_0, \cdots, o_{d-1}$ significantly influence the value of $\bar{b}(G)$, we prefer not to make them part of the notation $\bar{b}(G)$. For a vertex u of G and a particular broadcasting scheme for u in G, let t_u^i denote the time at which a message originated at u at time 0 will be received by the vertex at the other end of arc o_i. In the compound graph, this is an upper bound on the time at which the message will reach the copy connected to G by o_i. The value $\bar{t}_u = (1/d)(t_u^0 + \cdots + t_u^{d-1})$ is the average time for a message originated at u to arrive at the copies of G along the arcs o_i under the given broadcast scheme. If we let $\bar{b}(G, u)$ be the minimum of $\bar{t}_u$ over all possible broadcasting schemes for originator u in G, then $\bar{b}(G) = \max\{\bar{b}(G, u) | u \in V(G)\}$.

The examples of Fig. 6.1 illustrate how $\bar{b}(G)$ is calculated. In each case, the vertex labelled 0 is the originator. The label on each other vertex indicates the time at which the vertex receives the message (assuming that the first call is received at time 1). The label at the end of an arc indicates the time at which the message reaches the corresponding copy of G. Figure 6.1(a) shows the times for $G = K_2$. Since both adjacent copies receive the message at time 2, $\bar{b}(K_2) = 2$. Figure 6.1(b) shows that if G is the 3-star, the neighbouring copies receive the message at times 2, 3, and 4. Thus, $\bar{b}(G) = 3$ for this graph G. Figure 6.1(c) shows the time for $G = T_6$, where T_6 is a particular tree on six vertices. With this particular graph, the neighbouring copies receive the message at times 2, 4, 4,

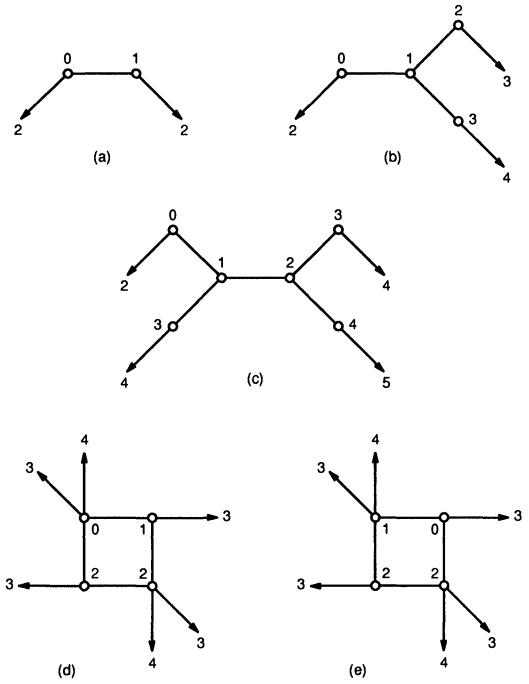


FIG. 6.1. *Average time.*

and 5 giving $\bar{b}(G) = 15/4$. Figures 6.1(d) and 6.1(e) show the times for the two possible nonisomorphic originators in the graph $G = C_4$, the cycle of four vertices. In both cases, four neighbouring copies receive the message at time 3, and the two other neighbouring copies receive the message at time 4. This gives $\bar{b}(C_4) = 20/6$. Other examples appear in § 7.

THEOREM 4. $b(G[B(d, D)]) \leqq (D + 1)\bar{b}(G) + b(G)$.

Proof. Choose, for each vertex u of G, a broadcasting scheme in G such that $\bar{b}(G, u) \leqq \bar{b}(G)$. Using this scheme, if a message arrives at the vertex $(x; u)$ in G_x at time t, it will be received at the other end of the arc o_i at time $t + t_u^i$ and we obtain a scheme for $G[B(d, D)]$.

Note that by the definition of $\bar{b}$, $\sum_{i=0}^{d-1} t_u^i \leqq d\bar{b}(G)$.

Consider an arbitrary originator, vertex $(x; j_0)$ with $x = (x_1, \cdots, x_D)$, and a goal copy G_y with $y = (y_1, \cdots, y_D)$. Consider also the d paths $z_0 = x = (x_1, \cdots, x_D)$, $z_1 = (x_2, \cdots, x_D, \alpha)$, $z_2 = (x_3, \cdots, x_D, \alpha, y_1), \cdots, z_i = (x_{i+1}, \cdots, x_D, \alpha, y_1, \cdots, y_{i-1}), \cdots, z_D = (\alpha, y_1, \cdots, y_{D-1}), z_{D+1} = (y_1, \cdots, y_D)$ in $B(d, D)$ with $\alpha \in \{0, 1, \cdots, d-1\}$. In all of these d paths in $G[B(d, D)]$, we enter the copy G_{z_i}, $0 \leqq i \leqq D$, via the same vertex, namely $(z_i; g(x_i))$. Copy $G_{z_{D+1}}$ is entered via vertex $(z_{D+1}; g(\alpha))$, which is different for each of the d paths. We leave the copy G_{z_i}, $1 \leqq i \leqq D + 1$, via different vertices, as the leaving arc depends on the d-arity of z_i, which changes with α. Of course, each path also leaves copy G_{z_0} by a different vertex. For a fixed α (and hence a fixed path $z_0, z_1, \cdots, z_{D+1}$) we denote by α_i the subscript j of the arc o_j leading from G_{z_i} to $G_{z_{i+1}}$. Then the time the information arrives (on this path) at the goal copy G_y is $t_\alpha = t_{j_0}^{\alpha_0} + t_{j_1}^{\alpha_1} + \cdots + t_{j_D}^{\alpha_D}$, where $j_i = g(x_i)$ for $i = 1, \cdots, D$.

Now consider different values of α. As α takes on all of the possible values $\{0, 1, \cdots, d-1\}$, the d-arity of z_i for each fixed $i > 0$ takes all the possible values

$\{0, 1, \cdots, d-1\}$. It follows that the α_i also takes on all of the possible values $\{0, 1, \cdots, d-1\}$. Therefore,

$$\sum_{\alpha=0}^{d-1} t_\alpha = \sum_{i=0}^{d-1} \sum_{k=0}^{D} t_{jk}^i = \sum_{k=0}^{D} \sum_{i=0}^{d-1} t_{jk}^i \leqq (D+1)d\bar{b}(G).$$

The minimum time at which G_y is informed is $\min_\alpha \{t_\alpha\} \leqq 1/d \sum_{\alpha=0}^{d-1} t_\alpha \leqq (D+1)\bar{b}(G)$. Since it requires at most $b(G)$ units of time to complete the broadcast within G_y, we have $b(G[B(d, D)]) \leqq (D+1)\bar{b}(G) + b(G)$. □

As noted in § 3, in the above scheme the message may arrive at some vertex more than once. By deleting redundant calls, a scheme can be obtained that completes the broadcast at the appropriate time.

7. Upper bounds. In this section, we give examples of the construction of specific bounded degree graphs and determine the broadcast time for these graphs. We have determined the broadcast time of many such graphs. The best result obtained for each degree Δ, $3 \leqq \Delta \leqq 16$, is given in Table 7.1.

Let us turn our attention to cubic graphs. As we saw in § 4, if we let $B = B(2, D)$ and $G = K_2$, the resulting compound graph $H = K_2[B(2, D)]$ is a 3-regular graph on $n = 2 \times 2^D = 2^{D+1}$ vertices and has broadcast time $b(H) \leqq 2D + 1 = 2 \log_2 n - 1$. (Note that Theorem 4 gives $b(H) \leqq 2D + 3$ since the broadcast scheme described in the proof of Theorem 4 is more general than the one developed for the specific case in § 4.) However, if we choose the 3-star as G and compound this graph with $B = B(3, D)$ we get $H' = G[B(3, D)]$, a graph on 4×3^D vertices. Since $\bar{b}(G) = 3$ (see Fig. 6.1(b)), we get $b(H') \leqq 3D + 6 = 3 \log_3 n - 3 \log_3 4 + 6 \approx 1.893 \log_2 n + 2.214$. The graph T_6 of Fig. 6.1(c) has $\bar{b}(T_6) = 15/4$. If we compound T_6 with $B = B(4, D)$, we get $H'' = T_6[B(4, D)]$, a graph on 6×4^D vertices with $b(H'') \leqq (15/4)D + 31/4 = (15/4) \log_4 n - (15/4) \log_4 6 + 31/4 \approx 1.875 \log_2 n + 2.903$. This is the best value we have obtained for cubic graphs.

As Δ gets larger, the number of possible constructions grows rapidly and the task of finding the best construction for a particular fixed Δ becomes difficult. Another approach is to start with a particular graph that is known to be good for broadcasting, such as a minimum broadcast graph or a sparse broadcast graph, and compound it with de Bruijn digraphs of various degrees to obtain compound graphs for various values of Δ.

TABLE 7.1
Best bounds.

Δ	G	$\bar{b}(G)$	d	Upper bound	Lower bound
3	T_6	15/4	4	1.875000	1.440420
4	C_8	34/8	8	1.416667	1.137467
5	40-sbg	266/40	40	1.249547	1.056215
6	C_6	50/12	12	1.162262	1.025404
7	14-mbg	150/28	28	1.114364	1.012034
8	56-sbg	828/112	112	1.086010	1.005842
9	62-mbg	926/124	124	1.073847	1.002874
10	126-mbg	2138/252	252	1.063536	1.001424
11	254-mbg	4822/508	508	1.056008	1.000709
12	126-mbg	3396/378	378	1.049273	1.000353
13	254-mbg	7614/762	762	1.043712	1.000176
14	510-mbg	16824/1530	1530	1.039394	1.000088
15	1022-mbg	36786/3066	3066	1.035909	1.000044
16	2046-mbg	79788/6138	6138	1.033017	1.000022

As an example, consider the Heawood graph (shown in Fig. 7.1(a)), a cubic graph that is a minimum broadcast graph on 14 vertices. Figure 7.1(b) describes a broadcast scheme for an arbitrary originator in the graph. (Only those edges on which calls are made are shown.) The label on each vertex indicates the time at which the vertex receives the message assuming that the first call is made at time 1. Note that if a vertex in this 14-vertex graph begins a broadcast at time 0, two vertices (specifically the originator and the first vertex it calls during the scheme) will have completed all of their calls at time 3, and the remaining twelve vertices will all be involved in a final call at time 4. The two former vertices will be "available" to call vertices external to G at time 4 and the twelve other vertices will be "available" at time 5.

Replace each vertex in the de Bruijn digraph $B = B(14, D)$ with a copy of G, the Heawood graph, such that each vertex of G has one in-arc and one out-arc. The resulting graph is $G[B(14, D)]$, which is regular of degree 5. In this particular graph, each vertex in each copy of G has one out-neighbour, so $\bar{b}(G) = 68/14$. Since $b(G) = 4$, from Theorem 4, $b(G[B(14, D)] \leqq (D + 1)(68/14) + 4 = (68/14)\log_{14} n + O(1) \approx 1.276 \log_2 n + O(1)$.

Similarly, we could use $B = B(28, D)$. In this case, replace each vertex in B with a copy of G, the Heawood graph, as before except that each vertex of G has two in-arcs and two out-arcs attached. The resulting graph, $G[B(28, D)]$, is a regular graph of degree 7. Since each vertex has two out-neighbours, we get $\bar{b}(G) = 150/28$. Since $b(G) = 4$, Theorem 4 gives $b(G[B(28, D)]) \leqq (D + 1)(150/28) + 4 = (150/28)\log_{28} n + O(1) \approx 1.114 \log_2 n + O(1)$. As indicated in Table 7.1, the Heawood graph produces the best-known bounded degree graph for $\Delta = 7$.

In fact, if we wish to produce an odd degree $\Delta \geqq 5$ compound graph, we can replace each vertex of the de Bruijn digraph $B(7(\Delta - 3), D)$ with a Heawood graph such that each vertex of each Heawood graph has $(\Delta - 3)/2$ in-arcs and $(\Delta - 3)/2$ out-arcs.

To produce a degree-4 compound graph, we can replace each vertex of the de Bruijn digraph $B(7, D)$ with a Heawood graph so that each vertex of each Heawood graph has either one in-arc or one out-arc. If the arcs are connected arbitrarily, we can be assured that each of the seven vertices with out-neighbours will be available by time 5, so that $\bar{b}(G) = 35/7 = 5$. However, if we use the fact that the Heawood graph is bipartite and the two vertices that are available at time 4 are adjacent, we can connect the arcs as shown in Fig. 7.2 and guarantee that one of the vertices with an out-neighbour is available at time 4, giving $\bar{b}(G) = 34/7$. Using this connection scheme, $b(G[B(7, D)]) \leqq (D + 1)34/7 + 4 = (34/7)\log_7 n + O(1) \approx 1.730 \log_2 n + O(1)$.

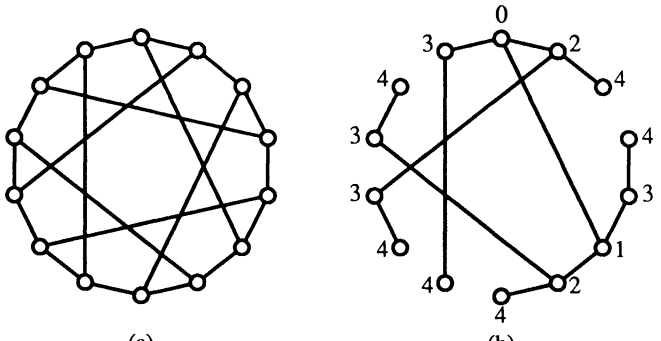


FIG. 7.1. *Heawood graph* (a) *and its broadcast scheme* (b).

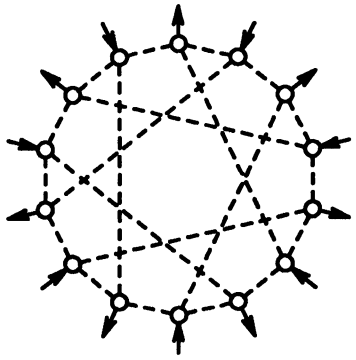


FIG. 7.2. *Connection scheme.*

To produce even degree $\Delta \geqq 6$ compound graphs, we can replace each vertex of the de Bruijn digraph $B(7(\Delta - 3), D)$ with a Heawood graph such that the seven vertices of one class of each Heawood graph have $\Delta/2 - 1$ in-arcs and $\Delta/2 - 2$ out-arcs, while the vertices in the other class have $\Delta/2 - 2$ in-arcs and $\Delta/2 - 1$ out-arcs.

Let us now consider the k-cubes, one of the two known infinite families of minimum broadcast graphs. The k-cube on $n = 2^k$ vertices is regular of degree k, and all vertices become available $k + 1$ time units after the originator receives the message.

A graph of degree Δ can be constructed by replacing each vertex of the de Bruijn digraph $B(m2^{\Delta-2m}, D)$ by G, the $(\Delta - 2m)$-cube, for any $1 \leqq m \leqq \lfloor(\Delta - 1)/2\rfloor$, and distributing the arcs from the de Bruijn digraph so that each vertex of each copy of G is given m in-arcs and m out-arcs. Each vertex of a cube can inform its m new neighbours at times $\Delta - 2m + 1, \Delta - 2m + 2, \cdots, \Delta - m$ after the message first enters the cube. This gives $\bar{b}(G) = [\sum_{i=1}^{m} (2^{\Delta-2m})(\Delta - 2m + i)]/[m2^{\Delta-2m}] = \Delta - (3m - 1)/2$. Applying Theorem 4, we obtain

$$\begin{aligned} b(G[B(m2^{\Delta-2m}, D)]) &\leqq (D+1)(\Delta-(3m-1)/2)+\Delta-2m \\ &= (\Delta-(3m-1)/2)\log_{m2^{\Delta-2m}} n + O(1) \\ &= [(\Delta-(3m-1)/2)/(\Delta-2m+\log_2 m)]\log_2 n + O(1). \end{aligned}$$

Thus,

$$(2) \qquad b((\Delta-2m)\text{-cube}\,[B(m2^{\Delta-2m}, D)]) \leqq \frac{\Delta - \dfrac{3m-1}{2}}{\Delta-2m+\log_2 m}\log_2 n + O(1).$$

The constant $(\Delta - (3m - 1)/2)/(\Delta - 2m + \log_2 m)$ from expression (2) can be simplified to the form $1 + c_m/(\Delta - d_m)$, where c_m and d_m are constants depending on m. The expression for the broadcasting time of the compound of a cube in a de Bruijn digraph therefore has the form $(1 + c_m/(\Delta - d_m)) \log_2 n + O(1)$. The smallest constant c_m that can be obtained from expression (2) is $c_3 = 2 - \log_2 3 \approx 0.415$, so the best asymptotic broadcasting time will result from the compound of a $(\Delta - 6)$-cube in a de Bruijn digraph $B(3 \times 2^{\Delta-6}, D)$.

THEOREM 5. $b(n, \Delta) \leqq (1 + (2 - \log_2 3)/(\Delta - 6 + \log_2 3)) \log_2 n + O(1) \approx (1 + 0.415/\Delta) \log_2 n$.

A graph of degree Δ can also be constructed by replacing each vertex of $B([(2m + 1)/2]2^{\Delta-2m-1}, D)$ by a $(\Delta - 2m - 1)$-cube for any $0 \leqq m \leqq \lfloor\Delta/2\rfloor - 1$.

However, the resulting asymptotic broadcasting time is not as good as the result of Theorem 5.

Another way to obtain a graph of degree Δ is to use the family of minimum broadcast graphs from [8]. These graphs are k-regular with $n = 2^{k+1} - 2$ vertices and broadcast time $k + 1$. A computation similar to the computation for cubes above shows that the best constant is obtained by compounding the $\Delta - 6$-regular graph from this family in $B(3 \times 2^{\Delta-6}, D)$. Asymptotically, this will not improve the constant of Theorem 5. However, for any fixed Δ, use of the $\Delta - 6$-regular graph gives a slightly smaller constant than the $(\Delta - 6)$-cube.

Similar calculations have been done using various graphs G including known minimum broadcast graphs and sparse broadcast graphs, cycles C_i for $i = 4, \cdots, 12$ and compounds of sparse broadcast graphs in cubes. (Note that the last case involves two compounding operations since the result of compounding in a cube is then compounded in a de Bruijn digraph.) The best values that we have obtained are shown in Table 7.1. The table shows the degree of the graph constructed (Δ), the graph G, the average time ($\bar{b}(G)$) needed to transmit a message originated in a copy of G to all of the out-neighbour copies, the indegree (outdegree) of the de Bruijn digraph used (d), the upper bound obtained by this graph, and the best lower bound known. Note that the upper bound is calculated by the simple formula $\bar{b}(G)/\log_2 d$. T_6 is used to denote the tree on six vertices shown in Fig. 6.1(c). C_6 and C_8 denote the cycles on six and eight vertices, respectively. Note that although C_6 is a minimum broadcast graph, C_8 is not. 14-mbg indicates the Heawood graph of Fig. 7.1(a), a minimum broadcast graph on 14 vertices. i-mbg indicates a minimum broadcast graph on i vertices and i-sbg is used to represent a sparse broadcast graph on i vertices (see [3] and [8]).

Using a generalization of the proof of Theorem 5, we can show that the asymptotic constant $1 + (2 - \log_2 3)/\Delta$ of Theorem 5 is the best possible asymptotic result for compounds in de Bruijn digraphs. Thus, the gap between the best known upper and lower bounds cannot be eliminated even by substituting arbitrarily large minimum broadcast graphs (which remain to be discovered) into de Bruijn digraphs.

Our technique of compounding graphs in de Bruijn digraphs does not give graphs for all values of n. Instead of de Bruijn digraphs, we could compound in Kautz digraphs [17] (see also [1] or [2]) or sequence graphs [9]. This will not give asymptotic improvements but will give different values of n for which the broadcasting times are the same as in Table 7.1. Graphs for other values of n can be obtained from those described in this paper using a technique from [20].

Acknowledgments. We thank the referees for their constructive suggestions. In particular, the remarks of one of the referees led to improvements in the proofs of Lemma 3 and Theorem 1.

REFERENCES

[1] D. AMETER AND M. GREE, *Graphs and Interconnection Networks*, to appear.

[2] J.-C. BERMOND, C. DELORME, AND J.-J. QUISQUATER, *Strategies for interconnection networks: some methods from graph theory*, J. Parallel and Dist. Comput., 3 (1986), pp. 433–449.

[3] J.-C. BERMOND, P. HELL, A. L. LIESTMAN, AND J. G. PETERS, *Sparse broadcast graphs*, Discrete Appl. Math., to appear.

[4] J.-C. BERMOND AND C. PEYRAT, *Broadcasting in de Bruijn networks*, in Proc. 19th SE Conference on Combinatorics, Graph Theory and Computing, Congr. Numer., 1988, pp. 283–292.

[5] S. C. CHAU AND A. L. LIESTMAN, *Constructing minimal broadcast networks*, J. Combin. Inform. System Sci., 10 (1985), pp. 110–122.

[6] F. R. K. CHUNG, *Diameters of graphs: old problems and new results*, in Proc. 18th SE Conference on Combinatorics, Graph Theory and Computing, Congr. Numer., 1987, pp. 295–317.
[7] N. G. DE BRUIJN, *A combinatorial problem*, Nederl. Akad. Wetensch. Indag. Math. Proc. Ser. A, 49 (1946), pp. 758–764.
[8] M. R. FELLOWS, *Algebraic constructions of efficient broadcast networks*, to appear.
[9] M. ESCUDERO, J. FABREGA, AND M. A. FIOL, *Routing and expansion of interconnection networks based on sequence graphs*, in Proc. 5th International Symposium on Applied Mathematics, Switzerland, 1987, pp. 47–52.
[10] A. FARLEY, *Minimal broadcast networks*, Networks, 9 (1979), pp. 313–332.
[11] A. FARLEY, S. HEDETNIEMI, S. MITCHELL, AND A. PROSKUROWSKI, *Minimum broadcast graphs*, Discrete Math., 25 (1979), pp. 189–193.
[12] L. GARGANO AND U. VACCARO, *On the construction of minimal broadcast networks*, Networks, 19 (1989), pp. 673–689.
[13] M. GRIGNI AND D. PELEG, *Tight bounds on minimum broadcast networks*, SIAM J. Discrete Math., 4(1991), pp. 207–222.
[14] S. T. HEDETNIEMI, S. M. HEDETNIEMI, AND A. L. LIESTMAN, *A survey of broadcasting and gossiping in communication networks*, Networks, 18 (1988), pp. 319–349.
[15] W. D. HILLIS, *The Connection Machine*, Massachusetts Institute of Technology Press, ACM Distinguished Dissertation, Cambridge, MA, 1985.
[16] M. R. JERRUM AND S. SKYUM, *Families of fixed degree graphs for processor interconnection*, IEEE Trans. Comput. C-33(2), 1984, pp. 190–194.
[17] W. H. KAUTZ, *Bounds on directed* (d, k) *graphs*, in Theory of Cellular Logic Networks and Machines, AFRCL68-0668 Final Report, 1968, pp. 20–28.
[18] D. E. KNUTH, *The Art of Computer Programming*, *Vol.* 3, *Sorting and Searching*, Sec. 5.4.2, Exer. 7, Addison-Wesley, Reading, MA, 1973.
[19] A. L. LIESTMAN, *Fault-tolerant broadcast graphs*, Networks, 15 (1985), pp. 159–171.
[20] A. L. LIESTMAN AND J. G. PETERS, *Broadcast networks of bounded degree*, SIAM J. Discrete Math., 1 (1988), pp. 531–540.
[21] E. P. MILES, JR., *Generalized Fibonacci numbers and matrices*, Amer. Math. Monthly, 67 (1960), pp. 745–757.
[22] S. MITCHELL AND S. HEDETNIEMI, *A census of minimum broadcast graphs*, J. Combin. Inform. System Sci., 5 (1980), pp. 141–151.
[23] J. G. PETERS, private communication, 1990.
[24] P. J. SLATER, E. COCKAYNE, AND S. T. HEDETNIEMI, *Information dissemination in trees*, SIAM J. Comput., 10 (1981), pp. 692–701.
[25] X. WANG, private communication, 1986.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 25–53, February 1992
© 1992 Society for Industrial and Applied Mathematics
003

AUGMENTING GRAPHS TO MEET EDGE-CONNECTIVITY REQUIREMENTS*

ANDRÁS FRANK†

Abstract. What is the minimum number γ of edges to be added to a given graph G so that in the resulting graph the edge-connectivity between every pair $\{u, v\}$ of its nodes is at least a prescribed value $r(u, v)$?

Generalizing earlier results of S. Sridhar and R. Chandrasekaran [*Integer Programming and Combinatorial Optimization*, R. Kannan and W. Pulleyblank, eds., Proceedings of a conference held at the University of Waterloo, University of Waterloo Press, Waterloo, Ontario, Canada, 1990, pp. 467–484] (when G is the empty graph), of K. P. Eswaran and R. E. Tarjan [*SIAM Journal on Computing*, 5 (1976), pp. 653–665] (when $r(u, v) \equiv 2$), and of G.-R. Cai and Y.-G. Sun [*Networks*, 19 (1989), pp. 151–172] (when $r(u, v) \equiv k \geqq 2$), we derive a min-max formula for γ and describe a polynomial time algorithm to compute γ. The directed counterpart of the problem is solved in the same sense for the case when $r(u, v) \equiv k \geqq 1$ and is shown to be NP-complete if $r(u, v) \equiv 1$ for $u, v \in T$, and $r(u, v) \equiv 0$ otherwise where T is a specified subset of nodes.

A fundamental tool in the proof is the splitting theorems of W. Mader [*Annals of Discrete Mathematics*, 3 (1978), pp. 145–164] and L. Lovász [lecture, Prague, 1974; North–Holland, Amsterdam, 1979]. We also rely extensively on techniques concerning submodular functions. The method makes it possible to solve a degree-constrained version of the problem. The minimum-cost augmentation problem can also be solved in polynomial time provided that the edge-costs arise from node-costs, while the problem for arbitrary edge-costs was known to be NP-complete even for $r(u, v) \equiv 2$.

Key words. network synthesis, augmenting edge-connectivity of graphs, polymatroids, network design, connectivity, combinatorial optimization

AMS(MOS) subject classifications. 05C, 90B, 68R

1. Introduction.

A typical problem in combinatorial optimization is to find a minimum number, or more generally, a minimum cost, of edges to be added to a graph so that the resulting graph satisfies some prescribed properties. In this paper we are concerned with edge-connectivity properties.

Main problem. What is the minimum number γ (respectively, a minimum cost) of edges to be added to a given directed or undirected graph G so that in the resulting graph the edge-connectivity $\lambda(u, v)$ between every pair $\{u, v\}$ of nodes is at least a prescribed value $r(u, v)$?

Here, the *edge-connectivity* $\lambda(u, v)$ of u and v means the maximum number of pairwise edge-disjoint (directed) paths from u to v. Note that $\lambda(u, v)$ can be interpreted as the maximum flow value between u and v if the capacities of the edges of G are defined to be 1.

To distinguish between the two versions of the main problem, we will sometimes refer to them as the "cardinality case" and "the min-cost case."

A capacitated version of the main problem is as follows.

Max-flow version. Suppose that $g(u, v)$ is a nonnegative capacity function on the pairs of nodes u, v $(u, v \in V)$, and let $r(u, v)$ be another nonnegative function on the pairs of nodes that serves as a maximum flow requirement. The problem is to increase the existing capacities so that in the resulting network the maximum flow value between u and v is at least $r(u, v)$ for each pair $\{u, v\}$ of nodes, such that the sum of capacity increments is minimum.

* Received by the editors March 26, 1990; accepted for publication (in revised form) December 14, 1990.

† Research Institute for Discrete Mathematics, Institute for Operations Research, University of Bonn, Nassestr. 2, Bonn–1, Germany D-5300. The author is on leave from the Department of Computer Science, Eötvös University, Múzeum krt. 6–8, Budapest, Hungary H-1088. This work was supported by Sonderforschungsbereich 303 (Deutsche Forschungsgemeinschaft).

Note that if $g(u, v)$ and $r(u, v)$ are integer-valued and the capacity increments are required to be integer-valued, then the main problem is equivalent to the flow version in the sense that a min-max result for one problem easily transforms to a min-max result for the other. Namely, the main problem can be formulated as a max-flow version by letting $g(u, v) = 1$ when (u, v) is an edge of G, and $g(u, v) = 0$, otherwise. Conversely, if g is integer-valued, we can define a graph having $g(u, v)$ parallel edges between each pair of nodes u and v; then a solution to the main problem on G yields a solution to the integer-valued max-flow problem.

This equivalence, however, does not mean algorithmic equivalence. We are going to develop strongly polynomial time algorithms for the more difficult max-flow augmentation problem.

(A polynomial time algorithm is called *strongly polynomial* if it uses, besides ordinary data manipulation, only the basic operations like comparing, adding, subtracting, multiplying, and dividing numbers, and if the number of these operations is independent of the numbers occurring in the input.)

Another useful observation is that in the flow version we may get better results if fractional increments are allowed. For example, let $V := \{a, b, c\}$, $g \equiv 0$, and $r \equiv 1$. If only integers are allowed for the increments, then the value of the best solution is 2: increase the capacity of edges ab and bc by 1. If we use fractional increments, then the value of the best solution is 1.5: increase the capacity of each edge by 0.5. On the other hand, we will see that, apart from some marginal cases, the integer-valued optimum is at most one half larger than the fractional optimum.

It is natural to consider node-connectivity augmentation problems as well. That is, given a prescribed value $r(u, v)$ for each pair of nodes u, v, what is the minimum number (respectively, a minimum cost) of edges to be added to a given directed or undirected graph G so that in the resulting graph there are $r(u, v)$ openly disjoint paths between every pair of nodes u, v? Two paths connecting u and v are called *openly disjoint* if they are node-disjoint, except for the end-nodes. No new contribution to this problem is given here and we mention it merely for the sake of completeness.

We briefly summarize some known special cases for which the above augmentation problems have been solved. Let us start with undirected graphs.

Gomory and Hu [19] algorithmically solved the fractional case of the max-flow version of the augmentation problem where $g \equiv 0$. See also [8]. The minimum cost version of the same problem was solved by Bland, Goldfarb, and Todd [1] via the ellipsoid method. Sridhar and Chandrasekaran [30] solved the main problem when the starting graph is the empty graph. (Actually, they solved the integer-valued max-flow version when $g \equiv 0$.) Frank and Chou [9] solved the same problem under the additional requirement that no parallel edges are allowed to be added. (Note that Frank and the present author are different.)

Suppose now that $r(u, v)$ is identically k. When $k = 1$, the min-cost case of the main problem transforms into a minimum cost spanning tree problem, that is nicely solvable. For $k = 2$, however, the min-cost problem turns out to be NP-complete, as the Hamiltonian circuit problem can easily be formulated in this form (see Eswaran and Tarjan [7]).

Eswaran and Tarjan described a polynomial time algorithm to find a minimum number of new edges the addition of which makes a graph 2-edge-connected. Generalizing this for arbitrary $k \geqq 2$, Watanabe and Nakamura [31] described a polynomial time algorithm to find a minimum number of new edges γ to be added to make a graph k-edge-connected. The same problem was also solved by Cai and Sun [2] who, in addition, provided a nice min-max formula for the minimum. Both solutions are rather compli-

cated. Recently, Naor, Gusfield, and Martel [28] developed an efficient algorithm for this edge-connectivity augmentation problem.

As for node-connectivity augmentation problems, we know much less. The problem was solved by Harary [21] when $r(u, v)$ is identically k and the starting graph is the empty graph. For arbitrary starting graphs the case $k = 2$ was settled by Eswaran and Tarjan [7] and the case $k = 3$ by Watanabe and Nakamura [32].

The following results concern directed graphs.

Suppose we are given a digraph G with a source s and a target t. Let $r(u, v) = k$ if $u = s$, $v = t$, and $r(u, v) = 0$ otherwise. In this case the main problem requires adding a minimum cost of edges so that in the resulting digraph there are k edge-disjoint paths from s to t. This problem can easily be reduced to a minimum cost flow problem in the union graph of the new and the original edges where the costs of the original edges are defined to be zero.

If we are interested in openly disjoint paths, rather than edge-disjoint, from a source-node s to a target-node t, then the problem can easily be reduced to the edge-disjoint case by using a simple node-duplicating device mentioned in [8, § I/11]. Unfortunately, in more general cases, the node-duplicating technique does not seem to help in reducing a node-connectivity augmentation problem to the corresponding edge-connectivity augmentation problem. This is the case in the following situation.

Improve a digraph by adding a minimum cost of new edges so as to have k edge-disjoint paths from a specified source-node to each other node. (That is, in the main problem $r(u, v) = k$ if $u = s$, and $r(u, v) = 0$ otherwise.) This problem can be reduced to a weighted matroid intersection problem where the first matroid is k times the circuit-matroid of the underlying undirected graph (that is, a subset of edges is independent if it is the union of k forests) while the second matroid is a partition matroid where a subset of edges is independent if it contains no more than k edges entering the same node. Since there are good algorithms for the matroid intersection problem [5], this problem is also solvable in polynomial time.

We consider the openly disjoint counterpart of the preceding problem; that is, we improve a digraph by adding a minimum cost of new edges so as to have k openly disjoint paths from a specified source-node to each other node. The problem was solved in [14] with the help of submodular flows (as we were unable to reduce the problem to the edge-disjoint case by using the node-duplicating technique).

Eswaran and Tarjan showed how to make a directed graph strongly connected by adding a minimum number of new edges. They also showed that the minimum cost version of the problem is NP-complete, as the directed Hamiltonian circuit problem can be formulated this way. Note, however, that the problem is solvable in strongly polynomial time if we are allowed to add a new edge (u, v) only if (v, u) is an original edge of the digraph (see Lucchesi and Younger [25] and Frank [10]).

The main problem for directed graphs when $r(u, v) \equiv k$ was solved by Fulkerson and Shapley [16] when the starting graph is $G = (V, \varnothing)$, and by Kajitani and Ueno [22] when the starting graph is a directed tree.

Finally, we mention a paper of Gusfield [20] in which a linear time algorithm is described to make a mixed graph strongly connected by adding a minimum number of new directed edges. (A *mixed graph* is one with possibly directed and undirected edges. It is called *strongly connected* if, for every pair of nodes u, v, there is a path from u to v that consists of directed edges in the right direction and arbitrary undirected edges.)

The main purposes of the present paper are as follows. For undirected graphs we completely solve the cardinality case of the main problem. Along the way we provide a short proof of the theorem of Cai and Sun. For directed graphs, we solve the cardinality

case if $r(i, j) \equiv k$. We also consider degree-constrained and minimum-cost augmentations for both directed and undirected graphs. The proofs give rise to algorithms that are strongly polynomial even in the max-flow version.

One basic tool in the proof is the so-called splitting technique. We are going to use three theorems. One is due to Lovász, while the other two are due to Mader. To make the paper as self-contained as possible, we prove Lovász's theorem [24] here, as well as one of the two Mader theorems [26], [27]. A relatively simple proof of Mader's other splitting-off theorem is given in a separate paper [12].

We consider whether there is a solution to the augmentation problem for directed graphs when $r(u, v)$ is arbitrary. Unlike the undirected case this problem turns out to be NP-complete even for the following simple demand function. Let T be a subset of nodes and s a specified node not in T. Define $r(s, v) \equiv 1$ if $v \in T$, and $r(s, v) \equiv 0$ otherwise. (In other words, the problem of finding a minimum number of edges to be added to G so that in the augmented digraph every element of T is reachable from s is NP-complete.)

For both the directed and the undirected case, the method makes it possible to solve a degree-constrained version when, in addition, upper and lower bounds are imposed at every node for the number of newly added edges incident to that node. We will show that in the above cases the minimum-cost augmentation problem can also be solved in polynomial time, provided that the edge-costs arise from node-costs. (As we mentioned above, the problem for arbitrary edge-costs is NP-complete even for $r(u, v) \equiv 2$.)

Another basic technique comes from the theory of submodular functions. (For a survey, see [13].) We describe, however, the main results and proofs so that they can be understood without any prior knowledge in this area. It will be clarified in a separate section how some basic features of submodular functions and polymatroids are in the background of our approach.

The structure of the paper is as follows. Section 2 comprises the necessary notations and notions from graph theory. Section 3 describes the results on directed graphs. Furthermore, a new proof of Mader's directed splitting-off theorem is given. In § 3 we provide a simple proof of a theorem of Cai and Sun, along with some degree-constrained versions. A new proof of Lovász's splitting-off theorem is also presented. Section 5 contains the general result for undirected graphs. Section 6 lists some notions and theorems from polymatroid theory, while §§ 7 and 8 explain the relationship between polymatroids and augmentation problems. In § 9 we consider the fractional augmentation problem and algorithmic aspects.

2. Notation and basic concepts. Typically, we work with a finite ground set V. We will not distinguish between a one-element set $\{x\}$ and its element x. The union of a set X and an element y is denoted by $X + y$. For $s, t \in V$ a subset X of V is called an *$t\bar{s}$-set* is $t \in X$ and $s \notin X$. For two sets X, Y, $X - Y$ denotes the set of elements in X, but not in Y. $X \subset Y$ denotes that X is a subset of Y and $X \neq Y$. Two subsets X, Y of V are called *intersecting* if none of $X \cap Y$, $X - Y$, $Y - X$ is empty. If, in addition, $V - (X \cup Y)$ is nonempty, then X, Y are called *crossing*. A family of subsets is called *laminar* if it includes no intersecting sets.

By a *partition* $\{X_1, X_2, \cdots, X_t\}$ of a set X, we mean a family of disjoint subsets of X whose union is X. By a *subpartition* of V, we mean a partition of a subset X of V.

Let $\mathcal{F}$ be the family of subsets of V possessing a certain property p. We say that a member $X \in \mathcal{F}$ is *maximal* (with respect to p) if no member of $\mathcal{F}$ includes X as a proper subset. For example, a maximal critical set means a critical set not included in any other critical set (whatever critical means).

Throughout, we use the term "graph" for an undirected graph and "digraph" for a directed graph. Let $G = (V, E)$ be a graph with node set V and edge set E. We denote an edge e connecting nodes u and v by uv or vu. This is not quite precise since there may be parallel edges between u and v. This ambiguity, however, will not cause any trouble. Both parallel edges and loops are allowed.

For a digraph $G = (V, E)$ a directed edge $e = uv$ is meant to be an edge from u to v. In this case, vu means the oppositely directed edge.

For a graph or digraph $G = (V, E)$, $E(X)$ denotes the set of edges with both end-nodes in X and is called the set of edges *induced* by X. For $X, Y \subseteq V$, $d(X, Y)$ denotes the number of edges between $X - Y$ and $Y - X$ (in any direction) and $\bar{d}(X, Y) := d(V - X, Y)$. We denote $d(X) := d(X, V - X)$. If $F \subseteq E$ and G is undirected, $d_F(X)$ stands for the number of edges in F entering X. The number of edges incident to a node v is called the *degree* of v. The contribution of a loop vv to the degree of v is, by definition, two.

Deleting an edge e means that we leave out e from E while the node set V is unchanged. For the resulting graph, we use the notation $G - e$. *Deleting* a subset C of nodes means that we leave out the elements of C and all the edges incident to some elements of C. The resulting graph is denoted by $G - C$. *Splitting off* a pair uv, vz of edges with $u \neq z$ means that we replace the two edges uv, vz by a new edge $e = uz$. Note that if $u = z$, e is a loop.

In a digraph $G = (V, E)$ the *in-degree* $\rho(X)$ (*out-degree* $\delta(X)$) is the number of edges entering (leaving) X. If $F \subseteq E$ $\rho_F(X)$ stands for the number of edges from F entering X. The contribution of a directed loop vv to the in-degree of v and to the out-degree of v is, by definition, one.

We call an edge e (node v) of a graph $G = (V, E)$ a *cut edge* (*cut node*) if $G - e$ ($G - v$) has more components than G.

A graph is called *k-edge-connected* if $d(X) \geqq k$ for every $\emptyset \subset X \subset V$. A digraph is called *k-edge-connected* if $\rho(X) \geqq k$ for every $\emptyset \subset X \subset V$.

One fundamental result from graph theory is as follows.

MENGER'S THEOREM 2.1 (Edge-version in [8]). *In a directed* (*respectively, undirected*) *graph* $G = (V, E)$ *there are* k *edge-disjoint paths from* s *to* t *if and only if* $\rho(X) \geqq k$ (*respectively*, $d(X) \geqq k$) *for every* ts*-set* $X \subset V$.

In a graph or digraph $\lambda(u, v)$ denotes the maximum number of edge-disjoint paths from u to v. $\lambda(u, v)$ is called the *edge-connectivity* from u to v. (In undirected graphs, obviously $\lambda(u, v) = \lambda(v, u)$.) The following identities are often used throughout the paper.

PROPOSITION 2.2. *Let* G *be an arbitrary graph* $G = (V, E)$ *and* $X, Y \subseteq V$. *Then*

$$d(X) + d(Y) = d(X \cap Y) + d(X \cup Y) + 2d(X, Y), \tag{2.1}$$

$$d(X) + d(Y) = d(X - Y) + d(Y - X) + 2\bar{d}(X, Y). \tag{2.2}$$

PROPOSITION 2.3. *Let* G *be an arbitrary digraph* $G = (V, E)$ *and* $X, Y \subseteq V$. *Then*

$$\delta(X) + \delta(Y) = \delta(X \cap Y) + \delta(X \cup Y) + d(X, Y), \tag{2.3}$$

$$\rho(X) + \rho(Y) = \rho(X \cap Y) + \rho(X \cup Y) + d(X, Y). \tag{2.4}$$

If, in addition, $\delta(X \cap Y) = \rho(X \cap Y)$, *then*

$$\delta(X) + \delta(Y) = \delta(X - Y) + \delta(Y - X) + \bar{d}(X, Y), \tag{2.5}$$

$$\rho(X) + \rho(Y) = \rho(X - Y) + \rho(Y - X) + \bar{d}(X, Y). \tag{2.6}$$

Each formula can easily be proved by taking into consideration the contribution of each type of edges to the two sides.

Let S be a finite ground-set and $b : 2^S \to R \cup \{\infty\}$ a set function. We call b *fully submodular* or, briefly, *submodular* if $b(X) + b(Y) \geqq b(X \cap Y) + b(X \cup Y)$ holds for every $X, Y \subseteq S$. If the inequality is required only for intersecting sets X, Y, then b is called *intersecting submodular*.

A set function p is called *supermodular* if $-p$ is submodular. A finite-valued set function m is called *modular* if $m(X) + m(Y) = m(X \cap Y) + m(X \cup Y)$ holds for every $X, Y \subseteq S$. A modular function m with $m(\emptyset) = 0$ is determined by its value on the singletons, namely $m(X) = \Sigma(m(s) : s \in X)$.

Suppose that $m : V \to R \cup \{\infty\}$ is a function and $X \subseteq V$. Generally, we will use the notation $m(X) := \Sigma(m(s) : s \in X)$ with the following exceptions: $d(X)$ (see above) is *not* $\Sigma(d(v) : v \in X)$, and the case is the same with ρ and δ.

3. Directed graphs. Fulkerson and Shapley [16] described a method to construct a k-edge-connected digraph on n nodes ($k \leqq n$) with a minimum number of edges. Kajitani and Ueno [22] solved the problem of optimally augmenting a directed tree in order to get a k-edge-connected digraph. Here we solve this problem for arbitrary directed graphs.

THEOREM 3.1. *Given a directed graph $G = (V, E)$ and a positive integer k, G can be made k-edge-connected by adding at most γ new edges if and only if*

$$\Sigma(k - \rho(X_i)) \leqq \gamma \quad \text{and} \tag{3.1}$$

$$\Sigma(k - \delta(X_i)) \leqq \gamma \tag{3.2}$$

hold for every subpartition $\{X_1, X_2, \cdots, X_t\}$ of V.

Proof. Necessity. Suppose $G' = (V, E \cup F)$ is a k-edge-connected supergraph of G, where F denotes the set of new edges. Then every subset X_i of V has at least $k - \rho(X_i)$ new entering edges. Therefore, the number of new edges in G' is at least $\Sigma(k - \rho(X_i))$ and (3.1) follows. (3.2) is analogous. □

Let s be a node not in V, and $V' := V + s$. Let $G' = (V', E')$ be a digraph with in- and out-degree functions ρ' and δ', respectively.

PROPOSITION 3.2. *Suppose for $A, B \subseteq V'$ that $\rho'(A) = \rho'(B) = k \leqq \min(\rho'(A \cap B), \rho'(A \cup B))$. Then $\rho'(A \cap B) = \rho'(A \cup B) = k$ and $d'(A, B) = 0$.*

Proof. Applying (2.4) we obtain $k + k = \rho'(A) + \rho'(B) = \rho'(A \cap B) + \rho'(A \cup B) + d'(A, B) \geqq k + k + d'(A, B)$, from which $k = \rho'(A \cap B) = \rho'(A \cup B)$ and $d'(A, B) = 0$ follows. □

We prove the sufficiency in two steps.

LEMMA 3.3. *G can be extended to a digraph $G' = (V + s, E')$ by adding a new node s, γ new edges entering s, and γ new edges leaving s in such a way that for every subset $\emptyset \neq X \subset V$*

$$\rho'(X) \geqq k \quad \text{and} \tag{3.3a}$$

$$\delta'(X) \geqq k \tag{3.3b}$$

hold where ρ' and δ' denote the in-degree and out-degree function of G', respectively.

Note that by Menger's theorem (3.3) is equivalent to saying that the edge-connectivity in G' between every pair of original nodes is at least k.

Proof. We are going to prove that it is possible to add γ edges leaving s so that (3.3a) is satisfied. This implies (by reorienting every edge) that it is possible to add γ edges entering s so that (3.3b) is satisfied.

First, we add a sufficiently large number of edges leaving s so as to satisfy (3.3a). (It certainly will do if we add k edges from s to v for every $v \in V$.)

Second, discard new edges, one by one, as long as possible without violating (3.3a). Let G' denote the final extended digraph. The following claim implies the lemma.

CLAIM. $\delta'(s) \leqq \gamma$.

Proof. Call a subset $\varnothing \subset X \subset V$ *in-critical* if $\rho'(X) = k$. Let $S := \{v \in V, sv$ is an edge in $G'\}$. An edge sv cannot be left out from G' without violating (3.3a) precisely if sv enters an in-critical set. Therefore, by the minimality of G', there is a family $\mathcal{F} = \{X_1, X_2, \cdots, X_t\}$ of in-critical subsets of V covering S, and we can assume that t is minimal.

Case 1. $\mathcal{F}$ consists of disjoint sets. Then we have $kt = \Sigma(\rho'(X_i): i = 1, \cdots, t) = \delta'(s) + \Sigma(\rho(X_i): i = 1, \cdots, t)$ and, hence, by (3.1), $\delta'(s) = \Sigma(k - \rho(X_i): i = 1, \cdots, t) \leqq \gamma$.

Case 2. There are two intersecting members A, B of $\mathcal{F}$. If $A \cup B \neq V$, then $A \cup B$ is in-critical by Proposition 3.2, and then replacing A and B in $\mathcal{F}$ by $A \cup B$ we are in contradiction with the minimal choice of t. Therefore, $A \cup B = V$.

Let $Y_1 := V - A$ and $Y_2 := V - B$. Then $\delta(Y_1) = \rho(A)$, and $\delta(Y_2) = \rho(B)$. By (3.2) we have $\gamma \geqq k - \delta(Y_1) + k - \delta(Y_2) = k - \rho(A) + k - \rho(B) \geqq k - \rho'(A) + k - \rho'(B) + \delta'(s) = \delta'(s)$.

Therefore, the proof of Lemma 3.3 is complete. □

The theorem immediately follows by γ repeated applications of the following theorem of Mader [27].

THEOREM 3.4 ([27]). *Suppose that for a node s of a digraph $G' = (V + s, E')$ $\delta'(s) = \rho'(s)$ and the edge-connectivity between any two nodes distinct from s is at least k (that is,* (3.3) *holds). Then for any edge st there is an edge vs such that vs and st can be split off without violating* (3.3).

Call a pair of edges vs, st *splittable* if they can be split off without violating (3.3). Here we provide a (new) proof of Mader's theorem that will be useful in § 9 to improve the complexity of an algorithm arising from the naive implementation of Mader's theorem.

Proof. We need the following proposition.

PROPOSITION 3.5. *Suppose that $\delta'(s) = \rho'(s)$ for a node s of a digraph G' and the edge-connectivity between any two nodes distinct from s is at least k. If X, Y are intersecting subsets of nodes for which $\{s\} = X \cap Y$ and $\delta'(X) = \delta'(Y) = k$, then $\delta'(X - Y) = \delta'(Y - X) = k$, and $\bar{d}'(X, Y) = 0$.*

Proof. Applying (2.5) we obtain $k + k = \delta'(X) + \delta'(Y) = \delta'(X - Y) + \delta'(Y - X) + \bar{d}'(X, Y) \geqq k + k + \bar{d}'(X, Y)$ from which $\delta'(X - Y) = \delta'(Y - X) = k$ and $\bar{d}'(X, Y) = 0$ follows. □

Call a subset $\varnothing \subset X \subset V$ *in-critical* if $\rho'(X) = k$ and *out-critical* if $\delta'(X) = k$. X is called *critical* if it is either out- or in-critical. (Note that V is never critical.)

PROPOSITION 3.6. *Let A and B be two intersecting critical sets. Then either* (i) *$A \cup B$ is critical or* (ii) *$B - A$ is critical and $\bar{d}'(A, B) = 0$.*

Proof. If both A and B are in-critical and $A \cup B \subset V$, then Proposition 3.2 implies alternative (i). If $A \cup B = V$, then Proposition 3.5, when applied to $X := V + s - A$, $Y := V + s - B$, implies (ii). The situation is analogous if both A and B are out-critical. Finally, let A be in-critical and B out-critical. Proposition 3.2, when applied to A and $V + s - B$, implies (ii).

A pair $\{vs, st\}$ of edges is not splittable precisely if there is a critical set containing both v and t. Therefore, if there is no critical set containing t, any pair $\{vs, st\}$ is splittable.

For two intersecting critical sets A, B containing t, only alternative (i) may hold in Proposition 3.6 since the existence of edge st implies $\bar{d}'(A, B) > 0$. Therefore, the union M of all critical sets containing t is critical again.

We claim that there is an edge vs with $v \in V - M$. We indirectly suppose that no such an edge exists. If M is in-critical, then $\delta'(V-M) < \rho'(M) = k$, contradicting (3.3b). If M is out-critical, then $\delta'(s) = \rho'(s)$ implies that $\rho'(V-M) = \delta'(M+s) < \delta'(M) = k$, contradicting (3.3a).

By the choice of M, no critical set contains both v and t; therefore, vs and st are splittable. □

Our next problem is to find an augmentation of minimum cardinality if upper and lower bounds are imposed both on the in-degrees and the out-degrees of the digraph of newly added edges. Let $f_{\text{in}} \leqq g_{\text{in}}$ and $f_{\text{out}} \leqq g_{\text{out}}$ be four nonnegative integer-valued functions on V (infinite values are allowed for g_{in} and g_{out}). (Recall that $g_{\text{in}}(X)$ denotes $\Sigma(g_{\text{in}}(v) : v \in X)$.)

THEOREM 3.7. *Given a directed graph $G = (V, E)$ and a positive integer k, G can be made k-edge-connected by adding a set F of precisely γ new edges so that*

$$f_{\text{in}}(v) \leqq \rho_F(v) \leqq g_{\text{in}}(v) \quad \textit{and} \tag{3.4a}$$

$$f_{\text{out}}(v) \leqq \delta_F(v) \leqq g_{\text{out}}(v) \tag{3.4b}$$

hold for every node v of G if and only if

$$k - \rho(X) \leqq g_{\text{in}}(X) \quad \textit{and} \tag{3.5a}$$

$$k - \delta(X) \leqq g_{\text{out}}(X) \tag{3.5b}$$

hold for every subset $\varnothing \subset X \subset V$, and

$$\Sigma(k - \rho(X_i) : i = 1, \cdots, t) + f_{\text{in}}(X_o) \leqq \gamma, \tag{3.6a}$$

$$\Sigma(k - \delta(X_i) : i = 1, \cdots, t) + f_{\text{out}}(X_o) \leqq \gamma \tag{3.6b}$$

hold for every partition $\{X_o, X_1, X_2, \cdots, X_t\}$ of V where X_o may be empty.

(At this point we emphasize that both in this theorem and later theorems, degree-constrained augmentations loops are allowed to be added to G. It may be interesting to consider the problem when loops are not allowed.)

Proof of the necessity. Suppose that there is a required set F of new edges. Then $k \leqq \rho(X) + \rho_F(X) \leqq \rho(X) + \Sigma(\rho_F(v) : v \in X) \leqq \rho(X) + g_{\text{in}}(X)$, and (3.5a) follows. The proof of (3.5b) is analogous. Similarly, $\rho_F(X_i) \geqq k - \rho(X_i)$ and $\rho_F(v) \geqq f_{\text{in}}(v)$ and hence $|F| \geqq \Sigma(\rho_F(X_i) : i = 0, 1, \cdots, t) \geqq \Sigma(k - \rho(X_i) : i = 1, \cdots, t) + f_{\text{in}}(X_o)$, and (3.6a) follows. (3.6b) is analogous again. □

To prove the sufficiency, we can apply the method of the proof of Theorem 3.1. Here, we only outline this, and a formal proof is postponed to § 8, where the use of polymatroids make clear why such a proof works. The sketch below also indicates an algorithm to find a desired augmentation.

Sketch of the proof of sufficiency. We can assume that g_{in} and g_{out} is finite, since if $g_{\text{in}}(v)$, say, is infinite, then $g_{\text{in}}(v)$ can be revised to be $\max(k, \gamma, f_{\text{in}}(v))$. This modification does not destroy the necessary conditions.

Extend G by a new node s. For each node $v \in V$, add $g_{\text{in}}(v)$ parallel edges from s to v, and $g_{\text{out}}(v)$ parallel edges from v to s. (3.5) ensures that (3.3) holds for the extended digraph G'. Since $\gamma \leqq \min(g_{\text{in}}(V), g_{\text{out}}(V))$, we have $\rho'(s) \geqq \gamma$ and $\delta'(s) \geqq \gamma$. Now delete new edges, one by one, so that (3.3) continues to hold and each node v has at least $f_{\text{in}}(v)$ newly entering and $f_{\text{out}}(v)$ newly leaving edges. This deletion procedure stops when the current in-degree and out-degree of s is γ. If we can reach such a situation, then Mader's splitting-off theorem can be applied, and we are done.

The only trouble may arise if $\delta'(s) > \gamma$ and no new edge leaving s can be left out, or if $\rho'(s) > \gamma$ and no new edge entering s can be left out. Suppose that the first case

occurs (the second is analogous). Now a new edge sv may not be left out because sv either enters an in-critical set or $\rho'(v) = f_{\text{in}}(v)$. That is, the set $\{v: \rho'(v) > f_{\text{in}}(v)\}$ can be covered by a family $\mathcal{F} = \{X_1, \cdots, X_t\}$ of in-critical sets. Suppose that t is as small as possible. If $\mathcal{F}$ consists of disjoint sets, then $\{X_o, X_1, \cdots, X_t\}$ violates (3.6a), where $X_o := V - U(X_i: i = 1, \cdots, t)$. If $\mathcal{F}$ includes two intersecting sets A and B, then $A \cup B = V$ and the partition $\{Y_0, Y_1, Y_2\}$ of V, where $Y_o := A \cap B$, $Y_1 := V - A$, and $Y_2 := V - B$, violates (3.6b).

We are interested in degree-constrained augmentation when there is no requirement for the number of new edges; see the following theorem.

THEOREM 3.8. *Given a directed graph $G = (V, E)$ and a positive integer k, G can be made k-edge-connected by adding a set F of new edges satisfying* (3.4) *if and only if* (3.5) *holds, and*

$$\Sigma(k - \rho(X_i): i = 1, \cdots, t) + f_{\text{in}}(X_o) \leqq \alpha, \tag{3.7a}$$

$$\Sigma(k - \delta(X_i): i = 1, \cdots, t) + f_{\text{out}}(X_o) \leqq \alpha \tag{3.7b}$$

hold for every partition $\{X_o, X_1, X_2, \cdots, X_t\}$ of V where X_o may be empty and $\alpha :=$ $\min(g_{\text{out}}(V), g_{\text{in}}(V))$.

Proof. (3.5) is clearly necessary. To see the necessity of (3.7), let F be a set of new edges satisfying the requirements. Then $\Sigma(k - \rho(X_i): i = 1, \cdots, t) + f_{\text{in}}(X_o) \leqq |F| \leqq \alpha$, and (3.7a) follows. (3.7b) is analogous.

To see the sufficiency, observe that by choosing $\gamma := \alpha$ if α is finite and $\gamma := k|V| + f_{\text{in}}(V) + f_{\text{out}}(V))$ if $\alpha = \infty$, condition (3.6) follows from (3.7), and then Theorem 3.7 applies. □

Let us consider the minimum cost k-edge-connected augmentations. As we mentioned in the Introduction, if costs are assigned to the edges, the problem is NP-complete even if $k = 1$. Suppose now that $c_{\text{in}} : V \to R_+$ and $c_{\text{out}} : V \to R_+$ are two nonnegative cost functions on the node-set V of G. Our object is to find a k-edge-connected augmentation of G for which $\Sigma\rho_F(v)c_{\text{in}}(v) + \Sigma\delta_F(v)c_{\text{out}}(v)$ is minimum, where F is the newly added edges. The algorithm is a version of the proof of Lemma 3.3, in which the selection of new edges to be discarded is governed by the cost of the end-nodes in a greedy fashion.

ALGORITHM TO FIND A MINIMUM NODE-COST k-EDGE-CONNECTED AUGMENTATION OF A DIGRAPH

Add a new node s to V.

PART 1. Add k new parallel edges from s to v for every $v \in V$. (For the resulting digraph G', (3.3a) holds.) Assume that the new edges $f_1, f_2, \cdots$ are ordered according to the decreasing order of the c_{in} costs of their end-node u_i. (The order of parallel edges from s to u_i does not matter.) Go through the new edges in the given order and discard an f_i if this can be done without destroying (3.3a). Let γ_1 be the number of remaining new edges.

PART 2. Add k new parallel edges from v to s for every $v \in V$. (For the resulting digraph G', (3.3b) holds.) Assume that the new edges $f_1, f_2, \cdots$ are ordered according to the decreasing order of the c_{out} costs of their tail-node u_i. (The order of parallel edges from u_i to s does not matter.) Go through the new edges in the given order and discard an f_i if this can be done without destroying (3.3b). Let γ_2 be the number of remaining new edges.

Let $\gamma := \max(\gamma_1, \gamma_2)$. If $\gamma_2 < \gamma_1$, add $\gamma_1 - \gamma_2$ parallel edges from u to s, where u is a node and $c_{\text{out}}(u)$ is minimum. If $\gamma_1 < \gamma_2$, add $\gamma_2 - \gamma_1$ parallel edges from s to u, where u is a node and $c_{\text{in}}(u)$ is minimum.

PART 3. Let G' denote the final digraph. In G', $\delta'(s) = \rho'(s) = \gamma$, and (3.3) holds. Apply γ times Mader's Theorem 3.4 to G'. Let $G_1 = (V, E \cup F)$ denote the resulting digraph.

THEOREM 3.9. *The graph G_1, constructed above, is a minimum cost k-edge-connected augmentation of G.*

The proof of this theorem is postponed until § 7, where the necessary tools from polymatroid theory are already available. In § 8 we will make comments on algorithmic aspects of the procedure above including its extension to the capacitated case.

Perhaps it is worth mentioning that, by the above algorithm, a minimum cost augmentation is automatically a minimum cardinality augmentation.

We close this section by pointing out that the following two versions of the augmentation problem answered by Theorem 3.1 are NP-complete.

Problem A. Let $G = (V, E)$ be a directed graph, s a specified node of G, $T \subset V$ a specified subset of nodes, and γ a positive integer. Decide if it is possible to add at most γ new edges to G so as to have a path from s to every element of T.

Problem B. Let $G' = (V, E')$ be a directed graph, $R \subset V$ a specified subset of nodes, and γ a positive integer. Decide if it is possible to add at most γ new edges so as to have a path from every node of R to any other node of R.

THEOREM 3.10. *Both problems* A *and* B *are* NP-*complete.*

Proof. The following set covering problem is known to be NP-complete [17]: Given k sets $X_1, \cdots, X_k$ and an integer γ, decide if there is a set X with cardinality at most γ that intersects all X_i's.

First we show that set covering can be solved in polynomial time if Problem A can be solved in polynomial time. Let $S := X_1 \cup \cdots \cup X_k$. For each X_i let t_i be a new element and $T := \{t_1, \cdots, t_k\}$. Let s be an element not in $S \cup T$, and $V := S \cup T \cup \{s\}$. Let $G = (V, E)$ be a directed graph, where $E := \{vt_i : \text{if } v \in X_i\}$.

It is easily seen that if Problem A has a solution, it has a solution in which every new edge is of the form sv where $v \in S$. Then there is a solution to set covering; namely, the heads of new edges from a subset X of at most γ elements intersecting all X_i's. Conversely, if X is a solution to set covering, then $\{sv : v \in X\}$ as the set of new edges forms a solution to Problem A. Therefore, Problem A is NP-complete.

To see that Problem B is NP-complete, suppose that it is solvable in polynomial time. We then show that Problem A is also solvable in polynomial time. Indeed, a set F of new edges is a solution to Problem A with input $\{G = (V, E), s, T, \gamma\}$ if and only if F is a solution to Problem B with input $\{G' = (V, E'), R, \gamma\}$, where $E' := E \cup \{vs : v \in T\}$ and $R := T + s$. □

4. Undirected graphs. In this section we first provide a simpler proof of a theorem of Cai and Sun [2]. One advantage of this proof is that it can be extended to the degree-constrained case. Another one is that we use Lovász's splitting-off theorem [23], [24] rather than Mader's, which is much more difficult. This way, the proof becomes self-contained as we provide a (new) proof of Lovász's theorem.

THEOREM 4.1 ([2]). *Given an undirected graph $G = (V, E)$ and an integer $k \geqq 2$, G can be made k-edge-connected by adding at most γ new edges if and only if*

$$\Sigma(k - d(X_i)) \leqq 2\gamma \tag{4.1}$$

holds for every subpartition $\{X_1, X_2, \cdots, X_t\}$ of V.

Proof. The proof is analogous to that of Theorem 3.1.

Necessity. Suppose $G' = (V, E \cup F)$ is a k-edge-connected supergraph of G, where F denotes the set of new edges. Then every subset X_i of V has at least $k - d(X_i)$ newly

entering edges. Therefore, the number of new edges in G' is at least $\Sigma(k - d(X_i))/2$, and (4.1) follows.

We prove the sufficiency in two steps. Let s be a node not in V, and $V' := V + s$.

LEMMA 4.2. *G can be extended to a graph $G' = (V + s, E')$ by adding a new node s, and 2γ new edges between V and s in such a way that for every subset $\emptyset \neq X \subset V$*

$$d'(X) \geqq k \tag{4.2}$$

holds where d' denotes the degree function of G'.

Proof. First, we add a sufficiently large number of edges leaving s so as to satisfy (4.2). (It certainly will do if we add k edges from s to v for every $v \in V$.)

Second, we discard new edges, one by one, as long as possible without violating (4.2). Let G' denote the final extended graph. The following claim implies the lemma.

CLAIM. $d'(s) \leqq 2\gamma$.

Proof. Call a subset $\emptyset \subset X \subset V$ *critical* if $d'(X) = k$.

PROPOSITION 4.3. *If X and Y are intersecting critical, then both $X - Y$ and $Y - X$ are critical, and $\bar{d}'(X, Y) = 0$.*

Proof. We have $k + k = d'(X) + d'(Y) = d'(X - Y) + d'(Y - X) + 2\bar{d}'(X, Y) \geqq k + k$ from which the proposition follows. □

Let $S := \{u \in V: su \in E'\}$. An edge su cannot be left out without violating (4.2), precisely if there is a critical set containing u. Let M_u denote a minimal critical set containing u ($u \in S$) and let $\mathcal{F} := \{M_u: u \in S\}$. Let $X_1, X_2, \cdots, X_t$ be the maximal members of $\mathcal{F}$.

PROPOSITION 4.4. *Sets X_i ($i = 1, \cdots, t$) are pairwise disjoint.*

Proof. We prove that $\mathcal{F}$ is laminar. If $M_u, M_v \in \mathcal{F}$ are intersecting, then, by Proposition 4.3, $M_v - M_u$ is critical and $\bar{d}'(M_u, M_v) = 0$, therefore $v \in M_v - M_u$ contradicting the minimal choice of M_v.

By (4.2) we have $d'(s) = \Sigma(d'(X_i) - d(X_i): i = 1, \cdots, t)) = \Sigma(k - d(X_i): i = 1, \cdots, t) \leqq 2\gamma$, as required for the claim.

Add one extra edge sv if $d'(s)$ is odd to make it even, and let $\gamma' := d'(s)/2$ ($\leqq \gamma$). Theorem 4.1 follows by γ' repeated applications of the following theorem of Lovász. □

THEOREM 4.5 ([23], [24]). *Suppose that in a graph $G' = (V + s, E')$ $d'(s) > 0$ is even, and for every subset $\emptyset \neq X \subset V$ (4.2) holds. Then for every edge st there is an edge su so that the pair $\{st, su\}$ can be split off without violating (4.2).*

Remark. Lovász announced this theorem in Prague [23] and gave a proof in his problem book [24]. There Lovász broke up the problem into two parts. Problem 6.51 is the above statement (with different notation) formulated for Eulerian graphs while Problem 6.53 in Lovász's book sounds as follows, "*Prove that, provided $k \geqq 2$, the assertion of* 6.51 *holds for non-Eulerian graphs as well.*" However, this formulation is not completely precise since the evenness of the degree of s cannot be dropped, as is shown by the complete graph on four nodes. The proof (which is otherwise correct) given by Lovász [24, p. 287] uses a "tripartite" submodular inequality. Here we provide another proof that avoids this and will also be useful in § 9 to improve the efficiency of an algorithm arising from the naive implementation of Theorem 4.5.

Proof. Call a set $\emptyset \subset X \subset V$ *dangerous* if

$$d'(X) \leqq k + 1. \tag{4.3}$$

A pair $\{st, su\}$ of edges is called *splittable* if they can be split off without violating (4.2). This is the case precisely if there is no dangerous set X with $t, u \in X$. Let $S := \{v \in V: sv \in E'\}$.

CLAIM A. *Let A and B be intersecting dangerous sets with $t \in A \cap B$. Then*
(i) $\bar{d}'(A, B) = 1$ *and*
(ii) $S \not\subseteq A \cup B$ (*in particular*, $A \cup B \neq V$).

Proof. By (2.2) we have $(k+1) + (k+1) \geqq d'(A) + d'(B) = d'(A - B) + d'(B - A) + 2\bar{d}'(A, B) \geqq k + k + 2$ from which (i) follows.

Suppose that, indirectly, $S \subseteq A \cup B$. Let $\alpha := d'(s, A - B)$ and $\beta := d'(s, B - A)$. By symmetry, we can assume that $\alpha \geqq \beta$. Since $\bar{d}'(A, B) = 1$ we have $k \leqq d'(V - A) = d'(A + s) = d'(A) - \alpha + \beta - 1 \leqq d'(A) - 1 \leqq k$ from which $\alpha \leqq \beta$, and thus $\alpha = \beta$ follows. But this is impossible since, if $S \subseteq X \cup Y$, then $d'(s) = \alpha + \beta + 1 = 2\alpha + 1$, an odd number. □

CLAIM B. *If A and B are intersecting dangerous sets with $t \in A \cap B$, and A is maximal dangerous, then $d'(A) = d'(B) = k + 1$ and $d'(A \cap B) = k$.*

Proof. By Claim A, $A \cup B \neq V$, and by the maximality of A, $d'(A \cup B) \geqq k + 2$. From (2.1) we have $(k+1) + (k+1) \geqq d'(A) + d'(B) \geqq d'(A \cup B) + d'(A \cap B) \geqq (k+2) + k$, from which the statement follows. □

If there is at most one maximal dangerous set X with $t \in X$, then for any edge sv with $v \notin X$ the pair st, sv is splittable. Such an edge exists since otherwise $d'(V - X) = d'(X + s) = d'(X) - d'(s) \leqq (k+1) - 2 = k - 1$, contradicting (4.2).

Suppose that X and Y are two distinct maximal dangerous sets with $t \in X \cap Y$ for which $M := X \cap Y$ is maximal. Then X and Y are intersecting, and Claim A implies that there is an edge sv with $v \notin X \cup Y$.

CLAIM. *The pair sv, st is splittable.*

Proof. Suppose that, indirectly, there is a maximal dangerous set Z with $t, v \in Z$. Applying Claim B to $A := X$ and $B := Y$ we have $d'(M) = k$. Z and M must not be intersecting for otherwise Claim B could be applied to $A := X$ and $B := M$ implying $d'(M) = k + 1$. Therefore $M \subseteq Z$ and by the maximal choice of M we have $X \cap Z = Y \cap Z = M$. By Claim A $\bar{d}'(X, Y) = \bar{d}'(Z, Y) = \bar{d}'(Z, X) = 1$, and therefore no other edge than st can leave M, contradicting $k \geqq 2$. □

Theorem 4.1 can be extended to a degree-constrained case when upper and lower bounds are imposed on the degrees of the graph of newly added edges. Let $f \leqq g$ be two nonnegative integer-valued functions on V (infinite values are allowed for g).

THEOREM 4.6. *Given an undirected graph $G = (V, E)$ and an integer $k \geqq 2$, G can be made k-edge-connected by adding a set F of precisely γ new edges so that*

$$f(v) \leqq d_F(v) \leqq g(v) \tag{4.4}$$

holds for every node v of G if and only if $2\gamma \leqq g(V)$ and

$$k - d(X) \leqq g(X), \tag{4.5}$$

holds for every subset $\emptyset \subset X \subset V$ and

$$\Sigma(k - d(X_i) : i = 1, \cdots, t) + f(X_o) \leqq 2\gamma \tag{4.6}$$

holds for every partition $\{X_o, X_1, X_2, \cdots, X_t\}$ of V where X_o may be empty.

Remark. This theorem, when applied to $f \equiv 0$, $g \equiv \infty$, immediately implies the theorem of Cai and Sun and, therefore, it would not have been necessary to prove first Theorem 4.1. We did so to exhibit the simplicity of the idea behind the proof. The next proof uses the very same idea along with some technicalities.

Proof. The necessity of the conditions is straightforward. To see the sufficiency, first, add a new node s to V and $\min(g(v), f(v) + k)$ new parallel edges between s and v for every $v \in V$. For the enlarged graph G', the number $d'(v) - d(v)$ of new edges

incident to v is at most $g(v)$ for every $v \in V$ and, by (4.5), (4.2) holds. If $d'(s) < 2\gamma$, add $2\gamma - d'(s)$ appropriate edges leaving s in such a way that $d'(v) - d(v)$ is still at most $g(v)$ $(v \in V)$. This is possible because we have assumed that $2\gamma \leqq g(V)$.

Second, discard new edges, one by one, as long as possible without violating (4.2) and the following inequalities: $d'(s) \geqq 2\gamma$, $d'(v) - d(v) \geqq f(v)$ $(v \in V)$. Let G' denote the final extended graph. By Proposition 4.4 the set $A := \{v \in V: d'(v) - d(v) > f(v)\}$ can be covered by disjoint critical sets $X_1, X_2, \cdots, X_t$. Let $X_o := V - \cup(X_i: i = 1, \cdots, t)$. Applying (4.6) we have $2\gamma \leqq d'(s) = \Sigma(d'(X_i) - d(X_i): i = 1, \cdots, t) + f(X_o) = \Sigma(k - d(X_i) : i = 1, \cdots, t) + f(X_o) \leqq 2\gamma$. Hence $d'(s) = 2\gamma$, and by γ applications of Theorem 4.5 we obtain the desired augmentation of G. □

We are interested in degree-constrained augmentations where the number of new edges does not matter; see the following theorem.

THEOREM 4.7. *Given an undirected graph $G = (V, E)$ and an integer $k \geqq 2$, G can be made k-edge-connected by adding a set F of new edges so that* (4.4) *holds for every node v of G if and only if* (4.5) *holds for every subset $\emptyset \subset X \subset V$ and* (*) *there is no partition $\mathcal{F} := \{X_o, X_1, X_2, \cdots, X_t\}$ of V, where only X_o may be empty, with the following properties: $f(X_o) = g(X_o)$, $g(X_i) = k - d(X_i)$, and $g(V)$ is odd.*

Proof. The necessity of (4.5) is clear. To see the necessity of (*) let $\mathcal{F}$ be a partition with the given properties, and F a set of new edges satisfying the requirements. Then $d_F(v) = g(v)$ for $v \in X_o$, $d_F(X_i) = k - d(X_i) = g(X_i)$ for $i = 1, \cdots, t$, and, furthermore, no X_i induces elements of F. Therefore $g(V) = \Sigma(g(X) : X \in \mathcal{F}) = 2|F|$, an even number.

To see the sufficiency, extend the graph with a new node s and add $\min(g(v), f(v) + k)$ new parallel edges from s to v for every $v \in V$. Let $V' := V + s$ and let $G' = (V', E')$ denote the extended graph. The number of new (parallel) edges between s and v is $d'(v) - d(v)$.

By this construction $f(v) \leqq d'(v) - d(v) \leqq g(v)$, and (4.5) implies that (4.2) holds for every subset $\emptyset \subset X \subset V$. Therefore, if $d'(s)$ is even, then Lovász's Theorem 4.5 implies the theorem.

Suppose that $d'(s)$ is odd. If there is a node $v \in V$ with $d'(v) - d(v) < g(v)$, then by adding one more edge sv to G' we are at the case of $d'(s)$ even.

Therefore, we can assume that $d'(v) - d(v) = g(v)$ for every $v \in V$. If there is an edge $e = su$ for which $f(u) < g(u)$ and su does not enter any critical set, then e can be deleted without destroying (4.4) and $f(v) \leqq d'(v) - d(v)$, and then $d'(s)$ becomes again even. So suppose that there is no such an edge; that is, every edge sv either enters a critical set or has $f(v) = g(v)$.

Then, by Proposition 4.4, there are disjoint critical sets $X_1, X_2, \cdots, X_t$ for which $k = d'(X_i) = d(X_i) + g(X_i)$ so that $\cup X_i$ contains all nodes v with $f(v) < g(v)$. Let $X_0 := \{v \in V: f(v) = g(v), v \notin \cup X_i\}$. We obtain that $f(X_o) = g(X_o)$, $g(X_i) = k - d(X_i)$ for $i = 1, \cdots, t$, and $g(V) = \Sigma(g(X) : X \in \mathcal{F}) = d'(s)$ is odd, contradicting (*). □

5. Generalization. In this section we exhibit a natural generalization of results from the preceding section. Let $G = (V, E)$ be an undirected graph and $r(u, v)$ $(u, v \in V)$ a nonnegative integer-valued function on the pair of nodes that serves as the demand for edge-connectivity between u and v. When can G be extended by adding γ new edges so that in the extended graph G' the edge-connectivity number $\lambda'(u, v)$ is at least $r(u, v)$ for every pair of nodes u, v? Such an augmentation is called *good* (with respect to the demand $r(u, v)$). It will be convenient to assume (and this can be done without loss of

generality) that

(5.0a) $$r(u,v) \geqq \lambda(u,v) \text{ for every } u,v \in V, \quad \text{and}$$

(5.0b) $$r(u,x) \geqq 1 \quad \text{and} \quad r(v,x) \geqq 1 \text{ imply that } r(u,v) \geqq 1.$$

By Menger's Theorem 2.1, G' is a good augmentation of G if

(5.1) $$d'(X) \geqq R(X)$$

holds for every set $\emptyset \subset X \subset V$ where $R(X) := \max(r(u, v): u \in X, v \in V - X)$. (The maximum on the empty set is defined to be 0.)

In order to obtain more general results on optimal augmentations, we need stronger theorems about splitting-off.

In a graph $G' = (V + s, E')$, let $\lambda'(X) := \max(\lambda'(u, v): u \in X, v \in V - X)$. Obviously $d'(X) \geqq \lambda'(X)$. We call a pair $\{su, sv\}$ of edges of G' *splittable* if after splitting off $\{su, sv\}$ the edge-connectivity between every two nodes distinct from s remains the same. Obviously, $\{su, sv\}$ is splittable precisely if there is no subset $X \subseteq V$ with u, $v \in X$ for which $d'(X) \leqq \lambda'(X) + 1$. We call such a set X *dangerous*.

Mader [26] proved the following extremely powerful result.

THEOREM 5.1 ([26]). *Let $G' = (V + s, E')$ be a connected undirected graph with $d'(s) \neq 3$ or 1.*

(a) *If s is not a cut-node (that is, $G' - s$ is connected), then there is a splittable pair of edges $\{su, sv\}$.*

(b) *If s is a cut-node but there is no cut-edge incident to s, then any pair of edges $\{su, sv\}$ is splittable provided that u and v belong to distinct components of $G' - s$.*

Remarks. The original proof of this theorem is rather difficult. In [12] a relatively simple proof is given. It is not necessarily true that, under the above assumptions, for a given edge st there is an edge su so that st and su are splittable. Furthermore, the theorem does not hold in general if $d'(s) = 3$, as is shown by a complete graph on four nodes. Note that Theorem 4.5 is a special case of Mader's theorem.

COROLLARY 5.2. *Suppose that in an undirected graph $G' = (V + s, E')$ degree $d'(s)$ is even and there is no cut-edge incident to s. Then the edge incident to s can be paired in such a way that splitting off each pair results in a graph with vertex set V in which the edge-connectivity between every two nodes u, v is equal to the original edge-connectivity $\lambda'(u, v)$.*

Proof. Apply Theorem 5.1 $d'(s)/2$ times and observe that after a splitting no edge incident to s becomes a cut-edge. □

Let us turn back to the augmentation problem. We introduce the following notation: $q(A) := R(A) - d(A)$. That is, $q(A)$ is the deficiency of $A \subseteq V$. The following condition is clearly necessary for the existence of a good augmentation using at most γ new edges:

(5.2) $$\Sigma q(X_i) \leqq 2\gamma$$

for every subpartition $\{X_1, \cdots, X_t\}$ of V. Theorem 4.1 of Cai and Sun asserted that, in the special case when $r(u, v) \equiv k \geqq 2$, (5.2) is sufficient as well. However, it is not sufficient, in general, as is shown by the empty graph on four nodes with $r(u, v) \equiv 1$.

Let C ($\neq V$) be a component of G. We call C a *marginal* component (with respect to the demand function $r(u, v)$) if $q(X) \leqq 0$ for every $X \subset C$ and $q(C) \leqq 1$. This is equivalent to saying that $r(u, v) \leqq \lambda(u, v)$ for $u, v \in C$ and $r(u, v) \leqq \lambda(u, v) + 1$ for $u \in C$, $v \in V - C$.

Our solution to the problem of finding a good augmentation consists of two steps. First, we show that marginal components can be easily eliminated; second, we prove that if there are no marginal components, then (5.2) is sufficient.

Let $\gamma(G, r)$ denote the minimum number of edges the addition of which to G results in a graph G' satisfying (5.1). Let C be a marginal component of G, and $G_1 := G - C$. Let r_1 denote the function r restricted on the pairs of nodes of $V - C$.

THEOREM 5.3. *For a marginal component C of G $\gamma(G, r) = \gamma(G_1, r_1) + q(C)$.*

Proof. Let $\gamma := \gamma(G, r)$ and $\gamma_1 := \gamma(G_1, r_1)$. First, we show that $\gamma \leqq \gamma_1 + q(C)$. Let G'_1 denote a minimal augmentation of G_1. If $q(C) = 0$, then clearly G'_1, along with C, yields a good augmentation of G and hence $\gamma \leqq \gamma_1 = \gamma_1 + q(C)$. If $q(C) = 1$, then there is a pair a, b with $a \in C$ and $b \in V - C$ for which $r(a, b) = 1$. We claim that adding C and an edge ab to G'_1 yields a good augmentation G' of G. Indeed, if this were not true, there would be a pair s, t of nodes of G for which $r(s, t) > \lambda'(s, t)$. Then precisely one of s and t, say s, is in C and t in $V - C$ (because C is marginal and G'_1 is good with respect to r_1). Since C is marginal, $r(s, t) = 1$ and then $\lambda'(s, t) = 0$. Hence in G'_1 there is no path between t and b, and therefore $r_1(t, b) = r(t, b) = 0$. (5.0b) shows that a and s must be distinct. Then, by (5.0a) $r(s, a) \geqq 1$. Applying (5.0b) twice, we obtain that $r(a, t) \geqq 1$ and $r(b, t) \geqq 1$, a contradiction. Therefore $\gamma \leqq \gamma_1 + q(C)$.

To see the other direction let $G_o = (V_o, E_o)$ be a graph obtained from G by replacing C with a new node v_c. Define $r_o(u, v) := r(u, v)$ if $u, v \in V - C$ and $r_o(v_c, v) := q(C)$. Let $\gamma_o := \gamma(G_o, r_o)$. Obviously, $\gamma_o \leqq \gamma$.

Let $G'_o = (V_o, E_o \cup F)$ be a minimal augmentation of G_o good with respect to r_o such that the number t of elements in F incident to v_c is as small as possible. If $t = 0$, then $q(C) = 0$ and the elements of F are induced by $G - C$. Hence $\gamma_1 \leqq |F| = \gamma_o \leqq \gamma = \gamma - q(C)$, as required. If $t = 1$, then, by the minimality of F, $q(C) = 1$. Let $f \in F$ be the edge incident to v_c. Adding $F - f$ to G_1, we obtain a good augmentation of G_1 and then $\gamma_1 \leqq |F| - 1 = \gamma_o - 1 \leqq \gamma - q(C)$, as required.

Finally, suppose that $t \geqq 2$ and let $v_c u_1, \cdots, v_c u_t$ be the t edges in F incident to v_c. Let F' be obtained from F by replacing $v_c u_i$ by $u_1 u_i$ ($i = 2, 3, \cdots, t$). It is not hard to see that $G_o + F'$ is also a good (and minimal) augmentation of G_o, contradicting the minimal choice of t. □

By Theorem 5.3 we can easily reduce the augmentation problem to a case when there is no marginal component. Namely, proceed as follows. Let $C_1, C_2, \cdots, C_t$ be components of G such that C_i is a marginal component of $G - (C_1 \cup \cdots \cup C_{i-1})$ ($i = 1, \cdots, t$) and $G - (C_1 \cup \cdots \cup C_t)$ has no marginal components. Leave out each C_i and find a minimal augmentation of the remaining graph (as to be described below). Take back the components C_i, and for each component C_i add $q(C_i)$ (which is 0 or 1) new edges, as described in the first part of the proof of Theorem 5.3.

Before formulating the main result of this section we prove the following.

PROPOSITION 5.4. *For arbitrary $X, Y \subseteq V$ at least one of the following inequalities holds*:

$$R(X) + R(Y) \geqq R(X \cap Y) + R(X \cup Y), \tag{5.3a}$$

$$R(X) + R(Y) \geqq R(X - Y) + R(Y - X). \tag{5.3b}$$

Proof. Suppose that $R(X) = r(x, x')$ and $R(Y) = r(y, y')$, where X separates x and x', and Y separates y and y'. Assume first that one of the two pairs, say x, x', is separated by both X and Y. By taking the complement of Y, if necessary, we can assume that $x \in X - Y$ and $x' \in Y - X$. (If Y is replaced by its complement, then (5.3a) and (5.3b) transform into each other.) If y, y' are separated by X, then $R(Y) = R(X) \geqq \min(R(X - Y), R(Y - X))$, and (5.3b) follows.

If y, y' are not separated by X, then either one of y and y', say y, is in $X \cap Y$ and $y' \in X - Y$; or else one of y and y', say y, is in $Y - X$ and $y' \in V - (X \cup Y)$. In the first

case, $R(X-Y) \geqq R(Y)$ and $R(Y-X) \geqq R(X)$, and (5.3b) follows. In the second case, $R(Y-X) \geqq R(Y)$ and $R(X-Y) \geqq R(X)$, and (5.3b) follows again.

Finally, assume that neither x, x' are separated by Y nor are y, y' separated by X. Again, we can assume that $x \notin Y$. Then $x' \in V-(X \cup Y)$. Now either one of y and y', say y, is in $X \cap Y$ and $y' \in X-Y$; or else one of y and y', say y, is in $Y-X$ and $y' \in V-(X \cup Y)$. In the first case, $R(X) \leqq R(X \cup Y)$ and $R(Y) \leqq R(X \cap Y)$, from which (5.3a) follows. In the second case, $R(X) \leqq R(X-Y)$ and $R(Y) \leqq R(Y-X)$, and (5.3b) follows. □

The main result of this section is as follows.

THEOREM 5.5. *If G has no marginal components, there is a good augmentation using at most γ new edges if and only if* (5.2) *holds for every subpartition $\{X_1, \cdots, X_t\}$ of V.*

Proof. The following lemma and Corollary 5.2 imply the theorem.

LEMMA 5.6. *G can be extended to a graph $G'=(V+s, E')$ by adding a new node s, and 2γ new edges between V and s so that none of the new edges is a cut-edge of G' and for every subset $\emptyset \subseteq X \subseteq V$*

$$d'(X) \geqq R(X) \tag{5.4}$$

holds where d' denotes the degree function of G'.

Proof. First, add a sufficiently large number of edges leaving s so as to satisfy (5.4). Second, discard new edges, one by one, as long as possible without violating (5.4). Let G' denote the final extended graph.

Claim. $d'(s) \leqq 2\gamma$.

Proof. We call a set $\emptyset \subset X \subseteq V$ *critical* if $d'(X)=R(X)$.

PROPOSITION 5.7. *If X and Y are critical sets, then at least one of the following statements holds*:

(5.5a) *both $X \cap Y$ and $X \cup Y$ are critical*;

(5.5b) *both $X-Y$ and $Y-X$ are critical and $\bar{d}'(X, Y)=0$.*

Proof. If (5.3a) holds, then $R(X)+R(Y)=d'(X)+d'(Y) \geqq d'(X \cap Y)+d'(X \cup Y) \geqq R(X \cap Y)+R(X \cup Y) \geqq R(X)+R(Y)$ and (5.5a) follows.

If (5.3b) holds, then $R(X)+R(Y)=d'(X)+d'(Y)=d'(X-Y)+d'(Y-X)+2\bar{d}'(X, Y) \geqq R(X-Y)+R(Y-X)+2\bar{d}'(X, Y) \geqq R(X)+R(Y)+2\bar{d}'(X, Y)$ and (5.5b) follows.

Let $S:=\{u \in V: su \in E'\}$. An edge su cannot be left out without violating (5.4) precisely if there is a critical set containing u. Let $\mathcal{F}:=\{X_1, X_2, \cdots, X_t\}$ be a family of critical sets that cover S so that t is minimal and, given this minimal t, $\Sigma|X_i|$ is minimal. We claim that the sets X_i's are disjoint.

Indeed, for $X, Y \in \mathcal{F}$ their union $X \cup Y$ cannot be critical by the minimality of t. Therefore (5.5b) must apply. Hence $X-Y$ and $Y-X$ are both critical and $\bar{d}'(X, Y)=0$, from which $S \cap (X \cap Y)=\emptyset$. This means that if we replace X and Y by $X-Y$ and $Y-X$, then we obtain another family of t critical sets covering S. By the minimal choice of $\Sigma|X_i|$ we have that $|X|=|X-Y|$ and $|Y|=|Y-X|$; that is, X and Y are disjoint.

By (5.2) we have $d'(s)=\Sigma(d'(X_i)-d(X_i): i=1, \cdots, t)=\Sigma(q(X_i): i=1, \cdots, t) \leqq 2\gamma$, which proves the claim.

By adding one extra edge (parallel to an existing edge su), if $d'(s)$ is odd, we can assume that $d'(s)=2\gamma'$ ($\leqq 2\gamma$). We claim that no edge incident to s is a cut edge of G'. Indeed, if $e=sv$ were a cut-edge, then let C be the component of $G'-e$ containing v but not s. There is precisely one edge in G' leaving C and therefore C must be a marginal component of G contradicting the assumption.

This way the proof of Lemma 5.6 is complete and so is the proof of the theorem. □

We mention two degree-constrained versions of Theorem 5.5. Their proofs are analogous, respectively, to those of Theorems 4.6 and 4.7, and we do not repeat them here. However, a proof will be provided in § 7 relying on a relationship between good augmentations and polymatroids.

THEOREM 5.8. *Suppose that* $f(C) \geqq 2$ *for every marginal component* C *of* G. *There is a good augmentation of* G *using a set* F *of precisely* γ *new edges so that* (4.4) *holds if and only if* $2\gamma \leqq g(V)$ *and*

$$q(X) \leqq g(X), \tag{5.6}$$

holds for every subset $\emptyset \subset X \subset V$ *and*

$$\Sigma(q(X_i): i = 1, \cdots, t) + f(X_o) \leqq 2\gamma \tag{5.7}$$

holds for every partition $\{X_o, X_1, X_2, \cdots, X_t\}$ *of* V *where* X_o *may be empty*.

THEOREM 5.9. *Suppose that* $f(C) \geqq 2$ *for every marginal component* C *of* G. *There is a good augmentation of* G *using a set* F *of new edges so that* (4.4) *holds if and only if* (5.6) *holds for every subset* $\emptyset \subset X \subset V$ *and* (*) *there is no partition* $\mathcal{F} := \{X_o, X_1, X_2, \cdots, X_t\}$ *of* V, *where only* X_o *may be empty, with the following properties*: $f(X_o) = g(X_o)$, $g(X_i) = q(X_i)$, *and* $g(V)$ *is odd*.

To close this section we consider minimum cost augmentations. As we mentioned in the Introduction, if costs are assigned to the edges, the problem is NP-complete even if $r \equiv 2$. Suppose now that $c: V \to R_+$ is a nonnegative cost function on the node-set V of G. Our object is to find a good augmentation of G for which $\Sigma d_F(v)c(v)$ is minimum, where F is the set of newly added edges.

We are concerned only with the case when G has no marginal components. If G does have marginal components, a reduction analogous to the one described in Theorem 5.3 can be applied.

Assume that the elements of V are ordered in such a way that $c_1 \geqq c_2 \geqq \cdots \geqq c_n$ where $c_i := c(v_i)$. Let $k := \max(q(X): X \subseteq V)$.

ALGORITHM TO FIND A GOOD AUGMENTATION OF MINIMUM NODE-COST

First, add a new node s to V, and k new parallel edges between s and v for every $v \in V$. For the resulting graph G', (5.1) holds; that is, $d'(X) \geqq R(X)$ for every $X \subseteq V$.

Assume that the new edges $f_1, f_2, \cdots$ are ordered according to the decreasing order of their end-node u_i. That is, first come the parallel edges from s to u_1, then the parallel edges from s to u_2, and so on. (The order of parallel edges between s and u_i does not matter.)

Next, go through the new edges in the given order and discard an f_i if this can be done without destroying (5.1). If at the end of the procedure there is an odd number of edges incident to s, add one further edge between s and v_n and let $G' = (V + s, E')$ denote the final graph. Note that the newly added edge is not a cut-edge of G', as we assumed that G has no marginal components.

Therefore we can apply Corollary 5.2 to G'. Let $G_1 = (V, E \cup F)$ denote the resulting graph.

THEOREM 5.10. *The graph* G_1 *constructed above is a minimum-cost good augmentation of* G.

This theorem will be proved in § 7, where the necessary tools from polymatroid theory are already available.

6. Generalized polymatroids. Before going into the details, let us indicate how polymatroid theory became involved in this research. The starting point was the paper of Cai and Sun [2]. We were trying to find a simple proof of their theorem when we realized that the degree vectors of good augmentations span a (generalized) polymatroid. This observation led to the desired proof and almost immediately to a proof of the directed version. The nice properties of polymatroids gave rise to the solution of the degree-constrained and the minimum node-cost augmentation problems.

Since there may be those who are interested in the augmentation problem but have no prior knowledge in polymatroid theory, some of the original proofs are converted here to avoid polymatroids. These proofs are included in the preceding sections. However, we do not want to hide this relationship, so this and the next two sections have been inserted.

First, we collect some results from polyhedral combinatorics. This environment helps us understand the background behind the results occurring in earlier sections. It also gives rise to possible generalizations and a proof of the two algorithms described at the end of §§ 3 and 5.

Generalizing the concept of matroid polyhedra, Edmonds [4] defined a polymatroid to be a polyhedron $P(b) := \{x \in R^V : x \geqq 0,\ x(A) \leqq b(A) \text{ for every } A \subseteq V\}$, where b is a submodular, monotone-increasing, finite-valued set-function with $b(\varnothing) = 0$. There are other classes of polyhedra of similar type. For example, Shapley [29] introduced, as we call it here, contrapolymatroids. Submodular polyhedra and basis polyhedra have been defined and investigated by Fujishige. For a general account, see [15]. Generalized polymatroids (in short, g-polymatroids) [11] serve as a common framework for all of these polyhedra.

Throughout this section we assume that any function in question is integer-valued (allowing $\pm\infty$).

Let $p : 2^V \to Z \cup \{-\infty\}$ be a supermodular function, and $b : 2^V \to Z \cup \{\infty\}$ a submodular function with $p(\varnothing) = b(\varnothing) = 0$ for which

$$b(X) - p(Y) \geqq b(X - Y) - p(Y - X) \tag{6.1}$$

holds for every $X, Y \subseteq V$.

A pair (p, b) with the above properties is called a *strong pair*. A polyhedron $Q(p, b) := \{x \in R^V : p(A) \leqq x(A) \leqq b(A) \text{ for every } A \subseteq V\}$ is called a *g-polymatroid*. For technical convenience, we consider the empty set a g-polymatroid. For a detailed account on properties of g-polymatroids, see [13]. Here we cite some without proof.

PROPOSITION 6.1. *A g-polymatroid $Q = Q(p, b)$ is nonempty and is spanned by its integral points. Q uniquely determines its defining strong pair, namely, $p(A) = \min(x(A) : x \in Q)$ and $b(A) = \max(x(A) : x \in Q)$.*

A pair (p', b') is called a *weak pair* if p' (respectively, b') is supermodular (respectively, submodular) only on intersecting sets, and (6.1) is required only for intersecting X and Y.

PROPOSITION 6.2. *For a weak pair (p', b') the polyhedron $Q = Q(p', b')$ is a g-polymatroid. Q is nonempty if and only if*

$$\Sigma p'(Z_i) \leqq b'(\cup Z_i) \quad \textit{and} \tag{6.2a}$$

$$\Sigma b'(Z_i) \geqq p'(\cup Z_i) \tag{6.2b}$$

hold for every subpartition $\{Z_1, \cdots, Z_t\}$ of V. If Q is nonempty, it contains an integer point.

Let $Q = Q(p, b)$ be a g-polymatroid defined by a strong pair (p, b). Let $f: V \to Z \cup \{-\infty\}$ and $g: V \to Z \cup \{\infty\}$ be two functions with $f \leqq g$ and let $-\infty \leqq \alpha \leqq \beta \leqq \infty$ be two integers.

PROPOSITION 6.3. *The intersection Q_o of a plank $\{x \in R^V: \alpha \leqq x(V) \leqq \beta\}$ and a g-polymatroid $Q(p, b)$ is a g-polymatroid. If Q_o is nonempty, its unique strong pair (p_o, b_o) is given by*

$$p_o(X) = \max(p(X), \alpha - b(V-X)), \tag{6.3a}$$

$$b_o(X) = \min(b(X), \beta - p(V-X)). \tag{6.3b}$$

PROPOSITION 6.4. *The intersection Q_1 of a box $\{x \in R^V: f \leqq x \leqq g\}$ and a g-polymatroid $Q(p_o, b_o)$ is a g-polymatroid. If Q_1 is nonempty, its unique strong pair (p_1, b_1) is given by*

$$p_1(X) = \max(p_o(Y) + f(X-Y) - g(Y-X): Y \subseteq V), \tag{6.4a}$$

$$b_1(X) = \min(b_o(Y) + g(X-Y) - f(Y-X): Y \subseteq V). \tag{6.4b}$$

PROPOSITION 6.5. *For a g-polymatroid $Q = Q(p, b)$ if $x \leqq y \leqq z$ are vectors so that $x, z \in Q$, then $y \in Q$.*

The greedy algorithm can also be extended to work on g-polymatroids Q. We need it only in the special case when the objective function $c: V \to R_+$ is nonnegative (c need not be integer-valued) and Q is bounded from below. (By Proposition 6.1 this is equivalent to requiring p to be finite.) The objective is to minimize cx over Q.

Suppose that the elements of V are ordered so that $c_1 \geqq c_2 \geqq \cdots \geqq c_n$.

PROPOSITION 6.6. *If Q is given by a strong pair (p, b), then* $\min(cx: x \in Q)$ *is attained by a vector z where $z_t = p(v_1, \cdots, v_t) - p(v_1, \cdots, v_{t-1})$ $(t = 1, \cdots, n)$.*

This proposition has a useful corollary, which shows that the greedy algorithm may be applied even if Q is not given by its strong pair.

COROLLARY 6.7. *Let Q be a g-polymatroid bounded from below. Define iteratively the components of a vector z as follows. Suppose $z_1, z_2, \cdots, z_{t-1}$ have already been defined. Let z_t be the smallest number for which $(z_1, \cdots, z_t, x_{t+1}, \cdots, x_n)$ belongs to Q for some appropriate $x_{t+1}, \cdots, x_n$. Then z is an integer-valued solution to* $\min(cx: x \in Q)$.

Note that the procedure in the corollary has nothing to do with the form in which Q is given. Therefore, it becomes a usable algorithm only if there is a way to compute the current z_t.

Actually, we will use the properties listed above mainly for the special case of contrapolymatroids. Let p be a supermodular, monotone-increasing (integer-valued) set-function with $p(\varnothing) = 0$. A polyhedron $C(p) := \{x \in R : x(A) \geqq p(A)$ for every $A \subseteq V\}$ is called a *contrapolymatroid*. A contrapolymatroid is a g-polymatroid, since $C(p) = Q(p, b)$ where $b :\equiv \infty$ (except $b(\varnothing) = 0$), and this (p, b) is a strong pair.

In applications we will encounter contrapolymatroids that are not given by their unique monotone supermodular function. Let $q: 2^V \to Z_+$ be a nonnegative integer-valued function. Suppose that $Q := \{x \in R^V: x(A) \geqq q(A)$ for every $A \subseteq V\}$ is a contrapolymatroid. Let $k := \max(q(X): X \subseteq V)$. Then, obviously, $(k, k, \cdots, k) \in Q$. For Q the greedy algorithm can be formulated as follows.

COROLLARY 6.7′. *Define iteratively the components of a vector y as follows. Suppose $y_1, y_2, \cdots, y_{t-1}$ have already been defined. Let y_t be the smallest number for which $(y_1, \cdots, y_t, k, \cdots, k)$ belongs to Q. Then y is an integer-valued solution to* $\min(cx: x \in Q)$.

Proof. We show that $z = y$ where z is the vector constructed in Corollary 6.7. If this were not true, then there is a smallest subscript t for which $z_t \neq y_t$. Obviously, $z_t < y_t$. Let $V_t := \{v_1, \cdots, v_t\}$. By the definition of y_t, there is a set A that prevented y_t from being smaller. This means that $v_t \in A$ and $q(A) = y(V_t \cap A) + k|A - V_t|$. Now if $A \subseteq V_t$, then $q(A) = y(V_t \cap A) > z(V_t \cap A) = z(A)$, contradicting that $z \in Q$. If $A \not\subseteq V_t$, then $q(A) \geqq y_t + k \geqq q(A)$ (by the definition of k). Hence $0 \geqq z_t < y_t = 0$, a contradiction. □

Let Q be the same as before. We will have to be able to solve the following optimization problem:

$$(*) \qquad \min (cx: x \in Q, x(V) \text{ is even}).$$

By applying the greedy algorithm, compute an integer vector $z' \in Q$ that minimizes cx over Q. If $z'(V)$ is even, let $z := z'$. If $z'(V)$ is odd, revise z' by increasing $z'(v_n)$ by 1. Let z denote the resulting vector.

PROPOSITION 6.8. *Vector z is an optimal solution to* $(*)$.

Proof. Suppose that $Q = C(p)$, where p is the unique supermodular function defining Q. If $p(V)$ is odd, modify p by increasing $p(V)$ by 1. The resulting p_o is fully supermodular and monotone increasing. The proposition follows by observing that the greedy algorithm described in Corollary 6.7′, when applied to $C(p_o)$, outputs vector z constructed above. □

Let $Q := C(p)$ be a contrapolymatroid, let $f: V \to Z$ and $g: V \to Z \cup \{\infty\}$ be two functions with $f \leqq g$, and let $0 \leqq \alpha \leqq \beta \leqq \infty$ be two integers. Let $Q_1 := \{x \in Q: f \leqq x \leqq g, \alpha \leqq x(V) \leqq \beta\}$.

PROPOSITION 6.9. *Q_1 is a g-polymatroid. Q_1 is nonempty if and only if*

$$(6.5) \qquad \alpha \leqq g(V) \text{ and}$$

$$(6.6) \qquad p(X) + f(V - X) \leqq \min(\beta, g(V)),$$

$$(6.7) \qquad p(X) \leqq g(X)$$

hold for every subset $X \subseteq V$.

Proof. By Propositions 6.3 and 6.4, it follows that Q_1 is a g-polymatroid.

Let $b(X) := \infty$ if $X \neq \emptyset$ and $b(\emptyset) = 0$. By applying Proposition 6.3 to this p and b, we obtain that $Q_o := \{x \in Q: \alpha \leqq x(V) \leqq \beta\}$ is a g-polymatroid with strong pair (p_o, b_o), where $p_o(X) := p(X)$ if $X \subset V$, $p_o(V) := \max(p(V), \alpha)$, and $b_o(X) := \beta - p(V - X)$ if $X \neq \emptyset$.

Define p' and b' as follows. $p'(X) := \max(p_1(X), f(X))$, $b'(X) := \min(b_1(X), g(X))$. By Proposition 6.4 (p', b') is a weak pair, and $Q_1 = Q(p', b')$. Therefore, Proposition 6.2 applies, and (6.2) in this case is equivalent to (6.5)–(6.7). □

Let (p_1, b_1) denote the strong pair defining Q_1. From (6.3) and (6.4) we can read off that

$$(6.8a) \qquad p_1(V) = \max[\alpha, \max(p(Y) + f(V - Y): Y \subseteq V)] \quad \text{and}$$

$$(6.8b) \qquad b_1(V) = \min(\beta, g(V)).$$

PROPOSITION 6.10. *Suppose that Q_1 is nonempty. Q_1 contains no integer vector y with $y(V)$ even if and only if $p_1(V) = b_1(V)$ is odd.*

Proof. Obviously, if $m := p_1(V) = b_1(V)$, then $x(V) = m$ for every $x \in Q_1$; therefore, if m is odd, $x(V)$ is odd as well. Conversely, if $p_1 \neq b_1$, then, by Proposition 6.1, there is an integer vector $x \in Q_1$ with $x(V) = p_1(V)$, and there is an integer vector $z \in Q_1$ with $z(V) = b_1(V)$. By applying Proposition 6.5, we obtain that there is an integer

vector $y \in Q$ with $y(V)$ even. If $p_1(V) = b_1(V)$ is even, then any integer vector of Q_1 will do. □

Finally, we mention one more result from [13].

PROPOSITION 6.11. *The intersection Q of two g-polymatroids $Q(p_1, b_1)$ and $Q(p_2, b_2)$ defined by strong pairs is nonempty if and only if $p_1 \leqq b_2$ and $p_2 \leqq b_1$. Moreover, Q is spanned by its integer points.*

7. Undirected augmentations and *G*-polymatroids. In this section we reveal a relationship between augmentations and g-polymatroids. First, let us consider the augmentation problem analysed in § 5 and recall the definition of a good augmentation. In Lemma 5.6 we showed how to extend G by a new node s and some new edges incident to s so as to satisfy (5.4). Given such an extension, let $z(v)$ denote the number of parallel edges between $v \in V$ and s. (5.4) is clearly equivalent to

$$z(A) \geqq q(A) \text{ for every } A \subseteq V. \tag{7.1}$$

Note that $q(A) := R(A) - d(A)$ denotes the deficiency of $A \subseteq V$. Also note that q is not intersecting supermodular, in general. Still, the following theorem asserts that q defines a contrapolymatroid.

THEOREM 7.1. *$Q := \{ z \in R^V : z \geqq 0$ and z satisfies (7.1)$\}$ is a contrapolymatroid $C(p)$, where the unique supermodular function defining Q is*

$$p(A) := \max (\Sigma q(A_i) : \{A_1, A_2, \cdots, A_t\} \text{ a subpartition of } A, A_i \neq \varnothing). \tag{7.2}$$

Proof. First, we show that $C(p) = Q$. Indeed, $C(p) \subseteq Q$ since $\{A\}$ is a subpartition of A. On the other hand, let $z \in Q$ and assume that $p(A) = \Sigma_i q(A_i)$ for some subpartition $\{A_1, A_2, \cdots, A_t\}$ of A ($A_i \neq \varnothing$). Since z satisfies (7.1) and is nonnegative, we have $z(A) = \Sigma_i z(A_i) + z(A - \cup A_i) \geqq \Sigma_i q(A_i) = p(A)$, and therefore $z \in C(p)$.

As p is clearly monotone increasing, all we have to show is that p is supermodular. Let A and B be two arbitrary subsets of V. Assume that $p(A) = \Sigma q(A_i)$ for some subpartition $\{A_1, A_2, \cdots, A_k\}$ of A, and let $p(B) = \Sigma q(B_i)$ for some subpartition $\{B_1, \cdots, B_h\}$ of B.

Let $\mathcal{F} := \{A_1, \cdots, A_k, B_1, \cdots, B_h\}$. Then $\mathcal{F}$ satisfies the following:

$$\begin{aligned} &\text{every } v \in A \cap B \text{ is covered at most twice,} \\ &\text{every } v \in (A-B) \cup (B-A) \text{ is covered at most once by } \mathcal{F}. \end{aligned} \tag{7.3}$$

By Propositions 2.2 and 5.4, q satisfies at least one of the following inequalities for every two subsets X, Y of V:

$$q(X) + q(Y) \leqq q(X \cap Y) + q(X \cup Y), \tag{7.4a}$$

$$q(X) + q(Y) \leqq q(X - Y) + q(Y - X). \tag{7.4b}$$

Denote $q(\mathcal{F}) := \Sigma(q(X) : X \in \mathcal{F})$. Assume that there are two intersecting sets A_i and B_j in $\mathcal{F}$. If $X := A_i$ and $Y := B_j$ satisfy (7.4a) (respectively, (7.4b)), revise $\mathcal{F}$ by replacing A_i and B_j by $X \cap Y$ and $X \cup Y$ (respectively, $X - Y$ and $Y - X$). Then the new family $\mathcal{F}_1$ satisfies (7.3), and by (7.4), $q(\mathcal{F}_1) \geqq q(\mathcal{F})$.

Apply this "uncrossing" operation as long as there are intersecting sets. Since in every step $\Sigma(|X|^2 - 2|V||X| : X \in \mathcal{F})$ strictly increases, after a finite number of steps we obtain an $\mathcal{F}_o$ satisfying (7.3), for which $q(\mathcal{F}_o) \geqq q(\mathcal{F})$ and $\mathcal{F}_o$ is laminar. Let $\mathcal{P}_1$ consist of the minimal members of $\mathcal{F}_o$ that are subsets of $A \cap B$, and $\mathcal{P}_2 := \mathcal{F}_o - \mathcal{P}_1$. Then $\mathcal{P}_1$ is a subpartition of $A \cap B$, and $\mathcal{P}_2$ is a subpartition of $A \cup B$. By definition, $p(A \cap B) \geqq q(\mathcal{P}_1)$ and $p(A \cup B) \geqq q(\mathcal{P}_2)$ so we have $p(A) + p(B) = q(\mathcal{F}) \leqq q(\mathcal{F}_o) = q(\mathcal{P}_1) + q(\mathcal{P}_2) \leqq p(A \cap B) + p(A \cup B)$, as required. □

Theorem 7.1, with the help of Theorem 5.2, provides the following relation between good augmentations of G and integer vectors in $C(p)$.

COROLLARY 7.2. *Let F be a set of new edges and define a vector $z_F \in Z^V$ by $z_F(v) := d_F(v)$ (for $v \in V$). If $(V, E \cup F)$ is a good augmentation, then $z_F \in C(p)$ and $z(V)$ is even. If $z \in C(p)$ is an integer-valued vector with $z(V)$ even and*

$$z(C) \neq 1 \text{ for every component } C \text{ of } G, \tag{7.5}$$

then there is a set F of new edges for which $z(v) = d_F(v)$ (for $v \in V$) and for which $(V, E \cup F)$ is a good augmentation.

Proof. The first part is clear from the definitions. To see the second part, let $z \in C(p)$ be a vector having the required properties. Extend G by a new node s and by $z(sv)$ new parallel edges between s and v ($v \in V$). By the hypotheses, the extended graph G' satisfies the hypotheses of Corollary 5.2, and therefore the required augmentation exists. □

Corollary 7.2 ensures that the results of § 6 concerning g-polymatroids can be utilised for augmentations. Let us first prove Theorems 5.8 and 5.9.

Proof of Theorem 5.8. The necessity of the conditions is clear, and we concentrate only on their sufficiency. Let p be the set-function defined in (7.2), and let $\alpha := \beta := 2\gamma$. We claim that the hypotheses of Theorem 5.8 imply (6.5)–(6.7). Indeed, $2\gamma \leqq g(V)$ implies (6.5). Since g is modular, (5.6) and (7.2) imply (6.7). Similarly, (5.7) and (7.2) imply $p(X) + f(V - X) \leqq \beta$, and by $f \leqq g$ we also have $p(X) + f(V - X) \leqq g(V)$; that is, (6.6) follows.

By Propositions 6.1 and 6.9, Q_1 contains an integer point z, the hypotheses of Corollary 7.2 hold, and therefore the required augmentation exists. □

Proof of Theorem 5.9. Again we are concerned only with the sufficiency. Let p be the set-function defined in (7.2) and let $\alpha := 0$ and $\beta := \infty$. Now (6.5) holds. (5.6) and (7.2) imply (6.7) and (6.6). Therefore Proposition 6.9 applies, and Q_1 is non-empty. We claim that Q_1 contains an integer vector z with $z(V)$ even. If this were not the case, then by Proposition 6.10, $b_1(V) = g(V)$ is odd, and $b_1(V) = p_1(V)$. That is, by (7.2) and (6.8a) we would have $g(V) = \Sigma q(X_i) + f(V - \cup X_i)$ for some subpartition $\{X_1, X_2, \cdots, X_t\}$ of V, contradicting (*) in Theorem 5.9.

We finish by applying Corollary 7.2 to this vector z. Note that (7.5) is satisfied because $f(C) \geqq 2$ for every marginal component C of G. □

Using the same technique, a good characterization can be derived from Propositions 6.9 and 6.10 for the existence of a set F of new edges for which $(V, E \cup F)$ is a good augmentation, $f(v) \leqq d_F(v) \leqq g(v)$ for every $v \in V$ and $\varphi \leqq |F| \leqq \gamma$.

Proof of Theorem 5.10. The theorem follows if we put together Corollaries 7.2 and 6.7′ and Proposition 6.8. □

8. Directed augmentations and G-polymatroids. Let $G = (V, E)$ be a digraph. Suppose that G can be extended by γ new edges to a k-edge-connected digraph; that is, (3.1) and (3.2) hold.

THEOREM 8.1. *$Q_{\text{in}} := \{z \in R^V: z \geqq 0, z(V) \geqq \gamma, z(X) \geqq k - \rho(X)$ for every $\varnothing \subset X \subset V\}$ is a contrapolymatroid $C(p_{\text{in}})$, where*

$$p_{\text{in}}(A) = \max(\Sigma(k - \rho(A_i)): \{A_1, \cdots, A_t\} \text{ a subpartition of } A) \tag{8.1}$$

if $A \subset V$ and $p_{\text{in}}(V) := \gamma$.

Proof. In the proof we will abbreviate p_{in} by p, and Q_{in} by Q. First, we show that $C(p) = Q$. Indeed, $C(p) \subseteq Q$ since $\{A\}$ is a subpartition of A. On the other hand, let $z \in Q$ and assume that $p(A) = \Sigma_i q(A_i)$ for some subpartition $\{A_1, A_2, \cdots, A_t\}$ of

A ($A_i \neq \emptyset$). Since $z(X) \geq q(X)$ for $X \subseteq V$ and z is nonnegative, we have $z(A) = \Sigma_i z(A_i) + z(A - \cup A_i) \geq \Sigma_i q(A_i) = p(A)$, and therefore $z \in C(p)$.

By the definition of γ we have $\gamma \geq \max(\Sigma(k - \rho(X_i)): \{X_1, \cdots, X_t\}$ a subpartition of $V)$, and hence p is monotone increasing. We are going to show that p is fully supermodular. Let $q(X) := k - \rho(X)$ if $\emptyset \subset X \subset V$, and $q(\emptyset) := q(V) := 0$. Then q is supermodular on crossing sets.

Let A and B be two arbitrary subsets of V. Suppose that $p(A) = \Sigma q(A_i)$ for some subpartition $\{A_1, A_2, \cdots, A_k\}$ of A, and let $p(B) = \Sigma q(B_i)$ for some subpartition $\{B_1, \cdots, B_h\}$ of B.

Let $\mathcal{F} := \{A_1, \cdots, A_k, B_1, \cdots, B_h\}$. Then $\mathcal{F}$ satisfies the following:

(8.2) every $v \in A \cap B$ is covered at most twice,
every $v \in (A - B) \cup (B - A)$ is covered at most once by $\mathcal{F}$.

Denote $q(\mathcal{F}) := \Sigma(q(X): X \in \mathcal{F})$. If there are two crossing sets A_i and B_j in $\mathcal{F}$, revise $\mathcal{F}$ by replacing A_i and B_j by $A_i \cap B_j$ and $A_i \cup B_j$. The new family $\mathcal{F}_1$ satisfies (8.2) and, since q is supermodular on crossing sets, $q(\mathcal{F}_1) \geq q(\mathcal{F})$.

Apply this "uncrossing" operation as long as there are crossing sets. Since in every step $\Sigma(|X|^2: X \in \mathcal{F})$ strictly increases, after a finite number of steps the procedure stops with an $\mathcal{F}_o$ satisfying (8.2) for which $q(\mathcal{F}_o) \geq q(\mathcal{F})$.

Assume first that $\mathcal{F}_o$ includes no intersecting sets; that is, $\mathcal{F}_o$ is laminar. Let $\mathcal{P}_1$ consist of the minimal members of $\mathcal{F}_o$ that are subsets of $A \cap B$, and $\mathcal{P}_2 := \mathcal{F}_o - \mathcal{P}_1$. Then $\mathcal{P}_1$ is a subpartition of $A \cap B$, and $\mathcal{P}_2$ is a subpartition of $A \cup B$. By definition, $p(A \cap B) \geq q(\mathcal{P}_1)$ and $p(A \cup B) \geq q(\mathcal{P}_2)$, so we have $p(A) + p(B) = q(\mathcal{F}) \leq q(\mathcal{F}_o) = q(\mathcal{P}_1) + q(\mathcal{P}_2) \leq p(A \cap B) + p(A \cup B)$, as required.

Second, assume that $\mathcal{F}_o$ includes two intersecting sets X and Y. They are not crossing, therefore $X \cup Y = V$. By (8.2), the other members of $\mathcal{F}_o$ are pairwise disjoint subsets of $A \cap B$. Therefore $p(A \cap B) \geq q(\mathcal{F}_o) - q(X) - q(Y)$.

By the assumption, (3.2) holds. Hence $k - \delta(V - X) + k - \delta(V - Y) \leq \gamma$; that is, $q(X) + q(Y) = k - \rho(X) + k - \rho(Y) \leq \gamma$. We have $p(A) + p(B) = q(\mathcal{F}) \leq q(\mathcal{F}_o) \leq p(A \cap B) + q(X) + q(Y) \leq p(A \cap B) + \gamma = p(A \cap B) + p(A \cup B)$, as required. □

By interchanging δ and ρ in Theorem 8.1 we obtain the following result.

THEOREM 8.1′. *$Q_{\text{out}} := \{z \in R^V: z \geq 0, z(V) \geq \gamma, z(X) \geq k - \delta(X)$ for every $\emptyset \subset X \subset V\}$ is a contrapolymatroid $C(p_{\text{out}})$, where*

(8.1′) $p_{\text{out}}(A) = \max(\Sigma(k - \delta(A_i)): \{A_1, \cdots, A_t\}$ *a subpartition of* $A)$

if $A \subset V$ and $p_{\text{out}}(V) := \gamma$.

COROLLARY 8.2. *If F is a set of γ new edges and $(V, E \cup F)$ is k-edge-connected, then $z \in C(p_{\text{in}})$ ($z \in C(p_{\text{out}})$) where $z \in Z$ is defined by $z(v) := \rho_F(v)$ ($z(v) := \delta_F(v)$) for every $v \in V$. Conversely, if $z_{\text{in}} \in C(p_{\text{in}})$ and $z_{\text{out}} \in C(p_{\text{out}})$ are integer-valued vectors with $\gamma := z_{\text{in}}(V) = z_{\text{out}}(V)$, then there is a set F of γ new edges for which $(V, E \cup F)$ is a k-edge-connected and $z_{\text{in}}(v) = \rho_F(v)$ and $z_{\text{out}}(v) = \delta_F(v)$ hold for every $v \in V$.*

Proof. The first part is clear from the definitions. To see the second part, let z_{in} and z_{out} be two vectors having the required properties. Extend G by a new node s, by $z_{\text{in}}(v)$ new parallel edges from s to v, and by $z_{\text{out}}(v)$ new parallel edges from v to s ($v \in V$). By the hypotheses the extended graph G' satisfies the hypotheses of Theorem 3.4, and therefore the required augmentation exists. □

By now we are in a position to prove Theorems 3.7 and 3.9.

Proof of Theorem 3.7. By Corollary 8.2 and Theorem 3.4 all we have to prove is that there is an integer vector z_{in} in $C(p_{\text{in}})$ for which $f_{\text{in}} \leq z_{\text{in}} \leq g_{\text{in}}$, $z_{\text{in}}(V) = \gamma$, and that there is an integer vector z_{out} in $C(p_{\text{out}})$ for which $f_{\text{out}} \leq z_{\text{out}} \leq g_{\text{out}}$, $z_{\text{out}}(V) = \gamma$. Apply

Proposition 6.9 to $C(p_{\text{in}})$ and to $C(p_{\text{out}})$ (separately) with the choice $\alpha := \beta := \gamma$. The assumption $\gamma \leqq \min(g_{\text{in}}(V), g_{\text{out}}(V))$ implies (6.5). Equation (6.6) follows from (3.6) and (8.1). Equation (6.7) follows from (3.5). □

Proof of Theorem 3.9. The proof is immediate if we observe that the algorithm in question is nothing but two (separate) applications of the greedy algorithm described in Corollary 6.7′ to Q_{in} and Q_{out}. □

9. Max-flow version and algorithmic aspects. This section is offered to make some comments on the complexity of algorithms implied by the proofs. It is certainly not our purpose to describe a detailed algorithm with data structure and precise time-bound. Instead, we briefly indicate the idea of a strongly polynomial algorithm.

Basically, we encountered two types of problems. Problem A consists of finding an appropriate enlargement of a starting graph or digraph using a new node s. Problem B consists of performing algorithmically the splitting operation.

We will consider these algorithms concerning the max-flow version. Let $G = (V, E)$ be a graph or a digraph, and $g : E \to Z_+$ an integer-valued capacity function. Let $r(u, v)$ be an integer-valued demand function so that there is no marginal component in the undirected case and $r \equiv k$ in the directed one. Recall that the max-flow version of the augmentation problem is as follows. Extend G by adding new edges with suitable capacities so that in the enlarged digraph the maximum flow value from every node u to any other node v is at least $r(u, v)$, and so that the sum of capacities of the newly added edges is minimum. (The algorithms below work with the same time complexity if g and r are not necessarily integer-valued.)

By replacing every edge e by $g(e)$ parallel edges, we can see that this max-flow version is theoretically equivalent to its noncapacitated case analyzed in §§ 3 and 5. We do not formulate the corresponding theorems but only mention a corollary of Theorems 5.5 and 3.1.

COROLLARY 9.1. (a) *Let $G = (V, E)$ be an undirected graph, $r(u, v)$ an integer-valued demand-function such that G has no marginal components, and g an integer-valued capacity function on E. There is an optimal solution to the undirected max-flow augmentation problem that is half integral. Furthermore, an optimal integer-valued solution is either optimal among the real-valued augmentations or its total increment is one half bigger than that of a (real-valued) optimal solution.*

(b) *If G is a directed graph and $r(u, v) \equiv k$, then there is an optimal solution to the directed max-flow augmentation problem that is integer-valued.*

Proof. Let γ and γ^* denote the minimum total increment of an integer-valued and a real-valued augmentation, respectively. By Theorem 5.5, $\gamma = \lceil \frac{1}{2} \max \Sigma q(X_i) \rceil$. Let us consider the augmentation problem concerning capacity function $g' := 2g$ and demand function $r' := 2r$. Let q' $(=2q)$ denote the deficiency function and γ' the minimum total increment of an integral augmentation x'. Clearly, $x'/2$ is a fractional solution to the original augmentation problem. Therefore $\gamma^* \leqq \gamma'/2$. On the other hand, by Theorem 5.5 again, we have $\gamma' = \lceil \frac{1}{2} \max \Sigma q'(X_i) \rceil = \frac{1}{2} \max \Sigma q'(X_i) = \max \Sigma q(X_i) \leqq 2\gamma^* \leqq \gamma'$.

Hence $x'/2$ is an optimal solution to the original augmentation problem and $\gamma = \lceil \gamma^* \rceil$ from which part (a) follows.

Part (b) follows directly from Theorem 3.1. □

Gomory and Hu [19] described a very simple solution method to the undirected max-flow augmentation problem when the starting graph G is the empty graph. Their algorithm provides only a half-integer solution. For the same problem, Sridhar and Chandrasekaran [30] described a polynomial time algorithm that finds an integer-valued optimal augmentation. Bland, Goldfarb, and Todd [1] showed how to apply the ellipsoid

method to find in polynomial time an optimal fractional solution to the minimum cost augmentation problem when the starting graph is arbitrary. (Recall that the min-cost integer-valued augmentation problem is NP-complete.)

From an algorithmic point of view, the capacitated augmentation problem is more difficult than the noncapacitated one. Intuitively, the situation is analogous to that of computing a maximum flow from a source node s to a sink node t in the sense that Ford and Fulkerson's augmenting path algorithm [8] runs in polynomial time when every capacity is 1, while the general case needs more sophisticated methods to obtain a (strongly) polynomial time algorithm.

We mention the max flow-min cut (in short, MFMC) problem not only for sake of analogy, but because it will be needed as a subroutine as well. The first strongly polynomial algorithm to compute a flow of maximum value is due to Dinits [3] and to Edmonds and Karp [6]. This algorithm constructs not only a maximum flow, but also a subset S of nodes, $s \in S$ and $t \notin S$, for which $\delta_g(S)$ is minimum.

During the past twenty years a great number of more efficient MFMC algorithms have been developed. For a recent survey, see [18].

Before considering the two algorithmic problems, we make some preparations concerning the general augmentation problem for undirected graphs. A demand function $r(u, v)\,(u, v \in V)$ can be given by $n(n-1)/2$ numbers. Gomory and Hu [19], however, showed that r can be encoded by $O(n)$ data, namely, with the help of a tree $T = (V, F)$, called a *dominant requirement* tree.

THEOREM 9.2 ([19]). *Let $T = (V, F)$ be a maximum cost tree with respect to the cost function r. For any graph G $\lambda_G(u, v) \geqq r(u, v)$ for every pair $\{u, v\}$ of nodes if and only if $\lambda_G(u, v) \geqq r(u, v)$ holds for the edges uv of T.*

For a proof, see [8]. Such a tree can be easily computed by the greedy algorithm, and henceforth we assume that T is available.

Let us turn to Problem A. Suppose that $G' = (V + s, E')$ is a graph or a digraph, and $g : E' \to Z_+$ is an integer-valued function on the edges satisfying

$$\rho'_g(X) \geqq k \text{ for every } \Phi \subset X \subset V, \tag{9.1a}$$

if G' is directed, and

$$d'_g(X) \geqq R(X) \text{ for every } \Phi \subset X \subset V, \tag{9.1b}$$

if G' is undirected, where $R(X)$ was introduced in § 5. We will refer to the special case $R(X) \equiv k$ as the case of uniform demands.

Since we are considering the capacitated case, we may also assume that there are no parallel edges in G'.

Problem A. Let $f_1, f_2, \cdots$ be the set of edges leaving s. Proceeding edge by edge in the given order, decrease the capacity of the current edge as much as possible without violating (9.1).

Problem A can be solved by $n\ (:= |V|)$ subsequent applications of the following subproblem. For a specified edge st, compute the largest value z such that $z \leqq g(st)$ and such that reducing $g(st)$ by z does not destroy (9.1).

For directed graphs (where the demands are uniform), we have $z = \min(z', g(s, t))$, where $z' := \min(m(u, t) - k: u \in V - t)$, where $m(u, t) := \min(\rho'_g(T): T \subseteq V - u, t \in T)$.

Value $m(u, t)$ can be computed as follows. Introduce a new edge su with ∞ capacity. From the MFMC theorem, it follows that $m(u, t)$ is the maximum value of a flow from s to t in this revised network. Hence z can be computed by $n - 1$ MFMC computations, and Problem A, for directed graphs, can be solved by at most n^2 MFMC computations.

For undirected graphs, one has $z = \min(z', g(s,t))$, where $z' := \min(m(u,v) - r(u,v): u, v \in V)$, where $m(u,v) := \min(d'_g(X): X$ separates $\{s,t\}$ and $\{u,v\})$. By Theorem 9.2, $z' = \min(m(u,v) - r(u,v): uv \in F)$.

Value $m(u,v)$ can be computed as follows. First, introduce new edges sv and ut with infinite capacity, and compute the max flow value m_1 from s to t. Second, introduce new edges su and vt with infinite capacity, and compute the max flow value m_2 from s to t. Then $m(u,v) = \min(m_1, m_2)$. Therefore $m(u,v)$ can be computed by two MFMC computations, and hence z can be computed by $2n^2$ MFMC computations.

Let us turn to our second algorithmic problem.

Let $G' = (V + s, E')$ be a directed or undirected graph endowed with a nonnegative integer capacity function g. Let $\{us, sv\}$ be a pair of edges and z an integer with $0 \leqq z \leqq \min(g(us), g(sv))$. We call the following operation a *weighted splitting* of value z. Reduce $g(us)$ and $g(sv)$ by z and increase $g(uv)$ by z.

Problem B. Assume that for directed graphs g satisfies $\rho'_g(s) = \delta'_g(s) > 0$ and

$$\rho'_g(X) \geqq k \text{ for every } \varnothing \subset X \subset V, \tag{9.2a}$$

$$\delta'_g(X) \geqq k \text{ for every } \varnothing \subset X \subset V. \tag{9.2b}$$

For undirected graphs $d_g(s)$ is even, and

$$d'_g(X) \geqq R(X) \text{ for every } \varnothing \subset X \subset V. \tag{9.3}$$

We call a weighted splitting *feasible* if it does not destroy (9.2) or (9.3). We call a sequence of feasible splittings *complete* if, in the final digraph, no edge leaving s has positive (revised) capacity.

The problem consists of computing in strongly polynomial time a complete sequence of feasible splittings. Theorems 3.4 and 5.2 ensure that such a sequence always exists.

Consider first the directed case. For a given pair $\{us, sv\}$ of edges, the biggest value $z(u,v)$ of a feasible splitting is the minimum of $g(us)$, $g(sv)$, m_1, and m_2, where

$$m_1 := \min(\rho'_g(X) - k: \{u,v\} \subseteq X \subset V) \quad \text{and} \tag{9.4a}$$

$$m_2 := \min(\delta'_g(X) - k: \{u,v\} \subseteq X \subset V). \tag{9.4b}$$

m_1 can be computed by $n - 2$ MFMC computations as follows. First, shrink u and v into one node t, then introduce an edge from s to a node $w \in V$ with infinite capacity, and, finally, compute the maximum flow value from s to t. m_1 is the smallest among these values over the possible choices of node $w \in V$. Value m_2 can be computed analogously. We will also need a set A, computed by the MFMC algorithm, where the minimum in (9.4) is attained.

Depending on which one is the smallest value among $g(us)$, $g(sv)$, m_1, and m_2, a weighted splitting-off $\{us, sv\}$ with value $z(u,v)$ either reduces $g(us)$ to 0, reduces $g(sv)$ to 0, creates an in-critical set A containing u and v, or creates an out-critical set A containing u and v. We refer to the first two possibilities as Case 1 and to the second two possibilities as Case 2.

The algorithm to solve Problem B consists of a sequence of splitting steps. One *splitting step* consists of choosing a pair $\{us, sv\}$ with $u \neq v$, $g(us) > 0$, $g(sv) > 0$, computing $z(u,v)$ as indicated above, and performing a weighted splitting-off operation of value $z(u,v)$.

The algorithm stops when there is no more edge sv with positive revised capacity. Since there may be at most n^2 possible pairs to be split off, the algorithm halts after at most n^2 splitting steps when the subsequent pairs are chosen in an arbitrary order.

This bound can be reduced to $4n$ if an appropriate ordering of the pairs is chosen. To this end, we maintain a family $\mathcal{F}$ of disjoint critical sets. At the beginning, $\mathcal{F}$ is empty.

In an intermediate step of the algorithm let the next pair $\{us, sv\}$ be chosen in such a way that $g(us) > 0$, $g(sv) > 0$ and u, v do not belong to the same member of $\mathcal{F}$. Such a pair always exists since otherwise there would be a critical set Y (in $\mathcal{F}$) such that $g(ws) = g(sw) = 0$ for every $w \in V - Y$ and then $X := V - Y$ would violate (9.2). Perform the splitting step on $\{us, sv\}$.

If Case 1 occurs, leave $\mathcal{F}$ unchanged and iterate. Clearly, Case 1 may occur at most $2n$ times.

If Case 2 occurs (and Case 1 does not), let A be a new critical set (found by the MFMC algorithm) containing u and v. Define $\mathcal{F}_1 := \{X \in \mathcal{F}: X - A \text{ critical}\}$, $A' := A \cup \cup(X: X \in \mathcal{F} - \mathcal{F}_1)$, and $\mathcal{F}' := \{X - A: X \in \mathcal{F}_1\} \cup \{A'\}$.

PROPOSITION 9.3. *$\mathcal{F}'$ consists of disjoint critical sets. Furthermore,*

$$|\cup\mathcal{F}'| - |\mathcal{F}'| > |\cup\mathcal{F}| - |\mathcal{F}|, \tag{9.5}$$

where $\cup\mathcal{F}$ stands for $\cup(X: X \in \mathcal{F})$.

Proof. For the first part, we must show that A' is critical. Let $X_1, X_2, \cdots, X_h$ be the elements of $\mathcal{F} - \mathcal{F}_1$. Let $Y_o := A$ and $Y_j := A \cup X_1 \cup X_2 \cup \cdots \cup X_j$ $(j = 1, \cdots, h)$. Clearly, $A' = Y_h$. By induction on j, we prove that Y_j is critical. This is true, by definition, for $j = 0$. Suppose we have already proved for a certain j $(<h)$ that Y_j is critical. Since $X_{j+1} - Y_j = X_{j+1} - A$ is not critical, by Proposition 3.6, $X_{j+1} \cup Y_j = Y_{j+1}$ is critical, as required.

To see (9.5) we distinguish some cases. If $u, v \in A - \cup\mathcal{F}$, then $|\cup\mathcal{F}'| \geqq |\cup\mathcal{F}| + 2$ and $|\mathcal{F}'| \leqq |\mathcal{F}| + 1$, and (9.5) follows.

If $u \in A - \cup\mathcal{F}$, $v \in X_i \in \mathcal{F}$, then $|\cup\mathcal{F}'| \geqq |\cup\mathcal{F}| + 1$. Furthermore, since the revised $g(sv)$ is still positive (as we are not at Case 1), in Proposition 3.6 alternative (ii) cannot hold for A and X_i, and therefore $X_i \notin \mathcal{F}_1$. Hence $|\mathcal{F}'| \leqq |\mathcal{F}|$, and (9.5) follows. The case when $v \in A - \cup\mathcal{F}$, $u \in X_i \in \mathcal{F}$ is analogous.

Finally, assume that $u \in X_i \in \mathcal{F}$, $v \in X_j \in \mathcal{F}$ $(i \neq j)$. Then $|\cup\mathcal{F}'| \geqq |\cup\mathcal{F}|$. By an argument similar to the one used before, we have $X_i \notin \mathcal{F}_1$ and $X_j \notin \mathcal{F}_1$. Hence $|\mathcal{F}'| \leqq |\mathcal{F}| - 1$, and (9.5) follows. □

PROPOSITION 9.4. *The algorithm for solving Problem* B *halts after at most* $4n$ *splitting steps.*

Proof. In Case 1 the capacity of an edge incident to s becomes 0. This can happen at most $2n$ times. By (9.5) Case 2 may occur at most $2n$ times. □

Since one splitting step can be carried out by $2n$ MFMC computations, Problem B can be solved by $8n^2$ MFMC computations. The time complexity of other calculations is inferior to that of n^2 MFMC computations.

Problem A was solved by n^2 MFMC computations. Since there are MFMC algorithms of order $O(n^3)$ (see the survey paper of Goldberg, Tardos, and Tarjan [18]), we can conclude that the overall complexity of the max-flow version of the directed augmentation problem is $O(n^5)$.

Consider now Problem B for undirected graphs. By Theorem 9.2 for a given pair $\{su, sv\}$ of edges the biggest value $z(u, v)$ of a feasible splitting is the minimum of $g(su)$, $g(sv)$, and $\lfloor m/2 \rfloor$, where

$$m := \min\,(m(ww') - r(ww'): ww' \text{ an edge of } T), \tag{9.6}$$

where T is the dominant requirement tree, and $m(ww') := \min\,(d'_g(X): X \text{ separates } w$

and $w', u, v \in X, s \notin X$). Value $m(ww')$ can be computed as follows. Shrink u and v into a node t. Introduce edges sw and tw' of infinite capacity and compute the maximum flow value m_1 between s and t. Likewise, compute the maximum flow value m_2 when the role of w and w' is interchanged. We have $m = \min(m_1, m_2)$.

Therefore $m(ww')$ can be computed by 2 MFMC computations and hence $z(u, v)$ is computed by $O(n)$ MFMC computations. Since there may be n^2 pairs to be split off and there is an MFMC algorithm of complexity $O(n^3)$, the flow version of the undirected augmentation problem can be solved in $O(n^6)$ time. (Note that the number of steps is independent on the demand function r.)

In case of uniform demands, the proof of Theorem 4.5 can be used to reduce the number of splitting steps from n^2 to $O(n)$, similarly to the directed case where Proposition 3.6 was used. Therefore, the overall complexity in this special case reduces to $O(n^5)$. We omit the details.

REFERENCES

[1] R. G. BLAND, D. GOLDFARB, AND M. J. TODD, *The ellipsoid method: a survey*, Oper. Res., 29 (1981), pp. 1039–1091.

[2] G.-R. CAI AND Y.-G. SUN, *The minimum augmentation of any graph to k-edge-connected graph*, Networks, 19, (1989), pp. 151–172.

[3] E. A. DINITS, *Algorithm for solution of a problem of maximum flow in a network with power estimation*, Dokl. Akad. Nauk SSSR, 194 (1970), pp. 754–757. (In Russian.) (English translation by Amer. Math. Soc., Soviet Math. Dokl., 11 (1979), pp. 1277–1289.)

[4] J. EDMONDS, *Submodular functions, matroids, and certain polyhedra*, in Combinatorial Structures and Their Applications, R. Guy, H. Hanani, N. Sauer, and J. Schönheim, eds., Gordon and Breach, New York, 1970, pp. 69–87.

[5] ———, *Matroid intersection*, Ann. of Discrete Math., 4 (1979), pp. 185–204.

[6] J. EDMONDS AND R. M. KARP, *Theoretical improvements in algorithmic efficiency for network flow problems*, J. Assoc. Comput. Mach., 19 (1972), pp. 248–264.

[7] K. P. ESWARAN AND R. E. TARJAN, *Augmentation problems*, SIAM J. Comput., 5 (1976), pp. 653–665.

[8] L. R. FORD AND D. R. FULKERSON, *Flows in Networks*, Princeton University Press, Princeton, NJ, 1962.

[9] H. FRANK AND W. CHOU, *Connectivity considerations in the design of survivable networks*, IEEE Trans. Circuit Theory, CT-17, 1970, pp. 486–490.

[10] A. FRANK, *How to make a digraph strongly connected*, Combinatorica, 1 (1981), pp. 145–153.

[11] ———, *Generalized polymatroids*, in Finite and Infinite sets, A. Hajnal et al., eds., North-Holland, Amsterdam, 1984, pp. 285–294.

[12] A. FRANK, *On a theorem of Mader*, in Proc. of the "Petersen" Graph Conference, Denmark, 1990, Annals of Discrete Mathematics, J. Bang-Jensen, T. Jensen, L. K. Joergensen, B. Toft, and P. D. Vestergaard, eds., to appear.

[13] A. FRANK AND E. TARDOS, *Generalized polymatroids and submodular flows*, Math. Programming, 42 (1988), pp. 489–563.

[14] ———, *An application of submodular flows*, Linear Algebra Appl., 114/115 (1989), pp. 320–348.

[15] S. FUJISHIGE, *Submodular Functions and Optimization*, Ann. Discrete Math., North-Holland, Amsterdam, 47, 1991.

[16] D. R. FULKERSON AND L. S. SHAPLEY, *Minimal k-arc-connected graphs*, Networks, 1 (1971), pp. 91–98.

[17] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, San Francisco, 1979.

[18] A. V. GOLDBERG, R. TARDOS, AND R. E. TARJAN, *Network flow algorithms*, in Paths, Flows, and VLSI Layout, B. Korte, L. Lovász, H-J. Prömel, and A. Schrijver, eds., Springer-Verlag, Berlin, New York, 1990, pp. 101–164.

[19] R. E. GOMORY AND T. C. HU, *Multi-terminal network flows*, SIAM J. Appl. Math., 9 (1961), pp. 551–570.

[20] D. GUSFIELD, *Optimal mixed graph augmentation*, SIAM J. Comput., 16 (1987), pp. 599–612.

[21] F. HARARY, *The maximum connectivity of a graph*, Proc. Nat. Acad. Sci. U.S.A., 48 (1962), pp. 1142–1146.

[22] Y. KAJITANI AND S. UENO, *The minimum augmentation of directed tree to a k-edge-connected directed graph*, Networks, 16 (1986), pp. 181–197.

[23] L. LOVÁSZ, *Conference on Graph Theory*, lecture, Prague, 1974.

[24] ———, *Combinatorial Problems and Exercises*, North–Holland, Amsterdam, 1979.

[25] C. L. LUCCHESI AND D. H. YOUNGER, *A minimax relation for directed graphs*, J. London Math. Soc., 2 (1978), pp. 369–374.

[26] W. MADER, *A reduction method for edge-connectivity in graphs*, Ann. Discrete Math., 3 (1978), pp. 145–164.

[27] ———, *Konstruktion aller n-fach kantenzusammenhangenden Digraphen*, European J. Combin., 3 (1982), pp. 63–67.

[28] D. NAOR, D. GUSFIELD, AND CH. MARTEL, *A fast algorithm for optimally increasing the edge-connectivity*, in Proc. 31st Annual Symposium on Foundations of Computer Sciences, St. Louis, 1990, pp. 698–707.

[29] L. S. SHAPLEY, *Cores of convex games*, Internat. J. Game Theory, 1 (1971), pp. 11–26.

[30] S. SRIDHAR AND R. CHANDRASEKARAN, *Integer solution to synthesis of communication networks*, in Integer Programming and Combinatorial Optimization, R. Kannan and W. Pulleyblank, eds., in Proc. of a conference held at the University of Waterloo, University of Waterloo Press, Waterloo, Ontario, Canada, pp. 467–484.

[31] T. WATANABE AND A. NAKAMURA, *Edge-connectivity augmentation problems*, Comp. System Sci., 35 (1987), pp. 96–144.

[32] ———, *A smallest augmentation to 3-connect a graph*, Discrete Appl. Math., 28 (1990), pp. 183–186.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 54–66, February 1992
© 1992 Society for Industrial and Applied Mathematics
004

COMPUTING EDGE-CONNECTIVITY IN MULTIGRAPHS AND CAPACITATED GRAPHS*

HIROSHI NAGAMOCHI† AND TOSHIHIDE IBARAKI†

Abstract. Given an undirected graph $G = (V, E)$, it is known that its edge-connectivity $\lambda(G)$ can be computed by solving $O(|V|)$ max-flow problems. The best time bounds known for the problem are $O(\lambda(G)|V|^2)$, due to Matula (*28th IEEE Symposium on the Foundations of Computer Science*, 1987, pp. 249–251) if G is simple, and $O(|E|^{3/2}|V|)$, due to Even and Tarjan (*SIAM J. Comput.*, 4 (1975), pp. 507–518) if G is multiple.

An $O(|E| + \min\{\lambda(G)|V|^2, p|V| + |V|^2 \log |V|\})$ time algorithm for computing the edge-connectivity $\lambda(G)$ of a multigraph $G = (V, E)$, where $p(\leqq |E|)$ is the number of pairs of nodes between which G has an edge, is proposed. This algorithm does not use any max-flow algorithm but consists only of $|V|$ times of graph searches and edge contractions. This method is then extended to a capacitated network to compute its minimum cut capacity in $O(|V||E| + |V|^2 \log |V|)$ time.

Key words. undirected multigraphs, edge-connectivity, capacitated networks, maximum flows, minimum cuts, polynomial time algorithms

AMS(MOS) subject classifications. 05C40, 68C25

1. Introduction.

In this paper, a graph $G = (V, E)$ stands for an undirected multigraph that satisfies $|V| \geqq 2$. Note that it may have multiple edges but has no self-loop, unless otherwise specified. Let $\lambda(G)$ denote the edge-connectivity of G. We consider the problem of computing $\lambda(G)$ for a given G. Most of the known algorithms [2], [4], [7], [11] are based on the fact that $\lambda(G)$ can be computed by solving $O(|V|)$ max-flow problems. The best-known bounds are $O(\lambda(G)|V|^2)$ [7] if G is simple and $O(|E|^{3/2}|V|)$ [2] if G is multiple, respectively. It is also known that, for a positive integer k, whether $\lambda(G) \geqq k$ can be tested in $O(k|V|^2)$ time [7] if G is simple and in $O(\min\{k, |E|^{1/2}\}|V||E|)$ time [2] if G is multiple.

Recently, an $O(|E|)$ time algorithm was developed [8] to find a k-edge-connected spanning subgraph $G' = (V, E')$ satisfying $|E'| \leqq k|V|$ for a k-edge-connected multigraph $G = (V, E)$. Only by preprocessing G by this algorithm, the above bound $O(\min\{k, |E|^{1/2}\}|V||E|)$ for testing k-edge-connectivity can be reduced to $O(|E| + \min\{k, (k|V|)^{1/2}\}k|V|^2)$ [8]. This is an improvement since $O(k|V|) \leqq O(|E|)$ can be assumed because $k|V|/2 > |E|$ trivially implies that G is not k-edge-connected.

We present even faster algorithms with time bounds $O(|E| + \min\{k|V|^2, p|V| + |V|^2 \log |V|\})$ for testing whether a multigraph $G = (V, E)$ is k-edge-connected, and $O(|E| + \min\{\lambda(G)|V|^2, p|V| + |V|^2 \log |V|\})$ for computing $\lambda(G)$, where $p(\leqq |E|)$ is the number of pairs of nodes between which G has an edge. Furthermore, this method is extended to a capacitated undirected network $N = (G, c)$, where $G = (V, E)$ is an undirected multigraph and c is an $|E|$-dimensional vector of positive real capacities $c(e)$, $e \in E$. The minimum cut capacity can be computed in $O(|E| + p|V| + |V|^2 \log |V|)$ time.

The organization of this paper is as follows. Necessary definitions are prepared in § 2. In § 3, a partition of edge set E of G based on a decomposition of G into spanning forests is introduced and some useful properties are shown. Based on these properties, a new algorithm for computing the edge-connectivity of an undirected multigraph is pro-

* Received by the editors January 10, 1990; accepted for publication (in revised form) October 26, 1990.

† Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Kyoto 606 Japan.

posed and analyzed in § 4. Its extension to capacitated undirected networks is then described in § 5. Finally, § 6 discusses some implications of the new algorithms.

2. Some definitions. Let $G = (V, E)$ be an undirected multigraph. The set of edges whose end nodes are u and v is denoted by E_{uv}. Throughout this paper, unless confusion arises, an edge $e \in E_{uv}$ is denoted by $e = (u, v)$. A graph $G' = (V', E')$ is called a subgraph of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. It is a spanning subgraph if $V' = V$. A graph G without cycle is called a forest, and a connected forest is called a tree. For a given graph G (possibly not connected), a spanning forest G' is maximal if the subgraph of G' induced by each connected component of G is connected. The graph obtained by removing a subset $F \subseteq E$ from a connected graph $G = (V, E)$ is denoted $G - F$. Such F is called a cut if $G - F$ is disconnected. A cut F is a minimum cut if $|F|$ is minimum among all cuts of G.

The graph obtained from G by contracting nodes x and y in G is denoted $G/\{x, y\}$, in which all self-loops (if edges $e = (x, y)$ exist) resulting from contraction are deleted. For a subset $X \subseteq V$, define $E(X) \equiv \{e \in E \mid$ one end node of e is in X, while the other end node is in $V - X\}$ $(= E(V - X))$. Any $E(X) \neq \phi$ is a cut. In particular, $E(\{x\})$ for $x \in V$ is denoted by $E(x)$. The minimum degree of G, i.e., $\min_{x \in V} |E(x)|$, is denoted by $\delta(G)$. G is called k-edge-connected if $G - F$ is connected for any $F \subseteq E$ with $|F| \leqq k - 1$. In other words,

(2.1) G is k-edge-connected if and only if $|E(X)| \geqq k$ for all $X \subseteq V$ with $X \neq \phi, V$.

The edge-connectivity $\lambda(G)$ of a graph G is defined to be k if G is k-edge-connected but not $(k + 1)$-edge-connected. In other words, a cut F is minimum if and only if $|F| = \lambda(G)$ holds. The local edge-connectivity $\lambda(x, y; G)$ for $x, y \in V$ with $x \neq y$ is defined to be the least number $|F|$ such that x and y are disconnected in $G - F$, where $F \subseteq E$. Clearly,

$$\lambda(G) = \min \{\lambda(x, y; G) \mid x, y \in V, x \neq y\} \tag{2.2}$$

and

$$\lambda(G) \leqq \delta(G) \leqq 2|E|/|V| \tag{2.3}$$

hold. For example, a graph G shown in Fig. 1 has $\delta(G) = 7$ and $\lambda(G) = 6$. See [1] for other basic terminologies such as cycles, simple graphs, and multigraphs.

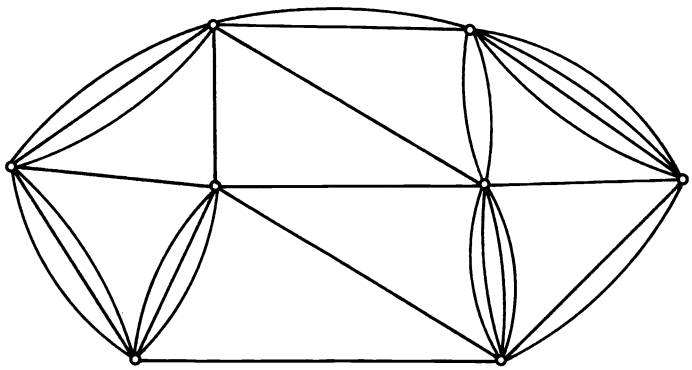


FIG. 1. *A multiple graph G with* $\delta(G) = 7$ *and* $\lambda(G) = 6$.

3. Finding a k-edge-connected spanning subgraph. The following lemma, which was independently found by Nishizeki and Poljak [10], and by Nagamochi and Ibaraki [8, Lemma 2.1], is the starting point of our algorithm. Let $E_1, E_2, \cdots, E_{|E|}$ be a partition of the edge set E of G defined by the property that $H_i = (V, E_i)$ is a maximal spanning forest in $G - E_1 \cup E_2 \cup \cdots \cup E_{i-1}$ for $i = 1, 2, \cdots, |E|$. Here $E_i \cap E_j = \phi$ for $i \neq j$, $\cup_i E_i = E$ and $E_i \cup E_{i+1} \cup \cdots \cup E_{|E|} = \phi$ holds possibly for some i.

LEMMA 3.1. *For the above partition $E_1, E_2, \cdots, E_{|E|}$ of E of a graph $G = (V, E)$, each spanning subgraph $G_i = (V, E_1 \cup E_2 \cup \cdots \cup E_i)$ satisfies that*

$$\lambda(x, y; G_i) \geqq \min\{\lambda(x, y; G), i\} \quad \textit{for all } x, y \in V \quad \textit{with } x \neq y. \tag{3.1}$$

A partition as stated in the above lemma can be obtained by the next algorithm FOREST [8].

```
    Procedure FOREST; {input: G=(V,E), output: E_1,E_2,...,E_|E|}
    begin
1     Label all nodes v ∈ V and all edges e ∈ E "unscanned";
2     r(v):=0  for all v ∈ V;
3     E_1:=E_2:=...:=E_|E|:=φ;
4     while there exist "unscanned" nodes do
        begin
5         Choose an "unscanned" node x ∈ V with the largest r;
6         for each node y adjacent to x by an "unscanned" edge  do
          {y is "unscanned" and every edge in E_xy is "unscanned"}
7           for each "unscanned" edge e ∈ E_xy do
              begin
8               E_r(y)+1:=E_r(y)+1 ∪ {e};
9               r(y):=r(y)+1;
10              Mark e "scanned"
              end;
11        Mark x "scanned"
        end;
    end.
```

To be precise, lines 6 and 7 are slightly different from the original description of FOREST in [8], which has no rule as to how to select an edge among all "unscanned" edges incident to x. Clearly, adding such restrictions as those stated in lines 6 and 7 influences neither the properties obtained from FOREST in [8] nor its running time (since sets E_{uv} for all $u, v \in V$ can be prepared in linear time at the beginning of the algorithm).

LEMMA 3.2 ([8, Thm. 2.1]). *Given a graph $G = (V, E)$, a partition E_i ($i = 1, 2, \cdots, |E|$) of E stated before Lemma 3.1 can be found by* FOREST *in $O(|V| + |E|)$ time, where each E_i satisfies $|E_i| \leqq |V| - 1$.*

As an example, the partition E_i, $i = 1, 2, \cdots$ obtained by applying FOREST to a graph G of Fig. 1 is illustrated in Fig. 2. In this figure, node x_i (edge e_j) represents that it is the ith node (jth edge) scanned by FOREST.

It is shown in [8] (or can be easily proved) that the partition E_i ($i = 1, 2, \cdots, |E|$) obtained by FOREST has the following properties.

LEMMA 3.3. *Let E_i ($i = 1, 2, \cdots, |E|$) be the partition obtained by* FOREST *from $G = (V, E)$, and let $G_i = (V, E_1 \cup E_2 \cup \cdots \cup E_i)$, $i = 1, 2, \cdots, |E|$.*

(1) $\lambda(G_i) \geqq \min\{\lambda(G), i\}$, $i = 1, 2, \cdots, |E|$.

(2) $E_{\delta(G)} \neq \phi$.

(3) *If $E_i \neq \phi$, then any edge $e = (u, v) \in E_i$ satisfies $\lambda(u, v; G) \geqq \lambda(u, v; G_i) \geqq i$.*

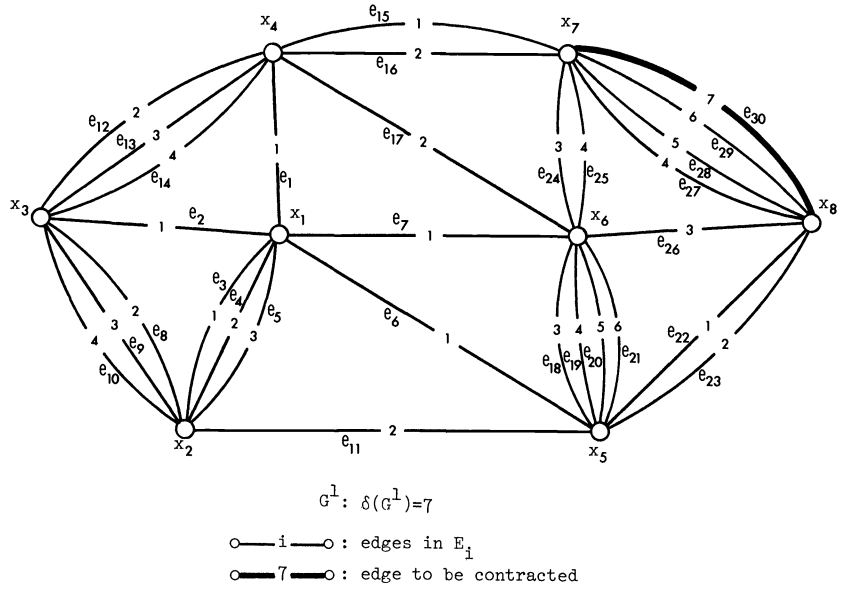


FIG. 2. *Partition* E_i ($i = 1, 2, \cdots, 7$) *of* G *of Fig.* 1 *obtained by* FOREST.

Proof. (1) The proof is immediate from Lemmas 3.1 and 3.2. (2) Let $t \in V$ be the last node scanned by FOREST. Then, it is known [8, Lemma 2.6] that $|E_i \cap E(t)| = 1$, $i = 1, 2, \cdots, |E(t)|$ holds. $|E(t)| \leqq \delta(G)$ by definition. (3) See [8, Lemma 2.4(a)]. □

4. Determining edge-connectivity. For a graph $G = (V, E)$ with $V = \{x, y\}$, $\lambda(G) = \delta(G) = |E| = |E(x)| = |E(y)|$ clearly holds. To find edge-connectivity $\lambda(G)$ of G with $|V| \geqq 3$, the following properties are useful.

LEMMA 4.1. *For a graph* $G = (V, E)$ *with* $|V| \geqq 3$, *let* x, y *be two nodes in* V.

(1) *If* $G/\{x, y\} - F$ *is disconnected, where* F *is a subset of edges in* $G/\{x, y\}$, *then* $G - F$ *is disconnected.*

(2) $\lambda(G) \leqq \lambda(G/\{x, y\})$.

(3) *If* $\lambda(x, y; G) \geqq k$, *then* $\lambda(G) \geqq k$ *if and only if* $\lambda(G/\{x, y\}) \geqq k$.

Proof. Let $G/\{x, y\} = (V', E')$ where $V' = V \cup \{z\} - \{x, y\}$ and z is the node in $G/\{x, y\}$ replacing x and y.

(1) The proof is immediate from that $F \subseteq E' \subseteq E$.

(2) The proof is immediate from (1).

(3) By (2), it suffices only to show the if-part. If $\lambda(G) < k$, then there exists $X \subseteq V$ with $|E(X)| = \lambda(G) < k$ in G. Here, $|X \cap \{x, y\}| \neq 1$, i.e., X does not separate x and y in G, since otherwise $|E(X)| < k$ contradicts $\lambda(x, y; G) \geqq k$. Then, without loss of generality (as X and $V - X$ are symmetric), assume $X \cap \{x, y\} = \phi$. This X satisfies $X \subseteq V'$ and $|E(X)| = \lambda(G) < k$ in $G/\{x, y\}$, a contradiction to $\lambda(G/\{x, y\}) \geqq k$. □

To compute $\lambda(G)$ of $G = (V, E)$, G is reduced in the following to a graph containing only two nodes by applying $|V| - 2$ edge contractions. Let $G^1 = (V^1, E^1)$ denote G, E_i^1 ($i = 1, 2, \cdots, |E^1|$) be the partition of E^1 obtained by FOREST. By Lemma 3.3(2), there is an edge $e^1 = (u^1, v^1) \in E^1_{\delta(G^1)}$. Based on this $\{u^1, v^1\}$, $G^2 = (V^2, E^2)$ is defined to be the graph $G^1/\{u^1, v^1\}$. In analogy with G^1, we define partition E_i^2 ($i = 1, 2, \cdots, |E^2|$) of E^2 and edge $e^2 = (u^2, v^2) \in E^2_{\delta(G^2)}$. Repeating this, G^j, u^j, v^j ($j = 1, 2, \cdots, |V| - 1$) are obtained.

LEMMA 4.2. *Let* $G^j, j = 1, 2, \cdots, |V| - 1$, *be defined as above. Then* $\lambda(G) = k$ *holds, where*

$$k = \min\{\delta(G^1), \delta(G^2), \cdots, \delta(G^{|V|-1})\}. \tag{4.1}$$

Proof. By Lemma 4.1(2), we have

$$\lambda(G^1) \leqq \lambda(G^2) \leqq \cdots \leqq \lambda(G^{|V|-1}). \tag{4.2}$$

Clearly,

$$\lambda(G^j) \leqq \delta(G^j) \quad \text{for } j = 1, 2, \cdots, |V| - 1. \tag{4.3}$$

From (4.1)–(4.3) and $G = G^1$,

$$\lambda(G) \leqq k \tag{4.4}$$

follows. Since $G^{|V|-1}$ contains only two nodes

$$\lambda(G^{|V|-1}) = \delta(G^{|V|-1}) \geqq k. \tag{4.5}$$

By Lemma 3.3(2) and (3),

$$\begin{aligned} \lambda(u^j, v^j; G^j) &\geqq \delta(G^j) \quad (\text{since } (u^j, v^j) \in E^j_{\delta(G^j)}) \\ &\geqq k \quad \text{for } j = 1, 2, \cdots, |V| - 1. \end{aligned} \tag{4.6}$$

From (4.5), (4.6), and Lemma 4.1(3),

$$\lambda(G^j) \geqq k \quad \text{for } j = |V| - 1, |V| - 2, \cdots, 1. \tag{4.7}$$

Therefore $\lambda(G^1) = k$ is concluded from (4.4) and (4.7). □

For the index j attaining $\delta(G^j) = k$ in (4.1), the set F of edges incident to a node with the minimum degree in G^j satisfies that $|F| = k$ and $G^j - F$ is disconnected. By repeated applications of Lemma 4.1(1), $G - F$ is then disconnected, i.e., F is a minimum cut of G. The resulting procedure to compute $\lambda(G)$ and a minimum cut F is described as follows.

Procedure TESTEC; {input: $G=(V,E)$, output: $\lambda(G)$ and a minimum cut $F \subseteq E$ of G}
begin
  $G' := G$; $k := +\infty$;
  while $|V'| \geqq 3$ in $G' = (V', E')$ **do**
    begin
      Compute partition $E_1, E_2, \cdots, E_{|E'|}$ of E' by applying FOREST to G';
      Choose a node $w \in V'$ with $|E'(w)| = \delta(G')$;
      If $\delta(G') < k$ **then** let $F := E'(w)$;
      $k := \min\{k, \delta(G')\}$;
      Let $G' := G'/\{u, v\}$ with an edge $(u,v) \in E_{\delta(G')}$
    end;
  {$|V'| = 2$ now holds, and hence $\lambda(G') = \delta(G') = |E'|$}
  If $\delta(G') < k$ **then** let $F := E'$;
  $k := \min\{k, \delta(G')\}$;
  Conclude that $\lambda(G) = k$ and F is a minimum cut of G
end.

The edge (u, v) used for contraction can be arbitrarily chosen from set $E_{\delta(G')}$. It may be convenient, however, to fix one of the nodes to be the last node t in V' scanned by FOREST, since $E(t) \cap E_{\delta(G')} \neq \phi$ as stated in the proof of Lemma 3.3(2).

The application of TESTEC to G of Fig. 1 is illustrated in Figs. 2 and 3(a)–(f). As a result, $\lambda(G) = 6$ and the minimum cut $F = \{e_6, e_7, e_{11}, e_{15}, e_{16}, e_{17}\}$ are obtained.

THEOREM 4.1. *Let $G = (V, E)$ be a multigraph. $\lambda(G)$ and a minimum cut $F \subseteq E$ of G can be found by* TESTEC *in $O(|E||V|)$ time.*

Proof. Since the validity of TESTEC has already been discussed, we derive the stated time bound. Clearly, the while-loop can iterate $|V| - 2$ times since the number of nodes in G' decreases by one in each iteration. The time to compute the partition $E_i (i = 1, 2, \cdots, |E'|)$ of E' from G' requires $O(|E'|) \leqq O(|E|)$ time by Lemma 3.2. Therefore, the total time is $O(|E||V|)$. □

This time bound can be improved as follows.

COROLLARY 4.1. *Let $G = (V, E)$ be a multigraph.*

(1) *Given $k > 0$, checking $\lambda(G) \geqq k$ can be computed in $O(|E| + k|V|^2)$ time; if $\lambda(G) \leqq k$, then a minimum cut can be constructed in the same time.*

(2) *$\lambda(G)$ and a minimum cut $F \subseteq E$ of G can be computed in $O(|E| + \lambda(G)|V|^2)$ time.*

Proof. Let $E_i (i = 1, 2, \cdots, |E|)$ be the partition of E computed from G by FOREST.

(1) By Lemma 3.3(1), $\lambda(G) \geqq k$ if and only if $\lambda(G_k) \geqq k$, where $G_k = (V, E' = E_1 \cup E_2 \cup \cdots \cup E_k)$. Then, by Lemma 3.2 and Theorem 4.1, G_k is constructed in $O(|E|)$ time, and $\lambda(G_k)$ and a minimum cut F of G_k can be found in $O(|E'||V|) = O(k|V|^2)$ time. Therefore the total time is $O(|E| + k|V|^2)$. We show that, if $\lambda(G_k) < k$ (i.e., $\lambda(G) < k$), a minimum cut F of G_k is a minimum cut of G. Since

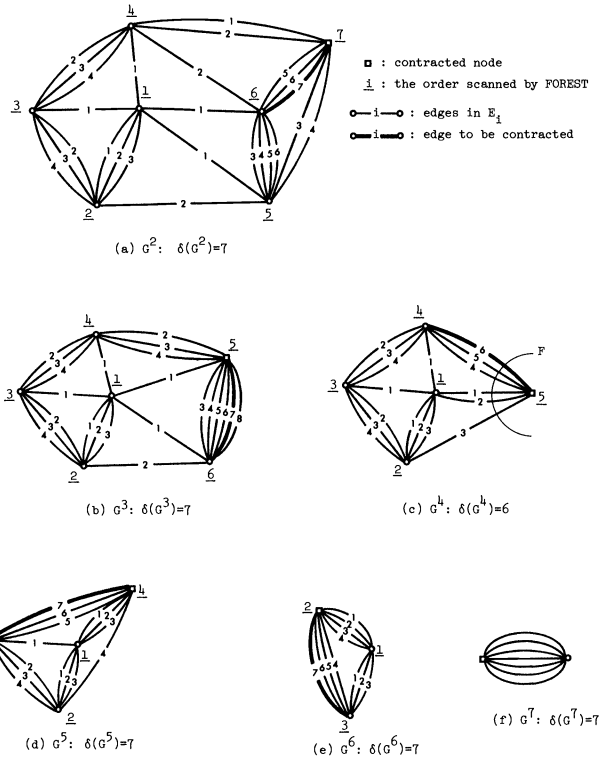


FIG. 3. *Application of* TESTEC *to G of Fig.* 1.

$\lambda(G) \geqq \lambda(G_k) = |F|$, it suffices to prove that $G - F$ is disconnected. From the minimality of F, $G_k - F$ consists of two connected components X and $V - X$. Then,

$$\lambda(x, y; G_k) = |F| \quad \text{for any } x \in X \quad \text{and} \quad y \in V - X. \tag{4.8}$$

If $G - F$ is connected, $\lambda(u, v; G) > |F|$ for some $u \in X$ and $v \in V - X$. However, from (3.1) and $k > \lambda(G_k) = |F|$, this means $\lambda(u, v; G_k) \geqq \min\{\lambda(u, v; G), k\} > |F|$, contradicting (4.8). Therefore $G - F$ is disconnected and $\lambda(G) = |F|$.

(2) Using the algorithm of (1), we check whether $\lambda(G) \geqq k$ for each $k = 2, 2^2, 2^3, \cdots$ to find the integer i satisfying $2^i \leqq \lambda(G) < 2^{i+1}$. Note that, from (1), a minimum cut F and $\lambda(G)(= |F|)$ are obtained after checking such $k = 2^{i+1}$. The total time is

$$O(|E|) + O(2|V|^2) + O(2^2|V|^2) + \cdots + O(2^{i+1}|V|^2)$$
$$= O(|E|) + O(2^{i+2}|V|^2) = O(|E| + \lambda(G)|V|^2). \qquad \square$$

These time bounds will be further improved in the next section (Corollary 5.1).

5. Capacitated networks. In this section, we consider the problem of finding the minimum capacity cut in a capacitated undirected network $N = (G, c)$ with $G = (V, E)$, where G is an undirected multigraph (without self-loops), and c is an $|E|$-dimensional vector of positive real numbers $c(e)$, $e \in E$. By letting $f(x, y; N)$ and $c(x, y; N)$ denote the maximum flow value between x and y and the value of a minimum capacity cut separating x and y, respectively,

$$f(x, y; N) = c(x, y; N) \quad \text{for } x, y \in V \quad \text{with } x \neq y$$

follow from the max-flow min-cut theorem.

Clearly, the value of a minimum capacity cut

$$c(N) \equiv \min\left\{ \sum_{e \in F} c(e) \mid G - F \text{ is disconnected} \right\} = \min\{c(x, y; N) \mid x, y \in V, x \neq y\}$$

is equal to

$$f(N) \equiv \min\{f(x, y; N) \mid x, y \in V, x \neq y\}.$$

Gomory and Hu [6] showed that all values $f(x, y; N)$ $(x, y \in V)$ can be computed by solving $|V| - 1$ max-flow problems. Based on this, $c(N) = f(N)$ can be obtained in $O(|V| T(|V|, |E|))$ time, where $T(|V|, |E|)$ is the time required to find the max-flow in a network with $|V|$ nodes and $|E|$ edges. The best time bound for $T(|V|, |E|)$ known to date is

$$O(|V||E| \log(|V|^2/|E|)) \text{ [5]}.$$

Therefore

$$O(|V| T(|V|, |E|)) = O(|V|^2 |E| \log(|V|^2/|E|)).$$

In the following, we develop a faster $O(|V||E| + |V|^2 \log |V|)$ time algorithm for computing $c(N)$.

First, consider a network with an integer capacity vector $c = (c(e), e \in E)$. Clearly, such a network $N = (G = (V, E), c)$ can be equivalently represented by a multigraph $G(N) = (V, E')$ obtained by replacing each edge $e \in E$ with $c(e)$ parallel multiple edges. Therefore $c(N)$ can be determined by applying TESTEC to $G(N)$. However, this straightforward implementation requires $O(|E'||V|) = O((\sum c(e))|V|)$ time by Theorem 4.1, which is not polynomial in $\sum \log c(e)$.

To remove $c(e)$ from the running time, FOREST is modified so that a capacitated network N can be directly handled. Instead of explicitly retaining sets E_i ($i = 1, 2, \cdots, |E'|$) in $G(N)$, a positive number $q(e)$ is assigned to each "scanned" edge $e \in E$. This number $q(e)$ is used to represent that the $c(e)$ multiple edges in $G(N)$ corresponding to e, respectively, belong to the following $c(e)$ sets of edges in the obtained partition:

$$E_{q(e)-c(e)+1}, E_{q(e)-c(e)+2}, \cdots, E_{q(e)}.$$

The labels r given to all nodes play a similar role as in the previous FOREST. The way of updating r is also modified to make the computation of the above q possible. Now we describe the modified FOREST.

```
    Procedure CAPFOREST; {input: N=(G=(V,E),c), output: f(e),e ∈ E}
    begin
 1    Label all nodes v ∈ V and all edges e ∈ E "unscanned";
 2    r(v):=0   for all v ∈ V;
 3    q(e):=0   for all e ∈ E;
 4    while there exist "unscanned" nodes do
        begin
 5        Choose an "unscanned" node x ∈ V with the largest r;
 6        for each node y adjacent to x by an "unscanned" edge do
          {y is "unscanned" and every edge in E_xy is "unscanned"}
 7          for each "unscanned" edge e ∈ E_xy do
              begin
 8              q(e):=r(y)+c(e);
 9              r(y):=r(y)+c(e);
10              Mark e "scanned"
              end;
11        Mark x "scanned"
        end;
    end.
```

We now show that CAPFOREST can be executed in $O(|E| + |V| \log |V|)$ time. For this, maintain the set of unscanned nodes X as a Fibonacci heap [3], where each item $x \in X$ has value $r(x)$. It is easy to see that the time to carry out CAPFOREST is dominated by the time to maintain the heap, which requires the following operations:

(1) INSERT(x): Insert a new item x with $r(x) = 0$ into the heap (line 2).
(2) FINDMAX: Return an item of maximum value in the heap (line 5).
(3) DELETEMAX: Delete an item of maximum value in the heap (line 10).
(4) INCREASE(x, Δ): Increase the value of item x in the heap by

$$\Delta(=c(e)) \text{ (line 9)}.$$

Since at most $|V|$ operations of INSERT, FINDMAX, and DELETEMAX, and $|E|$ operations of INCREASE are performed in FOREST, the total time required for these is $O(|E| + |V| \log |V|)$ [3, Thm. 1].

The correctness of CAPFOREST can be seen as follows. For an integer-capacitated network $N = (G, c)$, let $q(e)$, $e \in E$ be obtained by CAPFOREST by scanning the nodes in V in the order of $x_1, x_2, \cdots, x_{|V|}$. Then define the multigraph $G(N) = (V, E')$ by creating set E'_{uv} of edges joining u and v, for each $e = (u, v) \in E$, such that

$$E'_{uv} = \{e^1, e^2, \cdots, e^{c(e)}\}.$$

Based on the above $q(e)$, define a partition E'_i ($i = 1, 2, \cdots, |E'|$) of E' by

$$e^1 \in E'_{q(e)-c(e)+1}, \qquad e^2 \in E'_{q(e)-c(e)+2}, \cdots, e^{c(e)} \in E'_{q(e)},$$

for each E'_{uv} corresponding to $e = (u, v) \in E$. Then it is not difficult to see that this E'_i ($i = 1, 2, \cdots, |E'|$) is the partition obtained by FOREST when it is directly applied to $G(N)$ (instead of G) under the same node order $x_1, x_2, \cdots, x_{|V|}$. In this sense, CAPFOREST correctly computes a required partition of E'.

Now we remove the integrality condition on $c(e)$, and consider that c is a positive real vector. It is important to see that CAPFOREST still works under this condition, although the above interpretation of E' is no longer valid. The previous Lemma 3.3(2) and (3) can, however, be generalized under this new setting. For a network $N = (G = (V, E), c)$, let

$$\delta(N) \equiv \min \left\{ \sum_{e \in E(v)} c(e) \mid v \in V \right\}.$$

LEMMA 5.1. *For a network $N = (G, c)$ with a multigraph $G = (V, E)$ and positive real-valued capacities $c(e)$, $e \in E$, let $q(e)$, $e \in E$ be obtained by* CAPFOREST.

(5.1) (1) *There exists an edge $e \in E$ with $q(e) \geqq \delta(N)$.*

(5.2) (2) $f(u, v; N) \geqq q(e)$ *for any* $e = (u, v) \in E$.

Proof. (1) It suffices to show that

$$\max \{ q(e) \mid e \in E(t) \} = \sum_{e \in E(t)} c(e),$$

where $t \in V$ is the last node scanned by CAPFOREST. Since all edges in $E(t)$ have been scanned when CAPFOREST scans the last node t, all $c(e)$, $e \in E(t)$ have been added to $r(t)$. That is, the last edge e' in $E(t)$ scanned by CAPFOREST satisfies $q(e') = \sum_{e \in E(t)} c(e)$.

(2) Without loss of generality, G is assumed to be connected (otherwise the argument is applied to each component). In the case of an integer vector c, (5.2) follows from Lemma 3.3(3) and the observation after the description of CAPFOREST. Let CAPFOREST scan the nodes in V in the order of $x_1, x_2, \cdots, x_{|V|}$.

Before considering the case of a real vector c, assume that c is a rational vector. Starting with x_1, apply CAPFOREST to the network $\tilde{N} = (G, \tilde{c})$ with integer capacities $\tilde{c}(e) = M \cdot c(e)$, $e \in E$, where M is a common denominator of $c(e)$, $e \in E$. Clearly, CAPFOREST can scan the nodes of $\tilde{N}$ in the previous order $x_1, x_2, \cdots, x_{|V|}$ because labels $\tilde{q}$ and $\tilde{r}$ are now updated by $\tilde{q}(e) := \tilde{r}(y) + \tilde{c}(e)$ (i.e., $M \cdot q(e) := M \cdot r(y) + M \cdot c(e)$) and by $\tilde{r}(y) := \tilde{r}(y) + \tilde{c}(e)$ (i.e., $M \cdot r(y) := M \cdot r(y) + M \cdot c(e)$) in lines 8 and 9, respectively. Therefore upon completion of CAPFOREST, $\tilde{q}(e) = M \cdot q(e)$ holds for all $e \in E$. By the integrality of $\tilde{c}$, $f(u, v; \tilde{N}) \geqq \tilde{q}(e)$ holds for every $e = (u, v) \in E$. From this and $f(u, v; \tilde{N}) = M \cdot f(u, v; N)$ ($u, v \in V$), relation (5.2) is proved for N.

Finally, consider the case in which c is real valued. Assume that (5.2) does not hold, i.e.,

$$\text{some edge } e^* = (u^*, v^*) \quad \text{satisfies } f(u^*, v^*; N) < q(e^*).$$

That is, by max-flow min-cut theorem, there exists a cut $F \subseteq E$ separating u^* and v^* such that

(5.3) $$\Delta = q(e^*) - \sum_{e \in F} c(e) > 0.$$

To prove the lemma, we show that, for $\alpha = \Delta/|E|(>0)$, there exists a rational vector c' such that

$$c(e) < c'(e) < c(e) + \alpha, \quad e \in E, \tag{5.4}$$

and the nodes in the resulting network $N' = (G, c')$ can be scanned by

(5.5) CAPFOREST in the same node order $x_1, x_2, \cdots, x_{|V|}$ as $N = (G, c)$.

The existence of such c' will be proved by Lemma A in the Appendix. Once such a vector c' is found, we can easily derive a contradiction as follows. By (5.4) and (5.5),

$$q(e) < q'(e), e \in E \tag{5.6}$$

holds, where q' is computed by CAPFOREST applied to N'. Moreover, by (5.4),

$$\sum_{e \in F} c'(e) < \sum_{e \in F} c(e) + |F|\Delta/|E| \leqq q(e^*) < q'(e^*) \quad \text{(by (5.3) and (5.6))}.$$

This implies $f'(u^*, v^*; N') < q'(e^*)$ by $f'(u^*, v^*; N') \leqq \sum_{e \in F} c'(e)$, contradicting that q' satisfies (5.2) by rationality of c'. □

Using Lemma 5.1 in place of Lemma 3.3(2) and 3.3(3), Lemma 4.1(2) and 4.1(3) and Lemma 4.2 can also be directly extended to a network with positive real-valued capacities, as stated in the following.

LEMMA 5.2. *Let $N = (G = (V, E), c)$ be a network such that $|V| \geqq 3$ and c is a real-valued capacity vector. For two nodes $x, y \in V$, denote $N' = (G/\{x, y\}, c)$. Then the following properties hold.*

(1) $c(N) \leqq c(N')$.

(2) *If $c(x, y; N) \geqq k$, then $c(N) \geqq k$ if and only if $c(N') \geqq k$.*

LEMMA 5.3. *For a network $N = (G = (V, E), c)$ such that $|V| \geqq 3$ and c is a real-valued capacity vector, let $N^1 = N$ and N^j $(j = 2, \cdots, |V| - 1)$ be the network obtained by contracting an edge $e^{j-1} = (u^{j-1}, v^{j-1})$ that satisfies $f(u^{j-1}, v^{j-1}; N^{j-1}) \geqq \delta(N^{j-1})$ in N^{j-1}. Then*

$$c(N) = \min\{\delta(N^1), \delta(N^2), \cdots, \delta(N^{|V|-1})\}.$$

The following algorithm MINCUT obtained by modifying TESTEC now computes $c(N)$ for a real-valued capacity vector c.

Procedure MINCUT; {input: $N=(G=(V,E),c)$, output: $c(N)$ and a minimum cut F}
begin
  $N':=N$; $k:=+\infty$;
  while $|V'| \geqq 3$ in $N'=(G'=(V',E'),c')$ **do**
    begin
      Compute $q(e), e \in E'$ by applying CAPFOREST to N';
      Choose a node $w \in V'$ with $\sum_{e \in E'(w)} c(e) = \delta(N')$;
      If $\delta(N') < k$ **then** let $F := E'(w)$;
      $k := \min\{k, \delta(N')\}$;
      Find an $e=(u,v) \in E'$ such that $q(e) \geqq \delta(N')$ and let $G' := G'/\{u,v\}$ and $N' := (G', c')$
    end;
  {$|V'| = 2$ holds, and hence $c(N') = \delta(N')$}
  If $\delta(N') < k$ **then** $F := E'$;
  $k := \min\{k, \delta(N')\}$;
  Conclude that $c(N) = k$ and F is a minimum cut of N
end.

The validity of MINCUT follows from Lemmas 5.1–5.3 in an analogous manner to the case of TESTEC in § 4. Before resorting to MINCUT, we modify graph $G = (V, E)$ into the simple graph $G' = (V, E')$ by replacing each set E_{uv} of multiple edges with a simple edge $e' = (u, v)$ with

$$c(e') = \sum_{e \in E_{uv}} c(e).$$

Then $|E'| = p$, where p is the number of pairs of nodes between which G has an edge. This computation requires $O(|E|)$ time. Then apply MINCUT to G'. Its running time is dominated by

$$\begin{aligned} |V| \times (\text{time to execute CAPFOREST}) &= |V|(p + |V| \log |V|) \\ &= p|V| + |V|^2 \log |V|. \end{aligned}$$

From these, we obtain the next theorem.

THEOREM 5.1. *Let $N = (G, c)$ be a network such that c is a real-valued capacity vector. A minimum cut F of N and its value $c(N)$ can be computed by* MINCUT *in $O(|E| + p|V| + |V|^2 \log |V|)$ time, where p is the number of pairs of nodes between which G has an edge. If G is a simple graph, the time bound becomes $O(|E||V| + |V|^2 \log |V|)$.*

The previous time bound in Corollary 4.1 to determine $\lambda(G)$ of a multigraph $G = (V, E)$ can be slightly improved by making use of Theorem 5.1.

COROLLARY 5.1. *The $\lambda(G)$ of a multigraph $G = (V, E)$ can be determined in $O(|E| + \min\{\lambda(G)|V|^2, p|V| + |V|^2 \log |V|\})$ time, where p is the number of pairs of nodes between which G has an edge.*

Proof. Let $N(G) = (G', c)$ denote the network such that $G' = (V, E')$ is the simple graph obtained from G by replacing each set E_{uv} with a single capacitated edge $e = (u, v)$ with $c(e) = |E_{uv}|$. Then $|E'| = p$. Let j be the integer such that

$$2^j \leqq p/|V| + \log |V| < 2^{j+1}. \tag{5.7}$$

In a manner similar to the proof of Corollary 4.1, we test $\lambda(G) \leqq k$ by using TESTEC for $k = 2, 2^2, 2^3, \cdots, 2^{j+1}$. If $2^i \leqq \lambda(G) < 2^{i+1}$ is found out for some $i \leqq j$, then terminate. The time in this case is

$$\begin{aligned} &O(|E| + 2|V|^2 + 2^2|V|^2 + \cdots + 2^{i+1}|V|^2) \\ &\quad = O(|E| + \lambda(G)|V|^2) (\leqq O(|E| + p|V| + |V|^2 \log |V|)) \end{aligned}$$

by (5.7). Otherwise (i.e., $\lambda(G) \geqq 2^{j+1}$), we apply MINCUT to network $N(G)$. The time in this case is

$$\begin{aligned} &O(|E| + 2|V|^2 + 2^2|V|^2 + \cdots + 2^{j+1}|V|^2) + O(|E| + p|V| + |V|^2 \log |V|) \\ &\quad = O(|E| + p|V| + |V|^2 \log |V|) (\leqq O(|E| + \lambda(G)|V|^2)) \end{aligned}$$

by (5.7). □

6. Conclusion. In this paper, we proposed an $O(|E| + \min\{\lambda(G)|V|^2, p|V| + |V|^2 \log |V|\})$ time algorithm for determining connectivity $\lambda(G)$ of a multigraph $G = (V, E)$, where p is the number of pairs of nodes having an edge between them. This algorithm differs from the previous one [2], and requires no max-flow algorithm. Based on this method, we then showed that the minimum cut in a capacitated network can be found in $O(|E| + p|V| + |V|^2 \log |V|)$ time.

Finally, we emphasize that, as our algorithm is quite fundamental, it can be modified to solve other related problems. For example, we cite here an algorithm to

count the number of minimum cuts α in a given multigraph G [9], which runs in $O(|E| + \lambda(G)|V|^2 + \alpha\lambda(G)|V|)$ time.

Appendix. LEMMA A. *For any positive real number $\alpha > 0$, there exists a rational capacity vector c' satisfying* (5.4) *and* (5.5).

Proof. Define

$$E^-(x_i) \equiv \bigcup_{j=1}^{i-1} E_{x_j,x_i} \quad \text{for } i = 2, 3, \cdots, |V|.$$

Note that $E^-(x_2) \cup E^-(x_3) \cup \cdots \cup E^-(x_{|V|}) = E$ and $|E^-(x_i)| \geqq 1$, for all i by connectedness of G. After x_{i-1} is scanned in N, CAPFOREST chooses a next node x_i such that

$$r(x_i) \geqq r(x_j), \quad j = i, \quad i+1, \cdots, |V|. \tag{A.1}$$

Since

$$\text{edges in } E^-(x_i) \text{ are all scanned,} \tag{A.2}$$

the labels $r(x_i)$ generated by CAPFOREST satisfy

$$r(x_i) = \sum_{e \in E^-(x_i)} c(e).$$

Choose a rational capacity vector c' such that

$$\begin{aligned} c(e) + \theta(|V| - i)/|E^-(x_i)| \leqq c'(e) < c(e) + \theta(|V| - i + 1)/|E^-(x_i)| \\ \text{for each } e \in E^-(x_i), \qquad i = 2, 3, \cdots, |V|, \end{aligned} \tag{A.3}$$

where θ is a sufficiently small positive real number so that (5.4) holds. Starting with x_1, we show by induction on x_i that CAPFOREST applied to $N' = (G, c')$ can scan V in the same order $x_1, x_2, \cdots, x_{|V|}$ as CAPFOREST applied to $N = (G, c)$. Let r' denote the labels for N'. For x_1, this is obvious. After x_{i-1} is scanned,

$$r'(x_i) \geqq r(x_i) + \theta(|V| - i) \text{ (by (A.2) and (A.3))} \tag{A.4}$$

and

$$r'(x_j) < r(x_j) + \theta(|V| - j + 1), \quad j = i+1, \quad i+2, \cdots, |V| \text{ (by (A.3))} \tag{A.5}$$

hold, where $r(x_h)$, $h = i, i+1, \cdots, |V|$ are the labels of N obtained by CAPFOREST after x_{i-1}. Then, for any x_j $(j = i+1, i+2, \cdots, |V|)$,

$$\begin{aligned} r'(x_j) < r(x_j) + \theta(|V| - j + 1) \leqq r(x_j) + \theta(|V| - i) \leqq r(x_i) + \theta(|V| - i) \text{ (by (A.1))} \\ \leqq r'(x_i) \text{ (by (A.4))}. \end{aligned}$$

This implies that CAPFOREST applied to N' can choose x_i after x_{i-1}. Therefore all nodes in N' can be scanned in the order of $x_1, x_2, \cdots, x_{|V|}$. □

Acknowledgment. We wish to thank Professor Takao Nishizeki of Tohoku University for his valuable discussion as well as the information in reference [10] and the anonymous referees for their helpful comments. The authors were partially supported by a Scientific Grant-in-Aid from the Ministry of Education, Science, and Culture of Japan.

REFERENCES

[1] C. BERGE, *Graphs and Hypergraphs*, North-Holland, Amsterdam, 1973.

[2] S. EVEN AND R. E. TARJAN, *Network flow and testing graph connectivity*, SIAM J. Comput., 4 (1975), pp. 507–518.

[3] M. L. FREDMAN AND R. E. TARJAN, *Fibonacci heaps and their uses in improved optimization algorithms*, Proc. 25th Symposium on the Foundations of Computer Science, Singer Island, FL, 1984, pp. 338–346.

[4] Z. GALIL, *Finding the vertex connectivity of graphs*, SIAM J. Comput., 9 (1980), pp. 197–199.

[5] A. V. GOLDBERG AND R. E. TARJAN, *A new approach to the maximum flow problem*, Proc. 18th ACM Symposium on the Theory of Computing, Berkeley, CA, 1986, pp. 136–146.

[6] R. E. GOMORY AND T. C. HU, *Multi-terminal network flows*, J. SIAM, 9 (1961), pp. 551–570.

[7] D. W. MATULA, *Determining edge connectivity in $O(nm)$*, Proc. 28th IEEE Symposium on the Foundations of Computer Science, Los Angeles, CA, 1987, pp. 249–251.

[8] H. NAGAMOCHI AND T. IBARAKI, *Linear time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph*, Algorithmica, to appear.

[9] ———, *Counting the number of minimum cuts in undirected multigraphs*, IEEE Trans. on Reliability, to appear.

[10] T. NISHIZEKI AND S. POLJAK, *Highly connected factors with a small number of edges*, preprint, 1989.

[11] C. P. SCHNORR, *Bottlenecks and edge connectivity in unsymmetrical networks*, SIAM J. Comput., 8 (1979), pp. 265–274.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 67–73, February 1992

© 1992 Society for Industrial and Applied Mathematics
005

A FORWARD LOWER RESTRICTED ORDERING ALGORITHM FOR DIGRAPHS*

D. D. OLESKY† AND T. A. SLATER‡

Abstract. The concept of forward lower restricted (FLR) orderings of digraphs arises naturally from the study of inherited entries in LU factorizations of matrices. Polynomial time algorithms are presented for deciding if a given ordered digraph is FLR ordered and for finding an FLR ordering of an arbitrary digraph. The latter algorithm also detects that a digraph is not FLR orderable. In addition, characterizations of FLR-ordered trees and maximal FLR-ordered digraphs are given. All of the results of this paper extend to inheritance of entries in the matrix L of the LU factorization and to inheritance in UL factorizations.

Key words. inheritance, directed graphs, LU factorization, trees

AMS(MOS) subject classification. 05C50

1. Introduction. In [5] the term *inheritance* of matrix entries was introduced, generalizing the well-known concept of the preservation of zero entries found in sparse matrix analysis (see, e.g., [2], [3], or [7]). Specifically, inheritance refers to the situation in which one or more entries of the matrices of the LU factorization of a given matrix A are, for purely combinatorial reasons, identical to the corresponding entries of A (see § 2.2). Conditions on the digraph of the matrix A have been determined, which characterize the circumstances under which individual entries (local inheritance) or the entire strict upper triangular part (global inheritance) are preserved. Global inheritance for the strict upper triangular part is characterized by the digraph of the matrix being forward lower restricted (FLR) ordered. This paper is primarily concerned with developing algorithms to solve two problems concerning FLR-ordered digraphs. The first problem is to recognize when an ordered digraph is FLR ordered. The second problem is to find a numbering, if it exists, for the nodes of a digraph such that the resulting ordered digraph is FLR ordered.

2. Definitions and terminology.

2.1. Digraphs, graphs, and orderings. A *digraph* (*di*rected *graph*) is an ordered pair $G = (V, E)$, where V is a finite set (the nodes), and E (the edges) is a set of ordered pairs of elements of V. The cardinalities of V and E are denoted by n and e, respectively, and we assume that G is weakly connected so that $e \geqq n - 1$. If $u, v \in V$, v is *reachable* from u if there is a path of length $\geqq 1$ from u to v in G. Given $W \subset V$ and $u, v \in V - W$, we say that v is *reachable from u through W* if there is a path P: $u \to q_1 \to q_2 \to \cdots \to q_{t-1} \to q_t \to v$ from u to v in G such that $t \geqq 1$ and $\{q_1, q_2, \cdots, q_t\} \subseteq W$. A *graph* (or undirected graph) is a digraph, $G = (V, E)$, such that $u \to v \in E$ if and only if $v \to u \in E$.

An *ordered digraph* is a triple, $G_\alpha = (V, E, \alpha)$, where $G = (V, E)$ is a digraph and $\alpha : \{1, 2, \cdots, n\} \to V$ is a bijection. The bijection α is called an *ordering* on G. A *forward* [*backward*] edge in G_α is an edge $u \to v$ such that $\alpha^{-1}(u) < \alpha^{-1}(v)$ [$\alpha^{-1}(u) > \alpha^{-1}(v)$]. A path in an ordered digraph is said to be a *forward path* [*backward path*] if all edges in the path are forward [backward].

* Received by the editors November 16, 1989; accepted for publication September 26, 1990. The work of the first author was partially supported by Natural Sciences and Engineering Research Council of Canada grant A-8214 and the University of Victoria President's Committee on Faculty Research and Travel.

† Department of Computer Science, University of Victoria, Victoria, British Columbia V8W 3P6, Canada.

‡ Computing and Systems Services, University of Victoria, Victoria, British Columbia V8W 3P4, Canada.

An ordering α is said to be a *topological sort* on G if $\alpha^{-1}(u) < \alpha^{-1}(v)$ for every edge $u \to v \in E$. A topological sort exists for a digraph if and only if it is acyclic, and can be computed in time $O(e)$. An ordering α is said to be a *minimum degree ordering* (and G_α is said to be *minimum degree ordered*) if a node of minimum degree in G is numbered 1, and then (for $k = 1, 2, \cdots, n-1$) a node of minimum degree in $G - \{\alpha(1), \alpha(2), \cdots, \alpha(k)\}$ is numbered $k+1$. If a graph G is a tree, then α is said to be an *invariant ordering* (and G_α is said to be *invariantly ordered*) if for each $v \in V$ there is at most one $u \in V$ with $\alpha^{-1}(u) > \alpha^{-1}(v)$ such that u is adjacent to v. Usually only one ordering at a time is considered for a digraph, and when there is no ambiguity we abbreviate G_α to G.

For other graph-theoretic terms used, see [1] or [4].

2.2. Inheritance in LU factorizations. Let A be a matrix of order n. The *digraph of* A is defined by $G(A) = (V, E)$ where $V = \{1, 2, \cdots, n\}$ and $i \to j \in E$ if and only if $a_{ij} \neq 0$.

A matrix $A = (a_{ij})$ has a *left unit* LU *factorization* if $A = LU$, where U is upper triangular and L is lower triangular with all diagonal entries equal to 1. If A has a unique left unit LU factorization, then an entry u_{ij} of U is said to be *inherited* from A if, for purely combinatorial reasons, $u_{ij} = a_{ij}$ (see [5]). If G denotes an arbitrary digraph on n nodes, let A_G denote the set of all matrices A of order n such that $G(A)$ is a (partial) subdigraph of G and A has a *unique* left unit LU factorization. The following characterization of *global inheritance* is given in [5].

THEOREM 1. *Let G be a digraph on the node set $V = \{1, 2, \cdots, n\}$. Then for all $A \in A_G$ and for all pairs i, j such that $1 \leqq i < j \leqq n$, $u_{ij} = a_{ij}$ in the left unit* LU *factorization of A if and only if for all $1 \leqq i < j \leqq n$, node j is not reachable from node i in G through $\{1, 2, \cdots, i-1\}$.*

A digraph G that satisfies the condition of Theorem 1 is said to be *forward lower restricted* (FLR) *ordered*; i.e., for all $1 \leqq i < j \leqq n$, there does not exist a path in G of length $\geqq 2$ from i to j through $\{1, 2, \cdots, i-1\}$. We extend this definition to arbitrary orderings and arbitrary node sets by defining an ordering α on a digraph $G = (V, E)$ to be an FLR ordering if for every pair u, v of nodes in V such that $\alpha^{-1}(u) < \alpha^{-1}(v)$, v is not reachable from u in G through $\{\alpha(1), \alpha(2), \cdots, \alpha(\alpha^{-1}(u) - 1)\}$. An FLR-ordered digraph $G_\alpha = (V, E, \alpha)$ is said to be *maximal forward lower restricted* (MFLR) *ordered* if the digraph obtained by including in E any one additional edge $i \to j$ $(i \neq j)$ is not FLR ordered (with respect to α).

Example 1. Consider the ordered digraph $G_\alpha = (V, E, \alpha)$ in Fig. 1, where $V = \{a, b, c, d\}$ and $\alpha : \{1, 2, 3, 4\} \to V$. G_α is not FLR ordered because of the paths $2 \to 1 \to 3$ and $3 \to 1 \to 2 \to 4$. However, with $\beta : \{3, 4, 1, 2\} \to V$, the ordered digraph $G_\beta = (V, E, \beta)$ is FLR ordered.

Example 2. Consider the digraph $G = (V, E)$ in Fig. 2, where $V = \{a, b, c, d, e\}$. It can be shown that there exists no ordering α such that $G_\alpha = (V, E, \alpha)$ is FLR ordered. This can be seen, for example, by considering the two cycles of length 5, $a \to b \to c \to d \to e \to a$ and $a \to b \to e \to c \to d \to a$, and the corresponding implied relationships between the nodes a, c, and e.

We note that the presence or absence of loops (i.e., cycles of length one) is irrelevant to whether a digraph is FLR orderable. Therefore, it will henceforth be assumed that all digraphs are loop-free.

In this paper we consider the following two problems:

(i) Find an efficient algorithm to decide whether a given ordering on an arbitrary digraph is an FLR ordering; and

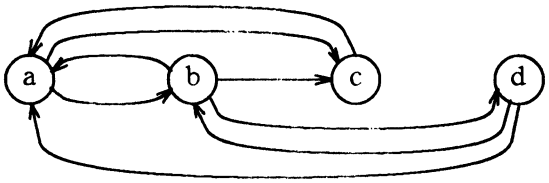


FIG. 1

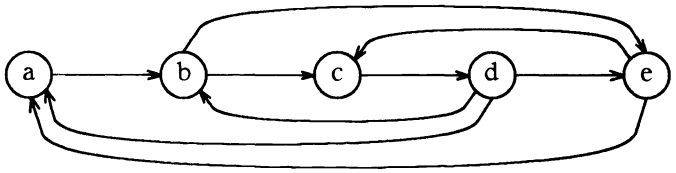


FIG. 2

(ii) Find an efficient algorithm to compute an FLR ordering on an arbitrary digraph when one exists, or determine that such an ordering does not exist.

The *recognition problem* (i) is solved in § 3. The *ordering problem* (ii) is solved for strongly connected digraphs in § 4, and its extension to arbitrary digraphs is given in § 5.

3. Preliminary results and a recognition algorithm. The following three lemmas are used to prove our main result. The proofs, being straightforward, are omitted.

LEMMA 1. *Let $G = (V, E)$ be a strongly connected digraph, let C be a simple cycle in G, and let $m \in V$ be such that m is not on C. Then there is some node c on C such that there is a path from c to m that does not contain any node on C other than c.*

LEMMA 2. *Let $G_\alpha = (V, E, \alpha)$ be an* FLR*-ordered digraph and let $C = (V_1, E_1)$ be a simple cycle in G. Suppose $v \in V$ is such that $\alpha^{-1}(v) > \alpha^{-1}(u)$ for all $u \in V_1$ and suppose that there is a path P from some $c \in V_1$ to v such that c is the only node in V_1 that lies on P. Then $\alpha^{-1}(c) = \max\{\alpha^{-1}(u) : u \in V_1\}$.*

The final lemma in this section defines a test for the FLR property in ordered digraphs that, in practice, is more convenient than the test provided by the definition.

LEMMA 3. *Let $G = (V, E, \alpha)$ be an ordered digraph. Then G is not* FLR *ordered if and only if there are three distinct nodes i, j, and $k \in V$ such that $\alpha^{-1}(j) < \alpha^{-1}(i) < \alpha^{-1}(k)$, $i \to j$ is in E, and there is a forward path not containing i from j to k.*

This lemma also provides a simple polynomial time solution to the recognition problem. A forward path $j \to j_1 \to \cdots \to j_s \to k$ is said to *jump* i if $\alpha^{-1}(j) < \alpha^{-1}(i) < \alpha^{-1}(k)$ and $i \notin \{j_1, j_2, \cdots, j_s\}$. By Lemma 3, if an ordered digraph is not FLR ordered, then there must be a backward edge $i \to j$ and a forward path, from j, that jumps i. This suggests the following algorithm for the recognition problem. First, compute H, the set of all pairs (i, j) such that i precedes j in the ordering and there is a forward path from i that jumps j. Then intersect H with $\{(i, j) : j \to i \text{ is a backward edge}\}$. The ordering is an FLR ordering if and only if this intersection is empty. It is easily shown that the running time for this algorithm is $O(ne)$, where n is the number of nodes and e the number of edges in the input digraph.

4. FLR orderings on strongly connected digraphs. The ordering problem for strongly connected digraphs is solved by Algorithm 2 in this section, and the extension to arbitrary

digraphs is made in § 5. One of the key concepts behind Algorithm 2 is the "feasible set" corresponding to an edge, which we now define. Let $G = (V, E)$ be a digraph. For each edge $i \to j$ in E, the *feasible set* FEAS $[i, j]$ is the set of all nodes in $V - \{i, j\}$ that are not reachable from j in the induced subdigraph $G - i$.

If $i \to j \in E$, then FEAS $[i, j]$ can be interpreted as the set of all nodes $k \in V$, for which it is *feasible* that $\alpha^{-1}(k) > \alpha^{-1}(i)$ when α is an FLR ordering on G, and $\alpha^{-1}(i) > \alpha^{-1}(j)$. Feasible sets are used in Algorithm 2 to construct acyclic subdigraphs from which FLR orderings can be derived.

The following algorithm, which runs in time $O(e^2)$, computes the feasible sets of an arbitrary digraph G. It uses a method adapted from a digraph exploration algorithm in [6, pp. 18–19].

ALGORITHM 1: An algorithm to compute the feasible sets of a digraph

Input: A digraph $G = (V, E)$ with adjacency list structure ADJ, where $V = \{1, 2, \cdots, n\}$.

Output: A set FEASIBLE of triples $(i, j,$ FEAS $[i, j])$, where FEAS $[i, j]$ is the feasible set associated with edge $i \to j$.

Data Structures: ACTIVE is a stack used to store a set of nodes from which outgoing edges are to be explored. For each edge $i \to j$ in E, REACH denotes the union of $\{i, j\}$ and the set of nodes in V that are reachable from j without passing through i.

Algorithm:

```
(1)  Initialize FEASIBLE to be empty;
(2)  for each edge i → j in E do begin
(3)    Initialize ACTIVE ← {j}, REACH ← {i, j};
(4)    while ACTIVE ≠ ∅ do begin
(5)      pop node v from ACTIVE;
(6)      for each w ∈ ADJ [v] do begin
(7)        if w ∉ REACH then
(8)          add w to REACH;
(9)          push w onto ACTIVE;
           endif
         endfor
         endwhile
(10)     FEAS [i, j] ← V-REACH;
(11)     add (i, j, FEAS [i, j]) to FEASIBLE;
       endfor
(12)   return FEASIBLE.
```

As another preliminary to Algorithm 2, a connection between FLR orderings on a digraph and topological sorts on a certain acyclic subdigraph is established. It is shown how the feasible sets of a digraph G (for which an FLR ordering exists) can be used to extract an acyclic subdigraph of G, any topological sort of which is an FLR ordering on G; this is the basis for the ordering algorithm.

If $G = (V, E)$ is a digraph and $u \in V$, then denote by $\text{FLRO}_G(u)$ the set of FLR orderings α on G such that $\alpha(n) = u$. If $P = (V, D)$ is an acyclic digraph (on the same node set) and $u \in V$, then denote by $\text{TS}_P(u)$ the set of topological sorts β on P such that $\beta(n) = u$.

THEOREM 2. *Let $G = (V, E)$ be a strongly connected digraph and let $u \in V$. Let $P_u = (V, D)$ be the spanning subdigraph of G such that $i \to j \in D$ if and only if $i \to j \in E$, $i \neq u$ and $u \notin$ FEAS$[i, j]$. Then $\text{FLRO}_G(u) \neq \varnothing$ if and only if P_u is acyclic. Furthermore, if P_u is acyclic, then $\text{FLRO}_G(u) = \text{TS}_{P_u}(u)$.*

Proof. Suppose $\mathrm{FLRO}_G(u) \neq \varnothing$, but P_u is not acyclic. Let $\alpha \in \mathrm{FLRO}_G(u)$ and let C: $c_0 \to c_1 \to \cdots \to c_k \to c_0$ be a simple cycle in P_u. Then C is also a simple cycle in G_α. The node u cannot be on C since P_u is constructed so that u has no outgoing edges. Consequently, by Lemma 1, there exists a node c_i on C such that there is a simple path in G_α from c_i to u that does not contain any other node on C and, by Lemma 2, the node c_i is uniquely determined. From the definition of P_u, $u \notin \mathrm{FEAS}[c_i, c_{i+1}]$ (where $c_{i+1} = c_0$ if $i = k$) so that there is a path in G_α from c_{i+1} to u that does not contain c_i. If c_j is the last node from C on this path, then $c_j \neq c_i$, which contradicts the uniqueness of c_i. Therefore, P_u is acyclic.

Conversely, assuming P_u is acyclic, there exists a topological sort β on P_u, and any path in P_u is a forward path with respect to β. For any node $w \neq u$, the strong connectivity of G implies the existence of a simple path Q from w to u in G. Since each edge in Q is also an edge in P_u, Q is a forward path in P_u, and hence in G_β. Consequently $\beta(n) = u$ for any topological sort β. Suppose $\mathrm{FLRO}_G(u) = \varnothing$. Then, in particular, β is not an FLR ordering on G so, by Lemma 3, there exist distinct nodes i, j, and k in V such that $\beta^{-1}(j) < \beta^{-1}(i) < \beta^{-1}(k)$, $i \to j \in E$, and there is a forward path R in G_β, not containing i, from j to k. Since $i \to j$ is not an edge in P_u (otherwise $\beta^{-1}(i) < \beta^{-1}(j)$) and $i \neq u$, $u \in \mathrm{FEAS}[i, j]$. Therefore, every path in G from j to u must contain i. The strong connectivity of G implies either $k = u$ or, as noted above, the existence of a forward path S in G_β from k to u. In the former case, the path R provides a contradiction. In the latter case, i cannot lie on the forward path S so the concatenation of the paths R and S is a path from j to u not containing i, a contradiction. Since assuming otherwise leads to a contradiction, we can conclude that $\mathrm{FLRO}_G(u) \neq \varnothing$.

The above argument proves that $\mathrm{TS}_{P_u}(u) \subseteq \mathrm{FLRO}_G(u)$. Suppose $\alpha \in \mathrm{FLRO}_G(u)$ and assume that α is not a topological sort of P_u. Then there exists an edge $i \to j$ in P_u (hence in G, also) such that $i \neq u$ and $\alpha^{-1}(i) > \alpha^{-1}(j)$. However, since the defining properties of P_u require that $u \notin \mathrm{FEAS}[i, j]$, and since $\alpha^{-1}(u) > \alpha^{-1}(i)$, Lemma 3 implies that α is not an FLR ordering on G, a contradiction. Therefore, $\mathrm{FLRO}_G(u) \subseteq \mathrm{TS}_{P_u}(u)$. □

Theorem 2 leads immediately to an ordering algorithm for strongly connected digraphs. If we first compute the feasible sets of the input digraph, then it is only necessary to execute a loop in which we compute P_u for each node u until either P_u is found to be acyclic (in which case a topological sort of P_u is output as an FLR ordering on G), or all nodes have been tried.

ALGORITHM 2: An FLR-ordering algorithm for strongly connected digraphs

Input: A strongly connected digraph $G = (V, E)$.

Output: An FLR ordering of G, if one exists; otherwise, a message of failure.

Data Structures: FEASIBLE is the array of triples returned by Algorithm 1. $P_u = (V, D)$ denotes the subdigraph computed from G by the method of Theorem 2 for a particular $u \in V$.

Subalgorithms: Algorithm 1 is used to compute the set FEASIBLE. The algorithm TOPSORT is used to return a topological sort (see § 2.1) of an acyclic digraph.

Algorithm:

```
(1)  Compute the set FEASIBLE using Algorithm 1;
(2)  D ← ∅;
(3)  for each u ∈ V do begin
(4)     for each i → j ∈ E do begin
(5)        if i ≠ u and u ∉ FEAS[i, j] then
```

```
(6)          add i → j to D;
           endif
         end for
(7)      if P_u is acyclic then
(8)        return TOPSORT (P_u);
         endif
       end for
(9)    return "No FLR ordering exists."
```

The worst-case running time of this algorithm is $O(e^2)$ if the feasible sets are represented by Boolean arrays, and $O(n^2e)$ if they are represented by lists.

Algorithm 2 considers each of the nodes, in turn, as a possible high node for an FLR ordering; however, it is sometimes possible to exclude certain nodes from consideration in the **for** loop of line (3). The simplest situation occurs when both $i \to j$ and $j \to i$ are edges and $\text{FEAS}[i, j] = \text{FEAS}[j, i] = \varnothing$. Then any FLR ordering of G must have either i or j as the highest numbered node in the ordering, and we need to loop over only $\{i, j\}$ in line (3). Furthermore, if there are two disjoint pairs with this property, then no FLR ordering can exist.

Algorithm 2 may not be the most efficient FLR-ordering algorithm for digraphs with certain additional properties. For instance, if G is a tree, then both the ordering and recognition problems can be solved in $O(n)$ time. This follows from the equivalence of FLR-ordered trees and minimum-degree-ordered trees (see § 2.1). We omit the proof, which is partially in [5] and [8].

THEOREM 3. *Let $T = (V, E, \alpha)$ be an ordered tree. Then the following conditions on α are equivalent*:

(1) *α is an* FLR *ordering on T,*

(2) *α is an invariant ordering on T,*

(3) *α is a minimum degree ordering on T.*

We conclude this section by noting a strong connection between invariantly ordered trees and MFLR-ordered digraphs (see § 2.2). The next two results of [8], stated here without proof, characterize digraphs that are MFLR ordered.

THEOREM 4. *Let $G = (V, E, \alpha)$ be a strongly connected,* MFLR*-ordered digraph. Then G contains an (invariantly ordered) spanning tree.*

THEOREM 5. *Let T be an invariantly ordered tree with ordering α. Then there is a unique* MFLR*-ordered digraph, with ordering α, having T as a spanning tree.*

5. Extensions of the results. There are obvious analogues of the global inheritance problem for inheritance of matrix entries in the matrix L of the right unit LU factorization and in the matrices L and U of the (normalized) UL factorizations; there are analogues of an FLR ordering for these problems (see [5]). With simple modifications, the results of § 4 can be extended to these other types of orderings on digraphs (see [8]).

As stated, Algorithm 2 only applies to strongly connected digraphs. There is a natural way to extend the algorithm to arbitrary digraphs, however.

If $G = (V, E)$ is an arbitrary digraph, then the set $S = \{s_1, s_2, \cdots, s_t\}$ of strongly connected components of G can be computed in time $O(e)$ (see [9]). Define a digraph $H = (S, D)$, where $s_p \to s_q \in D$ if and only if $s_p \neq s_q$ and there is an edge $i \to j \in E$ such that $i \in s_p$ and $j \in s_q$. H is acyclic, since a cycle $s_{p_1} \to s_{p_2} \to \cdots \to s_{p_r} \to s_{p_1}$ in H would connect $s_{p_1}, s_{p_2}, \cdots, s_{p_r}$ into a single strongly connected component.

If any element of S is not an FLR-orderable digraph, then G itself is not FLR orderable. Assume, therefore, that each strongly connected component of G has an FLR ordering. Since H is acyclic, there exists a topological sort $\beta : \{1, 2, \cdots, t\} \to S$. Denote

by s_i the component $\beta(i)$ in S, let k_i be the number of nodes in s_i and let α_i be an FLR ordering on s_i. The following algorithm produces an FLR ordering α on G, given the orderings α_i.

```
(1)  N ← n;
(2)  for i from 1 to t do begin
(3)    for j from k_i down to 1 do begin
(4)      α(N) ← α_i(j);
(5)      N ← N − 1;
       end for
     end for
(6)  return α.
```

This algorithm numbers the nodes in component s_i in the same relative order as α_i, and ensures that all such nodes are numbered higher than any node in any s_j for $j > i$. In other words, all edges joining nodes in different components in S are backward edges.

It is easily shown that α is an FLR ordering on G, thus proving the following theorem.

THEOREM 6. *Let G be an arbitrary digraph. Then G is* FLR *orderable if and only if all of the strongly connected components of G are* FLR *orderable.*

Acknowledgments. The authors thank Professors C. R. Johnson and P. van den Driessche for their assistance in formulating the problems solved in this paper.

REFERENCES

[1] J. A. BONDY AND U. S. R. MURTY, *Graph Theory with Applications*, North-Holland, New York, 1976.
[2] I. S. DUFF, A. M. ERISMAN, AND J. K. REID, *Direct Methods for Sparse Matrices*, Oxford University Press, Oxford, UK, 1986.
[3] A. GEORGE AND J. W. H. LIU, *Computer Solutions of Large Sparse Positive Definite Systems*, Prentice–Hall, Englewood Cliffs, NJ, 1981.
[4] F. HARARY, *Graph Theory*, Addison–Wesley, Reading, MA, 1969.
[5] C. R. JOHNSON, D. D. OLESKY, AND P. VAN DEN DRIESSCHE, *Inherited matrix entries*: LU *factorizations*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 94–104.
[6] K. MEHLHORN, *Data Structures and Algorithms* 2: *Graph Algorithms and* NP-*Completeness*, Springer-Verlag, Berlin, New York, 1984.
[7] D. J. ROSE AND R. E. TARJAN, *Algorithmic aspects of vertex elimination on directed graphs*, SIAM J. Appl. Math., 34 (1978), pp. 176–197.
[8] T. A. SLATER, *Recognition and ordering algorithms concerning global inheritance in* LU *factorizations*, M. S. thesis, University of Victoria, Victoria, BC, 1988.
[9] R. E. TARJAN, *Depth-first search and linear graph algorithms*, SIAM J. Comput., 1 (1972), pp. 146–160.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 74–80, February 1992
© 1992 Society for Industrial and Applied Mathematics
006

GENERALIZED CHROMATIC NUMBERS OF RANDOM GRAPHS*

EDWARD R. SCHEINERMAN†

Abstract. Given a hereditary (closed under taking induced subgraphs) class of graphs $\mathcal{P}$, the $\mathcal{P}$-chromatic number of a graph G, denoted $\chi_{\mathcal{P}}(G)$, is defined to be the least integer k such that $V(G)$ admits a partition into k subsets each of which induce a member of $\mathcal{P}$. The $\mathcal{P}$-chromatic number of random graphs G on n vertices with fixed edge probability $0 < p < 1$ is studied and it is shown that $\chi_{\mathcal{P}}(G) \sim cn$ in case $|\mathcal{P}| < \infty$ and $\chi_{\mathcal{P}}(G) = \Theta(n/\log n)$ in case $|\mathcal{P}| = \infty$. Also considered are generalized edge chromatic numbers.

Key words. random graphs, chromatic numbers
AMS(MOS) subject classifications. 05C80, 05C15

1. Introduction. Of the many graph parameters, perhaps the best known and most important is the *chromatic number*, $\chi(G)$. This is defined as the minimum integer k such that $V(G)$ can be partitioned $V = V_1 \cup V_2 \cup \cdots \cup V_k$ such that each of the induced subgraphs $G[V_i]$ is edgeless. Many other parameters are defined in a similar way. For example, the *vertex arboricity* of a graph G is the minimum k such that the vertex set of G can be partitioned into sets which induce acyclic graphs. Other examples include cochromatic number (each $G[V_i]$ must be either a complete graph or an edgeless graph) and vertex thickness (each $G[V_i]$ must be a planar graph, as in [5]).

Indeed, for *any* family of graphs $\mathcal{P}$ one can define a *$\mathcal{P}$-chromatic number* of a graph G, denoted $\chi_{\mathcal{P}}(G)$, to be the minimum integer k for which there exists a partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$ such that each of the induced subgraphs $G[V_i]$ is in $\mathcal{P}$. (We call such a partition of $V(G)$ a $\mathcal{P}$-partition of G. If no such partition exists, we say that $\chi_{\mathcal{P}}(G)$ is undefined.) Notice that $\chi_{\mathcal{P}}(G) = 1$ if and only if $G \in \mathcal{P}$. This notion of generalized chromatic number has been previously introduced and studied by others (see, for example, [2], [11], and [12] for a general discussion, as well as [18] for a discussion of the H-free chromatic number of sparse random graphs). If we were to allow $\mathcal{P}$ to be completely arbitrary, there is little of interest we can say about $\chi_{\mathcal{P}}$. We therefore impose the following restrictions on the classes of graphs $\mathcal{P}$ that we consider:

- The class $\mathcal{P}$ is called *hereditary* (or, by some authors, *monotone*) if $G \in \mathcal{P}$ and H is an induced subgraph of G, then $H \in \mathcal{P}$.
- The class $\mathcal{P}$ is called *nontrivial* if it is nonempty and not all graphs are in $\mathcal{P}$.

These restrictions on $\mathcal{P}$ imply, and are implied by, the following properties for $\chi_{\mathcal{P}}(G)$:

- For all graphs G, $\chi_{\mathcal{P}}(G)$ is defined,
- For some graph G, $\chi_{\mathcal{P}}(G) \neq 1$, and
- $\chi_{\mathcal{P}}$ is monotone; that is, for all graphs G and H, if G is an induced subgraph of H, then $\chi_{\mathcal{P}}(G) \leqq \chi_{\mathcal{P}}(H)$.

(The actual chromatic number χ satisfies a 'strong monotonicity' property; namely, if G is any subgraph of H, then $\chi(G) \leqq \chi(H)$. Strong monotonicity does not necessarily hold for the $\chi_{\mathcal{P}}$'s we consider.)

LEMMA 1. *If $\mathcal{P}$ is a class of graphs, then $\mathcal{P}$ is nontrivial and hereditary if and only if $\chi_{\mathcal{P}}$ is defined for all graphs, nonconstant and monotone.*

We are particularly interested in studying $\chi_{\mathcal{P}}$ for *random graphs*. The random graph $G_{n,p}$ is a graph on n labeled vertices. An edge is placed between two vertices with probability p (or the edge is absent with probability $1 - p$). The adjacency of each pair of vertices

* Received by the editors March 23, 1990; accepted for publication January 29, 1991.
† Department of Mathematical Sciences, The Johns Hopkins University, Baltimore, Maryland 21218. This research was supported in part by Office of Naval Research contract N00014-85-K0622.

is independent of any other pair (see [3] and [16]). Bollobás [4] has shown that for fixed p, the chromatic number of almost all $G_{n,p}$ is

$$\chi(G_{n,p}) \sim \frac{n}{2 \log_d n},$$

where $d = 1/(1-p)$. More precisely, this means that there is a function $f(n)$ with $f(n) \to 0$ as $n \to \infty$ such that

$$\lim_{n \to \infty} \Pr\left\{\left(\frac{1}{2} - f(n)\right)\frac{n}{\log_d n} \leqq \chi(G_{n,p}) \leqq \left(\frac{1}{2} + f(n)\right)\frac{n}{\log_d n}\right\} = 1.$$

The fact that $\chi(G_{n,p}) = \Theta(n/\log n)$ is not surprising. Indeed, for many graph theoretic parameters f we see $f(G_{n,p}) = \Theta(n/\log n)$ (see, for example, [17] in which the *interval number* of a random graph is shown to be $\Theta(n/\log n)$). One of the purposes of this paper is to provide a more general setting for this type of result.

Shamir [18] has considered the problem of computing H-free chromatic numbers of random graphs. These generalized chromatic numbers are the same as our $\chi_{\mathscr{P}}$ when $\mathscr{P}$ is the class of graphs that do not contain a fixed graph H as an induced subgraph. Shamir explores the behavior of the H-free chromatic number for random graphs in which the edge probability p tends to 0 as $n \to \infty$. Regrettably, his methods do not carry over to the case when p is constant, which we consider.

Our goal is to analyze the behavior of $\chi_{\mathscr{P}}$ for any hereditary property $\mathscr{P}$ in the case p = constant. We have two cases: When $\mathscr{P}$ is finite, we prove that $\chi_{\mathscr{P}}(G_{n,p}) \sim cn$ and determine the constant c. Otherwise ($|\mathscr{P}| = \infty$), we prove that $\chi_{\mathscr{P}}(G_{n,p}) = \Theta(n/\log n)$.

Here we collect various results required to prove our assertions. We begin with a result from Kingman [13] (see also [10] and [14]). A family of random variables $X_{s,t}$ (where $0 \leqq s < t$ are integers) is called a subadditive process if it satisfies the following conditions:

- For all $0 \leqq s < t < u$ we have $X_{s,u} \leqq X_{s,t} + X_{t,u}$,
- The joint distributions of the processes $(X_{s,t})$ and $(X_{s+1,t+1})$ are the same, and
- $E|X_{0,t}|$ exists and is finite, and $E(X_{0,t}) \geqq At$ for some constant A and all $t > 0$.

THEOREM 2 (Subadditive ergodic). *If $X_{s,t}$ is a subadditive process then*

1. $\gamma = \lim_n \frac{1}{n} E(X_{0,n}) = \inf_n \frac{1}{n} E(X_{0,n})$,
2. $X = \lim_n \frac{1}{n} X_{0,n}$ *exists almost surely and in* L_1, *and*
3. $E(X) = \gamma$.

Given a graph H, a second graph G is called *H-free* if no subset of $V(G)$ induces a graph isomorphic to H. In [6] Erdös and Gimbel investigate the size of the largest H-free subgraph of a random graph.

THEOREM 3 (H-free subgraphs). *Let H be a fixed graph and $0 < p < 1$ be a constant. There exist positive constants $c_1 < c_2$ (depending only on p and H) such that in almost every $G_{n,p}$ the largest H-free induced subgraph of G has between $c_1 \log n$ and $c_2 \log n$ vertices.*

We also use Ramsey's theorem (see [9]).

THEOREM 4 (Ramsey). *For every integer $n > 0$ there exists an integer $r(n)$ such that if G is a graph on at least $r(n)$ vertices, then G contains K_n or nK_1 as an induced subgraph.*

2. Finite $\mathscr{P}$. In this section we consider nontrivial, hereditary classes of graphs $\mathscr{P}$ that are finite. Our main result is that $\chi_{\mathscr{P}}(G_{n,p}) \sim cn$ for some constant n. Moreover,

we can put the random variables $\chi_{\mathscr{P}}(G_{n,p})$ in a common sample space and achieve the almost sure convergence of $\chi_{\mathscr{P}}(G_{n,p})/n$ to a constant random variable. Finally, we show that the constant c is the reciprocal of the number of vertices in a largest graph in $\mathscr{P}$.

Denote by $k_{\mathscr{P}}$ the maximum order of a graph in $\mathscr{P}$, i.e.,

$$k_{\mathscr{P}} = \max \{ |V(G)| : G \in \mathscr{P} \}.$$

Note that $k_{\mathscr{P}} < \infty$ if and only if $\mathscr{P}$ is finite.

LEMMA 5. *If $\mathscr{P}$ is a finite, nontrivial, hereditary class of graphs, then*

$$\chi_{\mathscr{P}}(G) \geqq \frac{|V(G)|}{k_{\mathscr{P}}}.$$

Proof. Let $V(G) = V_1 \cup \cdots \cup V_t$ be a partition of $V(G)$ with each $G[V_i] \in \mathscr{P}$ where $t = \chi_{\mathscr{P}}(G)$. Thus, $|V(G)| = |V_1| + \cdots + |V_t| \leqq tk_{\mathscr{P}}$ and therefore $\chi_{\mathscr{P}}(G) \geqq |V(G)|/k_{\mathscr{P}}$. □

Next we consider the infinite random graph $G(N, p)$ whose vertex set consists of the nonnegative integers in which two vertices are independently joined by an edge with probability p. Let $G_p[s, t)$ denote the induced subgraph of $G(N, p)$ on vertex set $\{s, s+1, \cdots, t-1\}$. Finally, define the random variable $X_{s,t} = \chi_{\mathscr{P}}(G_p[s, t))$.

LEMMA 6. *The random variables $X_{s,t}$ defined above form a subadditive process.*

Proof. First observe that for all $0 \leqq s < t < u$

$$X_{s,u} \leqq X_{s,t} + X_{t+1,u}$$

since if $V_1 \cup \cdots \cup V_a$ is a $\mathscr{P}$-partition of $G_p[s, t)$ and $W_1 \cup \cdots \cup W_b$ is a $\mathscr{P}$-partition of $G_p[t+1, u)$, then $V_1 \cup \cdots \cup V_a \cup W_1 \cup \cdots \cup W_b$ is a $\mathscr{P}$-partition of $G_p[s, u)$.

It is clear from the definition that the processes $(X_{s,t})$ and $(X_{s+1,t+1})$ have the same distribution.

Finally, the previous lemma shows that $X_{0,t} \geqq At$ where $A = 1/k_{\mathscr{P}}$. □

Thus by applying the subadditive ergodic theorem we know that $X_{0,n}/n$ converges almost surely to a random variable X. Furthermore, the following theorem holds.

THEOREM 7. *If $\mathscr{P}$ is a finite, nontrivial, hereditary class of graphs, and $0 < p < 1$ is a constant, then*

$$\chi_{\mathscr{P}}(G_{n,p}) \sim \frac{n}{k_{\mathscr{P}}}.$$

Moreover, considering $X_n = \chi_{\mathscr{P}}(G_{n,p}) = \chi_{\mathscr{P}}(G_p[0, n))$ as random variables defined on the common sample space $G(N, p)$ we have

$$\frac{\chi_{\mathscr{P}}(G_{n,p})}{n} \to \frac{1}{k_{\mathscr{P}}}$$

almost surely.

Proof. For any graph on n vertices we have $\chi_{\mathscr{P}}(G) \geqq n/k_{\mathscr{P}}$. Let $H \in \mathscr{P}$ with $|V(H)| = k_{\mathscr{P}}$. Now consider the following algorithm on a graph G with n vertices.

1. Let $S := V(G)$ and $i := 1$.
2. While $|S| > \log^2 n$, do
   (a) Let $V_i \subset S$ such that $G[V_i]$ is isomorphic to H.
   (b) Let $S := S - V_i$.
   (c) Let $i := i + 1$.
3. Note that $S = \{v_1, v_2, \cdots, v_s\}$ where $s = \lfloor \log^2 n \rfloor$, and let $W_j := \{v_j\}$ for $j = 1, \cdots, s$.

Note that this algorithm produces a $\mathscr{P}$-partition of G with $(1 + o(1))n/k_{\mathscr{P}}$ parts provided step (2a) is always successful. However, in almost all $G_{n,p}$'s, step (2a) must be successful since $|S| > \log^2 n$ and by the H-free subgraphs theorem, $G[S]$ must contain a copy of H.

The subadditive ergodic theorem implies the existence of a random variable X defined on $G(N, p)$ such that $X_{0,n}/n \to X$ almost surely and therefore in probability. The above theorem says that $X_{0,n}/n \to 1/k_{\mathscr{P}}$ in probability. Taken together, these imply that $X = 1/k_{\mathscr{P}}$ almost surely. □

(The "$|S| > \log^2 n$" test condition in step (2) can be replaced by "While H is an induced subgraph of $G[S]$." Also, step (3) may be replaced by a more sophisticated coloring scheme for the remainder of the graph. These improvements may produce a superior $\mathscr{P}$-coloring of the graph in question, but for random graphs, these improvements are $o(n)$.)

3. Infinite $\mathscr{P}$. In the previous section we considered the case $|\mathscr{P}| < \infty$. Although this case produces interesting results, the parameters $\chi_{\mathscr{P}}$ that are of greater interest are those in which $|\mathscr{P}|$ is infinite. We show that for constant p, almost all graphs have $\chi_{\mathscr{P}}(G_{n,p}) = \Theta(n/\log n)$.

We begin with the following consequence of Ramsey's theorem.

LEMMA 8. *Let $\mathscr{P}$ be a hereditary class of graphs. Then $|\mathscr{P}| = \infty$ if and only if for all n, $K_n \in \mathscr{P}$ or for all n, $nK_1 \in \mathscr{P}$.*

Proof. First suppose that $|\mathscr{P}| = \infty$ but that there exist positive integers n_1 and n_2 such that $K_{n_1} \notin \mathscr{P}$ and $n_2K_1 \notin \mathscr{P}$. Since $\mathscr{P}$ is hereditary, it follows if $n = \max\{n_1, n_2\}$ then neither K_n nor nK_1 is in $\mathscr{P}$. Thus, by Ramsey's theorem, if G has $r(n)$ (or more) vertices, G contains either K_n or nK_1 as an induced subgraph. Thus $G \notin \mathscr{P}$. Thus all graphs in $\mathscr{P}$ have fewer than $r(n)$ vertices, a contradiction.

Second, if $\mathscr{P}$ contains all K_n's or all nK_1's then clearly $|\mathscr{P}| = \infty$. □

This result has the following interesting corollary. Let $\bar{G}$ denote the complement of G.

LEMMA 9. *If $\mathscr{P}$ is an infinite, hereditary class of graphs and G is a graph, then*

$$\chi_{\mathscr{P}}(G) \leqq \max\{\chi(G), \chi(\bar{G})\}.$$

Proof. Note that $V(G)$ can be partitioned into $\chi(G)$ sets, each of which induces an edgeless graph (independent sets). Also, $V(G)$ can be partitioned into $\chi(\bar{G})$ sets, each of which induces a complete graph (cliques). Thus if $\mathscr{P}$ contains all nK_1's, then $\chi_{\mathscr{P}}(G) \leqq \chi(G)$ and if $\mathscr{P}$ contains all K_n's, then $\chi_{\mathscr{P}}(G) \leqq \chi(\bar{G})$. □

We now combine Bollobás's result with that of Gimbel and Erdös to prove the following result.

THEOREM 10. *Let $\mathscr{P}$ be an infinite nontrivial hereditary class of graphs and let $0 < p < 1$ be a constant. There exist constants $0 < c_1 < c_2$ such that for almost all $G_{n,p}$,*

$$c_1 \frac{n}{\log n} \leqq \chi_{\mathscr{P}}(G_{n,p}) \leqq c_2 \frac{n}{\log n}.$$

Proof. By Bollobás's result [4] and the previous lemma, almost every $G_{n,p}$ satisfies

$$\chi_{\mathscr{P}}(G_{n,p}) \leqq \left[\frac{1}{2} + o(1)\right] \max\left\{\frac{n}{\log_d n}, \frac{n}{\log_b n}\right\},$$

where $d = 1/(1-p)$ and $b = 1/p$.

Since $\mathscr{P}$ is nontrivial, then there is some graph $H \notin \mathscr{P}$. Let $t = \chi_{\mathscr{P}}(G)$ and let $V_1 \cup \cdots \cup V_t$ be a $\mathscr{P}$-partition of $V(G)$. Since $G[V_i] \in \mathscr{P}$, we have $G[V_i]$ is H-free.

Therefore, by the H-free subgraph theorem, $|V_i| \leqq c \log n$ for some constant c. Thus

$$n = |V_1| + \cdots + |V_t| \leqq (c \log n)t$$

and therefore $\chi_{\mathscr{P}}(G) = t \geqq n/(c \log n)$. □

Note. The lower bound in the above argument was also observed by Gimbel [8].

4. Open problems.

4.1. Matching upper and lower bounds: $\chi_{\mathscr{P}} \sim n/\alpha_{\mathscr{P}}$? When $\mathscr{P}$ is finite, we have shown that $\chi_{\mathscr{P}}(G_{n,p}) = [c + o(1)]n$, where c is a constant depending only on $\mathscr{P}$. When $\mathscr{P}$ is infinite, we have shown the existence of positive constants c_1 and c_2 (which depend on $\mathscr{P}$ and on p, but not n) such that almost all graphs $G_{n,p}$ satisfy

$$c_1 \frac{n}{\log n} \leqq \chi_{\mathscr{P}}(G_{n,p}) \leqq c_2 \frac{n}{\log n}.$$

However, we conjecture the following.

CONJECTURE. *Let $\mathscr{P}$ be an infinite nontrivial hereditary class of graphs and let p be a constant with $0 < p < 1$. There exists a constant c (depending on $\mathscr{P}$ and p, but not n) such that for almost all $G_{n,p}$ we have*

$$\chi_{\mathscr{P}}(G_{n,p}) = [c + o(1)]\frac{n}{\log n}.$$

(Let $\alpha_{\mathscr{P}}$ denote the expected maximum order of an induced subgraph $H \leqq G_{n,p}$ for which $H \in \mathscr{P}$. It seems likely that the more daring conjecture *almost all $G_{n,p}$ are such that* $\chi_{\mathscr{P}}(G_{n,p}) \sim n/\alpha_{\mathscr{P}}$ should hold. This was proved by Bollobás [4] in the case $\mathscr{P} = \{nK_1\}$ and by Shamir [18] in the case $p \to 0$ and $\mathscr{P}$ is the class of H-free graphs for some fixed H.)

The work of Shamir and Spencer [19] suggests that this conjecture is true. Using martingale methods, they prove that the chromatic number of almost all graphs cannot be too far from the expected chromatic number. By martingale methods (see [1] and [7]) they show that

$$\Pr\{|\chi - E(\chi)| \geqq \omega_n \sqrt{n}\} \to 0$$

as $n \to \infty$, where ω_n denotes any function of n that goes to infinity as n does.

Their proof [19] applies equally well to the $\chi_{\mathscr{P}}$ parameters we have discussed in this paper.

A special case of the above conjecture should be tractable. Let H be a fixed graph and $\mathscr{P}$ be the class of all H-free graphs. A refined version of the H-free subgraphs theorem [6] (which is based, in part, on [20]) coupled with the techniques in [4] and [18] may resolve the above conjecture in this special case. From there, it will be easy to resolve the case when there are only finitely many forbidden graphs for $\mathscr{P}$ (H is called forbidden for $\mathscr{P}$ if $H \notin \mathscr{P}$ but all induced subgraphs of H are in $\mathscr{P}$). The case when both $\mathscr{P}$ and the set of forbidden subgraphs for $\mathscr{P}$ are infinite seems to be the most challenging.

One special case, however, is easy to handle. Let $\mathscr{Q}_k$ denote the set of all k-colorable graphs, i.e., $\mathscr{Q}_k = \{G : \chi(G) \leqq k\}$. Now note that

- If $\mathscr{P} \subset \mathscr{Q}_k$ then for all G, $k\chi_{\mathscr{P}}(G) \geqq \chi(G)$ and
- If $\mathscr{P} \supset \mathscr{Q}_k$ then for all G, $\chi_{\mathscr{P}}(G) \leqq \lceil \chi(G)/k \rceil$.

Thus, if $\mathscr{P} = \mathscr{Q}_k$, then $\chi_{\mathscr{P}}(G_{n,p}) \sim n/(2k \log_d n)$, where $d = 1/(1-p)$.

4.2. Edge analogues. We can also generalize the edge chromatic number. For a given $\mathscr{P}$ define the $\mathscr{P}$-edge chromatic number of a graph $G = (V, E)$, de-

noted $\chi'_{\mathscr{P}}(G)$, to be the minimum k such that the edges can be partitioned $E = E_1 \cup \cdots \cup E_k$, so that each of the graphs (V, E_i) has property $\mathscr{P}$. For example, if $\mathscr{P} = \{G : \Delta(G) \leqq 1\}$ then $\chi'_{\mathscr{P}}(G) = \chi'(G)$. If $\mathscr{P}$ is the property 'is acyclic' then $\chi'_{\mathscr{P}}(G) = \Upsilon(G)$, the edge arboricity of G (see [15]).

We wish to determine the general asymptotic behavior of $\chi'_{\mathscr{P}}$ for a random graph. For $\chi_{\mathscr{P}}$ we saw essentially two kinds of behavior depending on whether or not $|\mathscr{P}|$ is finite. The point of the next result is to show that there are at least three possible behaviors for $\chi'_{\mathscr{P}}$: $\Theta(n^2)$, $\Theta(n)$, and $\Theta(\log n)$. Of additional interest, we show that $\Upsilon(G_{n,p}) \sim np/2$.

THEOREM 11. *Let p be a constant. For almost all graphs $G_{n,p}$ we have*

(A) *If $\mathscr{P}$ is the property 'has maximum degree at most 1' (i.e., $\chi'_{\mathscr{P}} = \chi'$) then $\chi'_{\mathscr{P}}(G_{n,p}) \sim np$.*

(B) *If $\mathscr{P}$ is the property 'is acyclic' (i.e., $\chi'_{\mathscr{P}} = \Upsilon$, the edge arboricity) then $\chi'_{\mathscr{P}}(G_{n,p}) \sim np/2$.*

(C) *If $\mathscr{P} = \mathscr{Q}_k$ is the property 'is k-colorable' then $\chi'_{\mathscr{P}}(G) \sim \log_k n$.*

(D) *If $\mathscr{P}$ is the property 'has at most one edge' then $\chi'_{\mathscr{P}}(G) \sim pn^2/2$.*

Proof of (A). Note that Vizing's theorem gives us $\chi' \leqq \Delta + 1$ and since random graphs have unique vertices of maximum degree we have $\chi' = \Delta \sim np$. (See [16, p. 156]. Bollobás also [3] gives the maximum degree much more precisely.) □

Proof of (B). We use the following result of Nash-Williams [15]:

$$\Upsilon(G) = \max_{H \leqq G} \left\lceil \frac{m_H}{n_H - 1} \right\rceil,$$

where n_H and m_H denote the number of vertices and edges of the induced subgraph H of G.

Since $G_{n,p}$ has $m \sim p\binom{n}{2}$ edges we have $\Upsilon(G_{n,p}) \geqq [1 - o(1)]pn/2$. For the upper bound we first note that if $H \leqq G_{n,p}$ has $n_H < np$ vertices then $m_H/(n_H - 1) \leqq \binom{n_H}{2}/(n_H - 1) = n_H/2 < np/2$. Thus it is enough to consider induced subgraphs H with at least np vertices.

Note that for a given induced subgraph H that m_H is the sum of $\binom{n_H}{2}$ 0–1 random variables with mean p. We can therefore apply a large deviation result (such as Theorem I.7(i) on p. 13 of [3]). Let $\varepsilon = n^{-1/4}$ and observe

$$\begin{aligned}
\Pr\left\{\frac{m_H}{n_H - 1} \geqq pn(1+\varepsilon)/2\right\} &\leqq \Pr\left\{\frac{m_H}{n_H - 1} \geqq pn_H(1+\varepsilon)/2\right\} \\
&\leqq \Pr\left\{m_H \geqq p\binom{n_H}{2}(1+\varepsilon)\right\} \\
&\leqq [\varepsilon^2 p\binom{n_H}{2}]^{-1/2} \exp\left\{-\varepsilon^2 p\binom{n_H}{2}/3\right\} \\
&\leqq \exp\left\{-n^{-1/2} p\binom{np}{2}/3\right\} \\
&= \exp\left\{-n^{3/2}p^3/6 + o(n)\right\} = o(2^{-n}).
\end{aligned}$$

Since these are fewer than 2^n induced subgraphs $H \leqq G_{n,p}$ with $n_H \geqq np$ it follows that the probability that some subgraphs G has $m_H/(n_H - 1) \geqq (1 + \varepsilon)np/2$ vanishes and therefore $\Upsilon(G_{n,p}) \geqq [1 + o(1)]np/2$ for almost all $G_{n,p}$. □

Proof of (C). We have $\mathscr{P} = \mathscr{Q}_k = \{G : \chi(G) \leqq k\}$. We first show that for any graph G we have $\chi'_{\mathscr{P}}(G) \leqq t$ if and only if $\chi(G) \leqq k^t$.

If $\chi'_{\mathscr{P}}(G) \leqq t$ partition $E(G) = E_1 \cup \cdots \cup E_t$, where each (V, E_i) is k-colorable. Let $c_i : V \to [k] = \{1, \cdots, k\}$ be a proper k-coloring of (V, E_i). Put $\mathbf{c}(v)$ equal to the vector $[c_1(v), \cdots, c_t(v)] \in [k]^t$ and observe that if $vw \in E(G)$, then $\mathbf{c}(v) \neq \mathbf{c}(w)$ and therefore G is k^t-colorable.

Conversely, if $\chi(G) \leqq k^t$, let $\mathbf{c} : V(G) \to [k]^t$ be a proper coloring. We form a t-partition of $E(G)$ by placing vw in any E_j for which $c_j(v) \neq c_j(w)$. Note that c_j is a proper k-coloring of (V, E_j) and therefore $E_1 \cup \cdots \cup E_t$ is a suitable partition of $E(G)$.

Finally, the equivalence $\chi'_{\mathscr{P}}(G) \leqq t \Leftrightarrow \chi(G) \leqq k^t$ implies $\chi'_{\mathscr{P}}(G) = \lceil \log_k \chi(G) \rceil$. Thus by Bollobás's result [4] that $\chi(G_{n,p}) = [1 + o(1)]n/(2 \log_d n)$ (where $d = 1/(1-p)$) we have

$$\chi'_{\mathscr{P}}(G) = \lceil \log_k n - \log_k \log_d n - \log_k 2 + o(1) \rceil \sim \log_k n. \quad \square$$

Proof of (D). Let $\mathscr{P}$ be the property 'has at most one edge.' Thus $\chi'_{\mathscr{P}}(G_{n,p}) = |E(G_{n,p})| \sim pn^2/2$. $\square$

Acknowledgment. The author would like to thank J. Gordon Gimbel for bringing the H-free subgraphs theorem to his attention and for an interesting discussion concerning these results.

REFERENCES

[1] K. AZUMA, *Weighted sums of certain dependent random variables*, Tôhuku Math. J., 2 (1967), pp. 357–367.
[2] S. BURR AND M. JACOBSON, *On inequalities involving vertex-partition parameters of graphs*, preprint.
[3] B. BOLLOBÁS, *Random Graphs*, Academic Press, New York, 1985.
[4] ———, *The chromatic number of random graphs*, Combinatorica, 8 (1988), pp. 49–55.
[5] G. CHARTRAND AND H. V. KRONK, *The point-arboricity of planar graphs*, J. London Math. Soc., 44 (1969), pp. 612–616.
[6] P. ERDÖS AND J. G. GIMBEL, *A note on the maximal order of an H-free subgraph in a random graph*, Proc. Sixth International Conference on the Theory and Applications of Graphs.
[7] D. A. FREEDMAN, *On tail probabilities for martingales*, Ann. Probab., 1 (1975), pp. 100–118.
[8] J. G. GIMBEL, personal communication.
[9] R. L. GRAHAM, B. L. ROTHSCHILD, AND J. H. SPENCER, *Ramsey Theory*, John Wiley, New York, 1980.
[10] J. M. HAMMERSLEY, *A few seedlings of research*, Proc. Sixth Berkeley Symp. Math. Statist. Prob., University of California Press, 1972.
[11] R. P. JONES, Partition Numbers of Graphs, M. Sc. thesis, University of Calgary, 1973.
[12] ———, *Hereditary properties and P-chromatic numbers*, in Proceedings of the British Combinatorial Conference 1973, T. P. McDonough and V. C. Mavron, eds., 1974, pp. 83–88.
[13] J. F. C. KINGMAN, *Subadditive ergodic theory*, Ann. Probab., 1 (1973), pp. 883–909.
[14] T. M. LIGGETT, *An improved subadditive ergodic theorem*, Ann. Probab., 13 (1985), pp. 1279–1285.
[15] C. ST. J. A. NASH-WILLIAMS, *Decomposition of finite graphs into forests*, J. London Math. Soc., 39 (1964), p. 12.
[16] E. M. PALMER, *Graphical Evolution: An Introduction to the Theory of Random Graphs*, John Wiley, New York, 1985.
[17] E. R. SCHEINERMAN, *On the interval number of random graphs*, Discrete Math., 82 (1990), pp. 105–109.
[18] E. SHAMIR, *Generalized stability and chromatic numbers of random graphs*, preprint.
[19] E. SHAMIR AND J. H. SPENCER, *Sharp concentration of the chromatic number of random graphs $G_{n,p}$*, Combinatorica, 7 (1987), pp. 121–129.
[20] R. M. WILSON, *An existence theory of pairwise balanced block designs*, III, J. Combin. Theory (A), 18 (1975), pp. 71–79.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 81–87, February 1992
© 1992 Society for Industrial and Applied Mathematics
007

A TECHNIQUE FOR LOWER BOUNDING THE COVER TIME*

DAVID ZUCKERMAN†

Abstract. A general technique for proving lower bounds on expected covering times of random walks on graphs in terms of expected hitting times between vertices is given. This technique is used to prove

(i) A tight bound of $\Omega(|V| \log^2 |V|)$ for the two-dimensional torus;

(ii) A tight bound of $\Omega(|V| \log^2 |V|/\log d_{\max})$ for trees with maximum degree $d_{\max}$;

(iii) Tight bounds of $\Omega(\mu^+ \log |V|)$ for rapidly mixing walks on vertex transitive graphs, where μ^+ denotes the maximum expected hitting time between vertices.

In addition to these new results, the technique allows several known lower bounds on cover times to be systematically proved, often in a much simpler way.

Finally, a different technique is used to prove an $\Omega(1/(1-\lambda_2))$ lower bound on the cover time, where λ_2 is the second largest eigenvalue of the transition matrix. This was previously known only in the case where the walk starts in the stationary distribution [*J. Theoret. Probab.*, 2 (1989), pp. 101–120].

Key words. random walks, cover time, torus, trees, vertex transitive

AMS(MOS) subject classification. 60J15

1. Introduction. A random walk on an undirected graph is the sequence of vertices visited by a particle that starts at a specified vertex and visits other vertices according to the following transition rule: if the particle is at vertex i at time t, then at time $t+1$ it moves to a neighbor of i picked uniformly at random. In this paper, we analyze the expected cover time, i.e., the expected time of the random walk to visit all the vertices.

Simulating a random walk on a graph requires very local information about the graph, while random walks have very nice global properties. This makes random walks useful in computation, where limited resources are available to determine global information. For example, random walks have proved useful in designing approximation algorithms for counting problems (see, e.g., [DFK] and [JS]), simulating complexity classes with few random bits [AKS], and assigning processes to nodes in networks [BC]. Bounds on cover times, in particular, were important in showing that UNDIRECTED *st*-CONNECTIVITY can be computed in RSPACE($\log n$) [AKLLR] and in analyzing the simulation of token rings on arbitrary networks [BK].

To understand what is known about cover times, consider for the moment the maximum expected cover time cov, where the maximum is taken over all start vertices. For interesting graphs, changing the start vertex changes the expected cover time by at most a constant factor, so analyzing cov is not really a restriction. Define $E_i T_j$ to be the expected time to get from vertex i to vertex j, and $\mu^+ = \max_{i,j} \{E_i T_j\}$. It is not hard to show that μ^+ characterizes cov to within a $\log n$ factor, where $n = |V|$, i.e., that $\mu^+ \leqq \text{cov} \leqq O(\mu^+ \log n)$ (see, e.g., [Z]). Good techniques have been developed to estimate μ^+, and involve calculating resistances of graphs [CRRST] and eigenvalues [A2]. Indeed, the $E_i T_j$'s are computable in polynomial time, while it is not known if cov is. Therefore, the difficult part in establishing tight bounds for cov tends to be deciding the extra $\log n$ factor.

The basic technique for showing upper bounds of $O(\mu^+)$ is based on spanning trees, first used in [AKLLR] to show an upper bound for all graphs of $O(|V||E|)$, even though μ^+ can be $\Theta(|V||E|)$. In [KLNS] this technique is extended to show the general

* Received by the editors June 5, 1989; accepted for publication (in revised form) September 26, 1990. This research was supported by a National Science Foundation Graduate Fellowship.

† Division of Computer Science, University of California, Berkeley, California 94720.

upper bound $O(|V||E|/d_{\min})$, and in [CRRST] another application of this technique is mentioned.

By contrast, a variety of techniques have been used to show the lower bound $\Omega(\mu^+ \log n)$, even though most of the work on lower bounds has been concentrated toward proving the conjectured lower bound of $\Omega(n \log n)$ for all graphs, regardless of the start vertex. Aldous [A3] has proved this bound if the walk starts from the stationary distribution. Examples of the different techniques are an inductive argument to show the $\Omega(n \log n)$ conjecture for trees [KLNS], a coupon-collector type argument to show the $\Omega(n \log n)$ conjecture for rapidly mixing walks [BK], and use of the $\Theta(\sqrt{n})$ standard deviation law to show an $\Omega(n \log^2 n/\log^2 d_{\max})$ lower bound for trees with small degree [Z].

In this paper, we present a general technique for showing the lower bound $\Omega(\mu^+ \log n)$ that yields all of the lower bounds described above except that given in [A3], as well as new lower bounds. All of our lower bounds are valid for any start vertex.

Our first bound is for the two-dimensional torus. The results in [A1] imply that the cover time for the k-dimensional torus is $\Theta(n \log n)$, for $k \geqq 3$. As the tight bound of $\Theta(n^2)$ is easy to show for the one-dimensional case, this only left open the time for the two-dimensional torus. It was known that $\mu^+ = \Theta(n \log n)$, which implied the best bounds on the cover time of $\Omega(n \log n)$ and $O(n \log^2 n)$ (see, e.g., [CRRST]). We show that the cover time is $\Theta(n \log^2 n)$.

Second, we improve the lower bound for trees in [Z] and [KLNS] to $\Omega(n \log^2 n/\log d_{\max})$; the case of the balanced k-ary tree shows that this is tight in terms of $d_{\max}$. This was obtained independently using a less general version of our method in [DS]. Aldous has since found the constant for balanced k-ary trees [A5].

Third, we give a lower bound for rapidly mixing walks. By rapidly mixing, we mean $\tau_2 \leqq n^{1-\delta}$, $\delta > 0$, where $\tau_2 = 1/(1 - \lambda_2)$ is a measure of how quickly the random walk approaches stationarity (λ_2 is the second largest eigenvalue). This lower bound implies the $\Omega(n \log n)$ bound attained in [BK] for all rapidly mixing walks, as well as tight bounds of $\Theta(\mu^+ \log n)$ for rapidly mixing walks on vertex transitive graphs. This generalizes the result in [A1] showing this for Cayley graphs.

Finally, we use a different technique to show that the expected time to visit a vertex chosen at random according to the stationary distribution does not depend on the start vertex. This lemma implies the conjectured $\Omega(n \log n)$ lower bound for slowly mixing walks. Our $\Omega(\tau_2)$ lower bound was known previously only in the case where the walk starts from the stationary distribution [BK]. This leaves the $\Omega(n \log n)$ conjecture open only for the cases $n^{1-o(1)} \leqq \tau_2 \leqq n \log n$.

Our main technique is of interest in its own right, based on ideas in [Ma]. The difficult part in analyzing the cover time is correlations between hitting times of vertices; i.e., if the particle has visited i, what is the probability that it has visited j? We get around this by specifying random vertices to be visited; it is then easy to calculate the correlations between random vertices.

2. Notation. Let $G(V, E)$ be the graph on which the random walk is performed. For all of the following definitions, assume $i, j \in V$:

$n = |V|$,

d_i = degree of i,

$d_{\max} = \max_i \{d_i\}$,

$d_{\min} = \min_i \{d_i\}$,

$d(i, j)$ = distance between i and j.

Let $\{X_t\}$ be the sequence of vertices visited by the random walk, and let A be the associated transition matrix; i.e., $A_{ij} = 1/d_i$ if j is a neighbor of i, and 0 otherwise. Let

$1 = \lambda_1 > \lambda_2 \geqq \lambda_3 \geqq \cdots \geqq \lambda_n$ be the eigenvalues of A. Define

T_j = time to first reach j,

C = time to cover G,

$P_i(\cdot)$ denotes the probability of $(\cdot)$ in a walk starting at i,

$E_i(\cdot)$ denotes the expectation of $(\cdot)$ in a walk starting at i,

π is the stationary distribution, i.e., $\pi A = \pi$, so $\pi(i) = d_i/2|E|$,

$E_\pi(\cdot)$ denotes the expectation of $(\cdot)$ in a walk starting from distribution π,

$\mu^+ = \max_{i,j}\{E_iT_j\}$,

$\mu^- = \min_{i,j}\{E_iT_j\}$,

$H_k = 1 + 1/2 + \cdots + 1/k$ is the kth harmonic number.

3. The key lemma. Our lemma is based on the following theorem of Matthews [Ma].

THEOREM 1. *For any $v \in V$, $\mu^- H_{n-1} \leqq E_vC \leqq \mu^+ H_{n-1}$.*

We generalize the lower bound so that we still get an extra $\log n$ factor even if we allow, for each i, a polynomial fraction of the j to be close to i. We also allow i and j to be chosen from only a polynomial fraction of the vertices.

LEMMA 2. *Let $V' \subseteq V$ such that $|V'| \geqq n^\alpha$, $\alpha > 0$, and let t be such that for all $i \in V'$, at most $1/n^\beta$ fraction of the $j \in V'$ satisfy $E_iT_j < t$, where $\beta > 0$. Then for any $v \in V$, $E_vC > t(\gamma \ln n - 2)$, where $\gamma = \min(\alpha, \beta)$.*

Proof. We elaborate Matthews' idea by adding an extra element of randomness. Assume without loss of generality that the start vertex $v \in V'$. Let $y_1, \cdots, y_{|V'|-1}$ be a uniformly random permutation of $V' - \{v\}$. Let $Y_k = \{y_1, \cdots, y_k\}$. Let S_k be the first time that all the vertices in Y_k are visited, and let $R_k = S_k - S_{k-1}$. Note that $C \geqq S_{|V'|-1}$.

First, we claim that $P[R_k \neq 0] = 1/k$. The event $R_k \neq 0$ corresponds to y_k being visited after all of Y_{k-1}. We condition on a given walk occurring and on the set Y_k; the randomness left is in the order $y_1, \cdots, y_k$ of Y_k. Then y_k has probability $1/k$ of being the last element of Y_k visited in the walk. Since this is independent of what we condition on, the claim follows.

Now condition on the walk up to time S_{k-1} and on Y_{k-1}, and let $i = X_{S_{k-1}}$. Then, considering only the randomness involved in picking y_k, $P[E_iT_{y_k} < t] \leqq n^{\alpha-\beta}/(n^\alpha - k)$, by definition of t. Note that $E[R_k | R_k \neq 0] = E_iT_{y_k}$. Therefore, for $k \leqq n^\alpha/2$,

$$E[R_k] \geqq (P[R_k \neq 0] - P[E_iT_{y_k} < t])t \geqq \left(1/k - \frac{n^{\alpha-\beta}}{n^\alpha - k}\right)t \geqq (1/k - 2/n^\beta)t.$$

Thus,

$$E_vC \geqq ES_{|V'|-1} = \sum_{k=1}^{|V'|-1} ER_k \geqq \sum_{k=1}^{n^\gamma/2} ER_k$$

$$\geqq t\left(H_{n^\gamma/2} - \frac{n^\gamma}{2}\frac{2}{n^\beta}\right) \geqq t(\gamma \ln n - \ln 2 - 1) \qquad \square$$

4. Application to two-dimensional torus. The two-dimensional torus is the graph $G = (V, E)$, where $V = \{0, 1, 2, \cdots, r\}^2$ and a vertex (a, b) is connected to the four vertices $(a \pm 1 \pmod r, b)$ and $(a, b \pm 1 \pmod r)$.

It is known that $\mu^+ = \Theta(n \log n)$ (see, e.g., [CRRST]), which implies the only known bounds on the expected cover time of $\Omega(n \log n)$ and $O(n \log^2 n)$. We apply our key lemma to show a lower bound of $\Omega(n \log^2 n)$.

LEMMA 3. *$E_iT_j = \Theta(n \log d(i, j))$. In particular, $E_iT_j > \frac{1}{4} n \ln 2d(i, j)$.*

Proof. This follows easily from the ideas in [CRRST]. $\square$

Remark. Aldous [A6] has pointed out that using the results in [C] and some extra work, it is possible to replace the $\frac{1}{4}$ above by $1/\pi + o(1)$, hence improving the constant in Theorem 4 to $1/8\pi + o(1)$.

THEOREM 4. *For any* $v \in V$, $E_v C \geq 1/32n \ln n(\ln n - 3)$.

Proof. We apply the key lemma with $V' = V$ and $t = 1/16n \ln n$. Note that there are at most $2d^2$ vertices j with $d(i, j) < d$. Taking $d = \frac{1}{2}n^{1/4}$, we see that there are $\leq n^{1/2}/2$ vertices j with $d(i, j) < d$, and if $d(i, j) \geq d$, then $E_i T_j > \frac{1}{4}n \ln n^{1/4} = t$. Thus, applying the key lemma with $\gamma = \beta = \log_n (n^{1/2}/2)$ gives the theorem. □

5. Application to trees with small degree. Previous lower bounds for general trees have been $\Omega(n \log n)$ [KLNS] and $\Omega(n \log^2_{d_{\max}} n)$ [Z]. We improve both of these bounds to $\Omega(n \log n \log_{d_{\max}} n)$. This is the best possible, given only n and $d_{\max}$, as we show it is tight for balanced trees. To apply the key lemma, we must analyze $E_i T_j$ for trees, we need the following lemma from [Mo].

LEMMA 5. *For neighbors* i, j, $E_i T_j = 2|A_{ij}| - 1$, *where* A_{ij} *is the subtree containing* i *obtained by deleting edge* $\{i, j\}$.

COROLLARY 6. *In a tree,* $E_i T_j \geq (d(i, j))^2$.

Proof. The above lemma implies that this time will be least in case our graph is a simple path from i to j, for which

$$E_i T_j = 1 + 3 + \cdots + 2d(i,j) - 1 = (d(i,j))^2. \qquad \square$$

LEMMA 7. *For any* i, *there are at most* $O(n^{3/4}\sqrt{\log n})$ *vertices* j *with* $E_i T_j < t = \frac{1}{4}n \log_{d_{\max}} n$.

Proof. We root our tree at i, and put parent-child relations on the vertices as usual. We construct a chain of vertices $i = i_1, i_2, \cdots, i_m$ as follows: we choose i_{j+1} to be the child of i_j with a subtree having at least $n/2$ vertices, if such exists, otherwise $m = j$.

Let I_j be the subtree with root i_j, so $I_j = A_{i_j, i_{j-1}}$ for $j > 0$, and set $I_{m+1} = \varnothing$. Note that if v, $w \in I_j - I_{j+1}$ for $j < m$, with w a child of v, then $|A_{vw}| \geq n/2 + 1$, so $E_v T_w \geq n$. Similarly, for any child w_m of i_m, $|A_{w_m, i_m}| \leq (n - 1)/2$, since otherwise we could have extended our chain. Thus, for v, $w \in I_m$, w a child of v, $|A_{vw}| \geq |A_{i_m, w_m}| = n - |A_{w_m, i_m}| \geq (n + 1)/2$, so $E_v T_w \geq n$.

Therefore, for $w \in I_j - I_{j+1}$ with $d(i_j, w) \geq \frac{1}{4} \log_{d_{\max}} n$, $E_i T_w \geq E_{i_j} T_w \geq t$. Furthermore, if $k \geq \sqrt{t}$, then for any $w \in I_k$, $d(i, w) \geq \sqrt{t}$, so by Corollary 6 $E_i T_w \geq t$. Thus, the only possible vertices w with $E_v T_w < t$ are those in $I_j - I_{j+1}$, $j < \min(\sqrt{t}, m + 1)$, such that $d(i_j, w) < \frac{1}{4} \log_{d_{\max}} n$. There can be at most $(\sqrt{t} + 1) d_{\max}^{1/4 \log_{d_{\max}} n} = (\sqrt{t} + 1) n^{1/4}$ such w, from which the lemma follows. □

THEOREM 8. *For trees, for any* $v \in V$, $E_v C \geq (1/16 - o(1)) n \log_{d_{\max}} n \ln n$.

Proof. The key lemma applies with $\alpha = 1$ and $\beta = \frac{1}{4} - \Theta(\log \log n / \log n)$. □

COROLLARY 9. *For balanced* k-ary *trees, for any* $v \in V$, $E_v C = \Theta(n \log_k n \ln n)$.

Proof. The lower bound follows from the above theorem. The upper bound follows from Theorem 1 by noting that $E_i T_j \leq 2n$ for i, j neighbors, so $E_i T_j \leq 2nd(i, j)$, and the diameter of these trees are at most $2 \log_k n$. □

Remark. Aldous [A5] has since shown that, for balanced k-ary trees, $E_v C \sim 2n \log_k n \ln n$.

6. Application to rapidly mixing walks. We now generalize both the $\Omega(\mu^+ \log n)$ lower bound for Cayley graphs and the $\Omega(n \log n)$ lower bound for rapidly mixing walks [BK], that is graphs where $\tau_2 \leq n^{1-\delta}$ for $\delta > 0$, where $\tau_2 = 1/(1 - \lambda_2)$. We will need the following well-known lemma, which shows that in $O(\tau_2 \log n)$ time the random walk approaches stationarity.

LEMMA 10. *If all eigenvalues are* $\geq -\lambda_2$, *then for* $t \geq (k+2)\tau_2 \ln n$,

$$|P_i[X_t = j] - \pi(j)| \leq n^{-k}.$$

This can be suitably modified if there are eigenvalues $< -\lambda_2$.

Proof. This follows from the spectral representation given in [K] and the facts that $\pi(i) \geq n^{-2}$ and $|\lambda_k|^t \leq \lambda_2^t \leq e^{-t/\tau_2}$. □

LEMMA 11. *Suppose* $\tau_2 \leq n^{1-\delta}$ *for* $\delta > 0$. *Then for any* $\varepsilon > 0$ *and any* $v \in V$,

$$E_v C \geq (1 - o(1))\delta \min_{\pi(i) < (1+\varepsilon)/n} \{E_\pi T_i\} \ln n.$$

Proof. We show that the key lemma applies with $\alpha = (1 - o(1))$, $\beta = \delta - o(1)$, and $t = (1 - o(1)) \min_{\pi(i) < (1+\varepsilon)/n} \{E_\pi T_i\}$. We set $V' = \{i: \pi(i) < (1+\varepsilon)/n\}$. Note that $|V - V'| \leq n/(1+\varepsilon)$, so $|V'| \geq (1 - 1/(1+\varepsilon))n$.

Now fix i. Let $J = \{j: P_i[T_j \leq 5\tau_2 \ln n] \geq 1/\ln n\}$. Then $|J| \leq 5\tau_2 \ln^2 n$, because the sum $\sum_j P_i[T_j \leq 5\tau_2 \ln n]$ is at most the expected number of vertices visited in time $5\tau_2 \ln n$, namely $5\tau_2 \ln n$.

But for any $j \in V' - J$, $E_i T_j \geq (1 - 1/\ln n) E_\rho T_j$, where ρ is the distribution after the first $5\tau_2 \ln n$ steps starting at i. By Lemma 10 and using that $\pi(j) \geq n^{-2}$, $\rho(j)/\pi(j) \geq 1 - 1/n$. Thus, $E_\rho T_j \geq (1 - 1/n) E_\pi T_j$, so $E_i T_j \geq t$. □

To see that this indeed implies an $\Omega(n \log n)$ lower bound, we show the following.

COROLLARY 12. *Suppose* $\tau_2 \leq n^{1-\delta}$, $\delta > 0$. *Then for any* $\delta' < \delta$ *and any* $v \in V$, $E_v C \geq (\delta' + o(1)) n \ln n$.

Proof. It suffices to show that for any $\varepsilon > 0$, if $\pi(i) \leq (1+\varepsilon)/n$, then $E_\pi T_i \geq (1 - o(1))n/(1+\varepsilon)$. But this follows from the result in [A4] that

$$E_\pi T_i \geq (1 - \pi(i))^2/\pi(i). \qquad \square$$

To see where the improvement comes in, we have the following result giving an $\Omega(\mu^+ \log n)$ lower bound for rapidly mixing walks on vertex transitive graphs, generalizing a similar result given for Cayley graphs in [A1].

THEOREM 13. *Define the average hitting time* $\alpha = \sum_{i,j} \pi(i)\pi(j) E_i T_j$. *Suppose* G *is vertex transitive graph and* $\tau_2 \leq n^{1-\delta}$. *Then for any* $v \in V$,

$$E_v C \geq (1 - o(1))\delta\alpha \ln n.$$

Moreover, $E_v C \leq (1 + o(1))\alpha \ln n$, *so* $E_v C = \Theta(\alpha \log n)$.

Proof. For vertex transitive graphs, $E_\pi T_i = \alpha$ for all i (see [A2]). Applying Lemma 11 with any $\varepsilon > 0$ then gives the first part. In general vertex transitive graphs, $\alpha \leq \mu^+ \leq 2\alpha$ (see [A2]). We can reduce the constant 2 in our situation by observing that if we walk for $5\tau_2 \ln n$ steps, the probability distribution on the vertices is within $(1 + o(1))$ of stationarity. Thus

$$E_i T_j \leq 5\tau_2 \ln n + (1 + o(1)) E_\pi T_j = (1 + o(1))\alpha.$$

The second part then follows from Theorem 1. □

7. Lower bound for slowly mixing walks. It is conjectured that $\Omega(n \log n)$ is a lower bound on the expected cover time of any graph. This has been proved in walks starting from the stationary distribution [A3], but it is still open for walks starting at an arbitrary vertex. We complement the above lower bound for rapidly mixing walks with one for slowly mixing walks. This improves the result in [BK], which gives the same lower bound but starting from stationarity. This leaves the general $\Omega(n \log n)$ lower bound open only for graphs with τ_2 between $n^{1-\delta}$ and $n \log n$.

Broder and Karlin prove their lower bound starting from stationarity by showing the following result.

LEMMA 14 ([BK]). *Let α be the average hitting time defined in Theorem* 13. *Then*

$$\alpha = \sum_{k=2}^{n} \tau_k, \quad \text{where } \tau_k = 1/(1-\lambda_k).$$

We improve their result by showing that the expected time to get to a random vertex is independent of the start vertex.

LEMMA 15. *For any i, $\alpha = \sum_j \pi(j) E_i T_j$.*

Our proof makes use of the following lemma.

LEMMA 16 ([KS]). *Except for a slight modification in the bipartite case, the limits*

$$Z_{ij} = \sum_{n=0}^{\infty} (P_i[X_n = j] - \pi(j))$$

exist, and are finite. Moreover,

$$E_\pi T_j = \frac{Z_{jj}}{\pi(j)},$$

$$E_i T_j = \frac{Z_{jj} - Z_{ij}}{\pi(j)}.$$

Proof of Lemma 15. Using Lemma 16,

$$\sum_j \pi(j) E_i T_j = \sum_j (Z_{jj} - Z_{ij}).$$

Rearranging summations, however, yields

$$\sum_j Z_{ij} = \sum_{n=0}^{\infty} \sum_j (P_i[X_n = j] - \pi(j)) = \sum_{n=0}^{\infty} (1-1) = 0.$$

Thus

$$\sum_j \pi(j) E_i T_j = \sum_j Z_{jj} = \sum_j \pi(j) E_\pi T_j = \alpha,$$

as required. □

We can now prove the following theorem.

THEOREM 17. *For any $v \in V$, $E_v C \geq \sum_{k=2}^{n} \tau_k$.*

Proof. Lemma 15 implies that for any v, we can pick a w with $E_v T_w \geq \alpha$. Thus $E_v C \geq E_v T_w \geq \alpha$. □

COROLLARY 18. *If $\tau_2 = \Omega(n \log n)$, then for any $v \in V$, $E_v C = \Omega(n \log n)$.*

Proof. The proof follows immediately from Theorem 17. □

Acknowledgments. The author thanks Umesh Vazirani for helpful discussions and much assistance in the writing of this paper, and David Aldous for helpful comments and advice.

REFERENCES

[A1] D. J. ALDOUS, *On the time taken by random walks on finite groups to visit every state*, Z. Wahrsch. Verw. Gebiete, 62 (1983), pp. 361–374.

[A2] ———, *Hitting times for random walks on vertex transitive graphs*, Math. Proc. Cambridge Phil. Soc., 106 (1989), pp. 179–191.

[A3] ———, *Lower bounds for covering times for reversible Markov chains and random walks on graphs*, J. Theoret. Probab., 2 (1989), pp. 91–100.

[A4] ———, *Applications of random walks on finite graphs*, in Selected Proceedings of Symposium on Applied Probability, Sheffield, 1989, I. V. Basawa and R. L. Taylor, eds., Institute of Mathematical Statistics, Hayward, CA, 1991.

[A5] ———, *Random walk covering of some special trees*, J. Math. Anal. Appl., to appear.

[A6] ———, personal communication, 1989.

[AKLLR] R. ALELIUNAS, R. M. KARP, R. J. LIPTON, L. LOVASZ, AND C. RACKOFF, *Random walks, universal traversal sequences, and the complexity of maze problems*, Proc. 20th Annual IEEE Symposium on Foundations of Computer Science, San Juan, 1979, pp. 218–233.

[AKS] M. AJTAI, J. KOMLOS, AND E. SZEMEREDI, *Deterministic simulation in* LOGSPACE, Proc. 19th Annual ACM Symposium on Theory of Computing, New York, 1987, pp. 132–140.

[BC] S. BHATT AND J. Y. CAI, *Take a walk, grow a tree*, Proc. 29th Annual IEEE Symposium on Foundations of Computer Science, 1988, pp. 469–478.

[BK] A. BRODER AND A. R. KARLIN, *Bounds on covering times*, J. Theoret. Probab., 2 (1989), pp. 101–120.

[C] J. T. COX, *Coalescing random walks and voter model consensus times on the torus*, Ann. Probab., 17 (1989), pp. 1333–1366.

[CRRST] A. K. CHANDRA, P. RAGHAVAN, W. L. RUZZO, R. SMOLENSKY, AND P. TIWARI, *The electrical resistance of a graph, and its applications to random walks*, Proc. 21st Annual ACM Symposium on Theory of Computing, Seattle, 1989, pp. 574–586.

[DFK] M. DYER, A. FRIEZE, AND R. KANNAN, *A random polynomial time algorithm for approximating the volume of convex bodies*, Proc. 21st Annual ACM Symposium on Theory of Computing, Seattle, 1989, pp. 375–381.

[DS] L. DEVROYE AND A. SBIHI, *Inequalities for random walks on trees*, Tech. Report, McGill University, Montreal, Quebec, Canada, 1989.

[JS] M. JERRUM AND A. SINCLAIR, *Conductance and the rapid mixing property for Markov chains: the approximation of the permanent resolved*, Proc. 20th Annual ACM Symposium on Theory of Computing, Chicago, 1988, pp. 235–244.

[KLNS] J. D. KAHN, N. LINIAL, N. NISAN, AND M. E. SAKS, *On the cover time of random walks in graphs*, J. Theoret. Probab., 2 (1989), pp. 121–128.

[K] J. KEILSON, *Markov Chain Models—Rarity and Exponentiality*, Springer-Verlag, Berlin, New York, 1979.

[KS] J. G. KEMENY AND J. L. SNELL, *Finite Markov Chains*, Van Nostrand, New York, 1969.

[Ma] P. C. MATTHEWS, *Covering problems for Brownian motion on spheres*, Ann. Probab., 16 (1988), pp. 189–199.

[Mo] J. W. MOON, *Random walks on random trees*, J. Austral. Math. Soc., 15 (1973), pp. 42–53.

[Z] D. ZUCKERMAN, *Covering times of random walks on bounded degree trees and other graphs*, J. Theoret. Probab., 2 (1989), pp. 147–157.

SIAM J. DISC. MATH.
Vol. 5, No. 1, pp. 88–111, February 1992
©1992 Society for Industrial and Applied Mathematics
008

BOUNDED ROUND INTERACTIVE PROOFS IN FINITE GROUPS*

LÁSZLÓ BABAI†

Abstract. This paper considers "black box groups," i.e., finite groups whose elements are uniquely encoded by strings of uniform length, with group operations being performed by a *group oracle* B. Let G, H be such groups, each given by a list of generators. It is known that the problem of *membership* in G belongs to NP^B [L. Babai and E. Szemerédi, *Proceedings of the 25th* IEEE *Symposium on the Foundation of Computer Science*, 1984, pp. 229–240]. The following problems are shown to belong to the complexity class AM^B; i.e., they possess bounded-round randomized interactive proofs (Arthur–Merlin protocols): *nonmembership* in G, the verification of the *order* of G, *isomorphism* of G and H, and checking the list of *composition factors* of G. A group oracle B is constructed, under which none of these problems belongs to NP^B, even for abelian groups.

All the results extend to "black box factor groups," i.e., groups defined as factor groups G/N, where G is a black box group, $N \triangleleft G$ is a normal subgroup, and both G and N are given by lists of generators.

A list of consequences puts verification of a large number of basic group theoretic constructions in $(\mathrm{AM} \cap co\mathrm{AM})^B$. These include homomorphisms, kernels, intersection of subgroups and cosets, membership in double cosets, centralizers, the center, cores, minimal normal subgroups, and the maximal solvable normal subgroup. A notable extension of the applicability of the results is obtained by observing that subgroups of the automorphism group of a black box group G (given by a list of generators in their action on the generators of G) can be viewed (in a nondeterministic setting) as black box groups themselves.

The results are applicable to matrix groups over finite fields and to factor groups thereof. (Matrix operations replace the group oracle.) In this case, most problems listed are conjectured to belong to NP, but a proposed approach to the proof of this statement requires detailed knowledge of the classification of finite simple groups. In contrast, the material presented here relies on the elements of group theory only (with the exception of the composition factors result). These applications provided the *original motivation for introducing the Arthur–Merlin protocols* in [L. Babai, *Proceedings of the 17th Annual* ACM *Symposium on the Theory of Computing*, 1985, pp. 421–429], where some of the results of this paper were announced.

The key to the basic results is a "local expansion lemma" for groups, which has since found applications in designing polynomial time algorithms.

Key words. interactive proofs, randomization, nondeterminism, complexity, group, expansion

AMS(MOS) subject classifications. 68Q15, 68R05, 20D60, 05C25

1. Introduction.

1.1. Black box groups. A natural general framework for the complexity theory of algorithmic problems for finite groups is the theory of *black box groups*, introduced in [BSz]. In this model, the multiplication and inversion tables of an infinite family of finite groups $B(q)$ (q is an arbitrary string over a finite alphabet) are stored by a *group oracle* B. The elements of each group $B(q)$ are uniquely encoded as strings of uniform length $|q|$. "*Black box groups*" are subgroups of the $B(q)$, given by lists of generators. Procedures over such groups rely on oracle queries to perform group operations. Because of their particular importance in concrete cases, we also consider

*Received by the editors June 7, 1988; accepted for publication (in revised form) December 14, 1990.

†Eötvös University, Budapest, Hungary 1088, and Department of Computer Science, The University of Chicago, 1100 East 58th Street, Chicago, Illinois 60637. This research was partially supported by Hungarian National Foundation for Scientific Research grant 1812 and National Science Foundation grant CCR 8710078.

"*black box factor groups*," i.e., factor groups of black box groups by their normal subgroups, also given by lists of generators.

This approach is analogous to the common practice of considering algorithmic problems for combinatorial structures such as matroids given by various oracles (independence, rank, etc.) (See [GLS] for a comprehensive treatment; cf. also [KUW]). The positive results imply oracle-free consequences for explicitly given structures (such as linear matroids, or matrix groups over finite fields in our case). The separation results, however, do not have unrelativized analogues (for good reason: they would settle problems like NP $\neq$ *co*NP).

The foremost concrete examples of groups we can treat in our context are *matrix groups over finite fields* and their factor groups. All finite simple groups of Lie type have natural representations in this form (cf. [Car]). In algebraic geometry, linear algebraic groups provide important examples [Hu]. Different kinds of groups, not easily represented as matrix groups, occur in algebraic number theory. The members of the ideal class group of the quadratic number field with negative discriminant Δ are represented by triples of integers of absolute value less than $|\Delta|$; composition is performed using arithmetic operations and greatest common divisor (g.c.d.) [BŠ], [LL], [Sch1]. For the groups of elliptic curves over finite fields (cf. [Sil], [Sch2], [LL]), group operations are defined by rational functions of the coordinates. The situation is much more complicated for the Jacobian varieties of hyperelliptic curves [Can] and more general algebraic curves [Pi]. These groups are abelian, but nonabelian group varieties (e.g., semi-abelian varieties [Fa]) have also been studied.

In these examples, group operations can be performed in polynomial time, or at least in time (polynomial in some of the parameters [Pi], although this fact or even the nature of the representation amenable for computation is often highly nontrivial).

Using polynomial time subroutines to replace oracle queries, we infer unrelativized consequences from our results on black box groups. We note that despite the impressive array of number-theory-related implementations of the black box group concept, the commutative and even the solvable cases are quite easily handled within the classical complexity classes NP and NP $\cap$ *co*NP, respectively (see [BSz]), and do not require the Arthur–Merlin framework, discussed below. So the domain where we have interesting unrelativized consequences are, still, matrix groups over finite fields and factor groups of subgroups of direct products of matrix groups.

1.2. Algorithmic problems and complexity results. The three basic algorithmic problems we discuss in this paper are *membership in*, and *order of*, a black box factor group, and *isomorphism* of two black box factor groups. We consider some more advanced questions as well, such as the *Sylow subgroups* and the *composition factors* of a black box factor group. In addition, we give a long list of consequences, including the verification of *homomorphisms, kernels, intersections of subgroups and subcosets, centralizers, cores, and the maximal solvable normal subgroup* (§12).

Since even in the case of one-by-one matrices over a finite field, the problems of membership and order seem no more tractable than discrete logarithm or factoring the integer $q-1$, in a sense the best we may expect is putting the problems listed above in NP$\cap$*co*NP. We "almost" succeed in doing so even for black box factor groups in general.

We consider the complexity class AM, a natural randomized extension of NP (or a nondeterministic version of BPP; see the definition below). We show that, relative to a group oracle B, membership in a black box factor group belongs to $(\mathrm{NP} \cap \mathit{co}\mathrm{AM})^B$; verification of the order of a black box factor group belongs to $(\mathrm{AM} \cap \mathit{co}\mathrm{AM})^B$; and,

as a consequence, isomorphism of two black box factor groups belongs to AM^B. Both Sylow subgroups and composition series can be verified in $(\mathrm{AM} \cap co\mathrm{AM})^B$. It follows that for subgroups of direct products of matrix groups over finite fields (given by a list of generators) and their factor groups, these languages belong to the corresponding unrelativized classes $\mathrm{NP} \cap co\mathrm{AM}$, $\mathrm{AM} \cap co\mathrm{AM}$, and AM, respectively. We conjecture that these problems actually belong to $\mathrm{NP} \cap co\mathrm{NP}$ (or to NP in the case of isomorphism). While a proof of this seems to be forthcoming, the proposed approach depends on detailed information on the classification of finite simple groups (cf. [BKLP]; some cases are still missing). To trade such tools and a currently unproven hypothesis for randomization was this author's main motivation behind the introduction of the class AM [Ba1] (cf. [BM]). Some of the results of this paper were announced in [Ba1].

1.3. Arthur–Merlin protocols and complexity classes. The complexity class NP can be thought of as the class of languages L, where membership in L has a short (polynomial length) proof or certificate that can be verified by a deterministic polynomial time bounded verifier. AM is a natural randomized extension of NP in which the prover is allowed to use overwhelming statistical evidence to convince the verifier. It is defined via *Arthur–Merlin protocols.* An Arthur–Merlin protocol is a *randomized interactive proof scheme* employing public coin flips. When executing the protocol, two players (Arthur, the *verifier,* and Merlin, the *prover*) receive an input string w and take turns to print strings of polynomial length. After a certain number of moves (determined by the protocol), the game terminates and a deterministic polynomial time judge evaluates the game (declares Arthur or Merlin the winner). Arthur's moves are random, Merlin's moves are optimal. It is required that Merlin's winning chances be bounded away from $\frac{1}{2}$; they cannot fall between $\frac{1}{3}$ and $\frac{2}{3}$, say. (This gap can be amplified to $[2^{-|w|}, 1-2^{-|w|}]$ and beyond, by playing the same game several times and declaring the winner of the majority of games to be the winner.) The language defined by such a protocol consists of those input strings w for which Merlin is the favored player.

The complexity class $\mathrm{AM}(t(n))$ consists of those languages recognizable by Arthur–Merlin protocols consisting of $t(|w|)$ moves on input w, with Arthur moving first. Of particular interest is the class $\mathrm{AM} := \mathrm{AM}(2)$ (a single move of Arthur followed by a single move of Merlin). All finite levels of the $\mathrm{AM}(k)$ hierarchy are known to collapse to this class: $\mathrm{AM} = \mathrm{AM}(3) = \mathrm{AM}(4) = \cdots$ [Ba1]. This result relativizes to any oracle. (More generally, if $t(n) \geq 2$, then $\mathrm{AM}(2t(n)) = \mathrm{AM}(t(n))$. This result, as well as more detailed formalism and a discussion of other related results and underlying philosophy, can be found in [BM], a companion paper to this one.)

Our aim is to put the black box group problems in *as low complexity classes as we can.* Since we succeed within AM, the higher levels of the hierarchy (such as the class $\mathrm{AM(poly)} = \bigcup_{k\geq 1} \mathrm{AM}(n^k)$) will not be our concern. Recently, extremely powerful Arthur–Merlin protocols have been constructed by Lund et al. [LFKN] and Shamir [Sh], showing that all languages in PSPACE belong to AM(poly).

There are some interesting languages known to belong to AM but not to NP. Examples include "graph nonisomorphism" [GMW], and its generalization to permutation groups, "coset disjointness" [BM]. (We generalize this result to black box groups in this paper; see §12.)

In contrast to the fact that AM(poly) = PSPACE, a strong indication of "smallness" of the class AM is that AM is unlikely to contain *co*NP. If it does, then the polynomial time hierarchy [MS], [St1] collapses to $\mathrm{AM} = \Sigma_2^P = \Pi_2^P$ [BHZ]. (This fact actually follows from the collapse AM = MAM; cf. [BM, p. 260].) Since we consider a

relativized model, it is worth pointing out that, relative to some oracle, even AM(poly) does not include *co*NP [FS].

The perception that the class AM is a very natural randomized version of NP is further enhanced by the fact that AM consists of precisely those languages that belong to NP^A for *almost every oracle* A (Nisan [N]). Here the term "almost every oracle" refers to the probability space of oracles that, on each query, respond 0 or 1 with probability $\frac{1}{2}$ independently. The set of such oracles is the product of countably many uniform 2-element spaces, endowed with the product measure. The result follows from Nisan's exciting discovery of how to drastically reduce the number of independent random bits consumed by polynomial size, constant depth randomized Boolean circuits. (This result puts AM in direct analogy with BPP, which has been known to consist of precisely those languages that belong to P^A for almost every oracle A [BG], [Ku].)

Seemingly more powerful randomized interactive proof systems have been introduced by Goldwasser, Micali, and Rackoff [GMR], the difference being that the GMR system employs *private coin flips* (hidden from the prover), and reveals to the prover a polynomial time computable function of the random bits and of the game history in each move. The equivalence of the two systems was established by Goldwasser and Sipser [GS]. Some interesting protocols are easier to describe in the GMR system [GMR], [GMW], where many protocols have been devised with the additional property of being *zero knowledge*, a property that is not expected to be transferable to Arthur–Merlin protocols [GMR, Conjecture 7.3]. The more transparent nature of Arthur–Merlin protocols has been exploited in the study of complexity classes (see [Ba1], [AGH], [BM], [BHZ], [FS], [Sa], etc.) Curiously, the recent extremely powerful interactive protocols [LFKN], [Sh] also use the public coin version (Arthur–Merlin protocols), as do the protocols in this paper (in the spirit of [Ba1], where our basic results were announced without proof).

1.4. Organization of the paper. In §2 we introduce the precise formalism of black box groups. In §3 we review one of our key tools, the *Reachability Lemma* [BSz], which establishes the existence of short straight line programs in finite groups. In §4 we state the main results of the paper and derive them from the one that asserts that verification of the *order* of black box groups belongs to AM^B (Theorem 4.2A). With the exception of §§8 and 12, we spend the rest of the paper on proving this latter result.

Verification of the order is broken into two parts: lower and upper bound verification. In §5 we define the concept of Arthur–Merlin protocols for approximate upper and lower estimation. In §6 we review a general lower-bound protocol, based on universal hashing in the spirit of [Sip], for the number of accepting computations for NP-languages. This technique is shown in §7 to yield the following intermediate result: given a black box group G and an integer m, verification of the *divisibility of the order of* G *by* m is in AM^B. (B is the group oracle.) A *separation* result is the subject of §8: we show that there exists an *abelian* group oracle B under which the nonmembership problem does not belong to NP^B (while it belongs to $(AM \cap coNP)^B$ under any group oracle). We mention that the BPP counterpart of this separation result (namely, that abelian black box group membership does not belong to BPP^B) is proven in [Ba2].

These separation results show the extent to which we have evidence that matrix groups might be easier to handle than black box groups in general (cf. [BSz, §10]): for abelian or, more generally, for solvable matrix groups (and conceivably for all matrix groups) over finite fields, the nonmembership problem, as well as verification of the

order, belong to NP.

The second half of the paper is mostly devoted to deriving an approximate upper-bound AM^B-protocol for the order of black box groups. Section 9 introduces a general principle *to estimate, from above, the number of equivalence classes*, assuming the number of classes is close to the number of elements. The elementary group theory lemmas, throughout which this technique will be used, are stated in §10. Lemma 10.2 establishes a *local expansion property of groups*, which is of independent interest in its own right. This lemma has already found other applications. In [Ba2] it is used to analyze random walks on vertex-transitive graphs and derive a polynomial time Monte Carlo algorithm to construct *nearly uniformly distributed random elements in a black box group*. In [BCFS] the local expansion lemma serves as a basic tool to guarantee early termination of an algorithm designed to manipulate permutation groups with small bases in nearly linear time.

The *approximate upper-bound protocol* is given in §11. We conclude that section with the proof of our central result, Theorem 4.2A, stating the *verifiability of the order* of black box groups.

A list of applications is given in §12. These include verification of numerous important constructions, including homomorphisms, kernels, intersections of subcosets, centralizers, cores, and the maximal solvable normal subgroup. We indicate in §13 how a subgroup of the automorphism group of a black box group can be treated as a (nondeterministic) black box group, thus expanding the range of applications of all the results stated.

Open problems are listed in §14.

2. Group oracles and black box groups. We use the following model of *oracle Turing machines*. The Turing machine is endowed with a query tape and a response tape. The machine has a query state; when it enters this state, the oracle prints (magically, at no cost) a response string on the response tape. The response is a (deterministic) function of the query; we identify the oracle with this function.

A *group oracle* B accepts queries of the form $(q, x, y,$ prod), $(q, x,$inv), and $(q,$id), where q, x, y are strings of *equal length* over a finite alphabet. The response to each of these queries is either a string of length $|q|$, which we denote $\mathrm{prod}_q(x, y)$, $\mathrm{inv}_q(x)$, and id_q, respectively, or a symbol indicating "invalid query." We omit the subscript q when its value is clear from the context.

We denote by $B(q)$ the set of those strings x of length $|q|$ for which the query $(q, x,$inv) is valid. An oracle B is a *group oracle* if it has the following properties:

1. For each q, the set $B(q)$ is either empty or it is a group under the operation prod_q.
2. The inverse of $x \in B(q)$ is $\mathrm{inv}_q(x)$.
3. The identity element of $B(q)$ is id_q.

In particular, the query $(q,$id) is valid precisely if $B(q) \neq \emptyset$.

The oracle defines the groups $B(q)$. Subgroups of these groups, defined by lists of generators, will be called *black box groups*. Factor groups of black box groups, given as G/N, by lists of generators for the black box group G and its normal subgroup N, will be called *black box factor groups*.

Remark 2.1. The role of the "invalid query" symbol is to test membership in $B(q)$. This is only necessary to check if the input is correct, i.e., the generators listed to define a black box group $G \leq B(q)$ do indeed belong to $B(q)$. None of the procedures to be described will ever make an invalid query on valid input.

Remark 2.2. The definition presented here is more restrictive than the one given in [BSz]. The difference is that we now require code words to uniquely represent group

elements. In [BSz] the possibility that different strings encode the same element of a group was also allowed. (The purpose was to enable handling factor groups of black box groups as black box groups.) In the context of the present paper, we neither need, nor are able to maintain, this greater generality. (String counting methods break down in the case of ambiguous encoding.) On the other hand, as we see in §4, we are still able to handle factor groups with no difficulty.

An extension in the direction of the generality of the [BSz] definition becomes necessary when we want to apply the results of this paper to *black box group actions on black box groups.* The simple modification of the definitions is given in §13.

Remark 2.3. Suppose that we are given a family of group oracles B_i. By querying these oracles, a "direct product oracle" B can be simulated, which, for each tuple $(q_1, \cdots, q_k)$, produces the group operations in $B_1(q_1) \times \cdots \times B_k(q_k)$. In this sense, direct products of black box groups are black box groups themselves. We rely on this fact in §§4 and 12.

3. Membership testing. The *membership* problem for black box groups is the recognition problem of the language $\{(q, x, G) : x \in G\}$ (where G is a subgroup of $B(q)$, given by a list of generators).

The problem of verifying the *order* of black box groups is to recognize the language $\{(q, G, m) : |G| = m\}$ (where m is an integer and G, as before, is a subgroup of $B(q)$).

The *isomorphism problem* for black box groups is the recognition problem of the language $\{(q_1, G_1, q_2, G_2) : G_1$ and G_2 are isomorphic$\}$. (Again, $G_i \leq B(q_i)$.)

Our concern is the complexity of these problems and their analogues for black box factor groups. The following result plays a central role here.

THEOREM 3.1 (see [BSz]). *The membership problem for black box groups relative to the group oracle B belongs to* $\mathrm{NP}^B z$.

This is an immediate consequence of Lemma 3.2 below.

A *straight line program* in a group G with respect to a set S of generators is a sequence of group elements $g_1, \cdots, g_t$ such that each member of this sequence is either a member of S, or the inverse of a preceding member, or the product of two preceding members. A straight line program is said to *construct* its members *from* S. The length of the program is t.

LEMMA 3.2 (Reachability Lemma, [BSz]). *Given a group G of order n, a set S of generators of G, and an element $g \in G$, there exists a straight line program of length less than* $(1 + \log_2 n)^2$ *constructing g from S.*

To derive Theorem 3.1 from this lemma, we observe that for a black box group $G \leq B(q)$, the order of G is at most $c^{|q|}$ (where c is the size of the alphabet). The length of the input being greater than or equal to $|q|$, the Reachability Lemma guarantees the existence of a polynomial (quadratic) length straight line program constructing $x \in G$ from the given generators of G. This program can be guessed and verified by a nondeterministic oracle Turing machine.

It follows that for $G, H \leq B(q)$ black box groups, the relation $H \leq G$ can be certified (belongs to NP^B). (It suffices to certify that every generator of H belongs to G.) If, in addition, we wish to certify that $H \triangleleft G$, it suffices to certify that $g^{-1}hg \in G$ for each generator h of H and each generator g of G (cf. [BSz, §4]).

COROLLARY 3.3. *The membership problem for black box factor groups relative to the group oracle B belongs to* NP^B.

Indeed, having certified $N \triangleleft G$, the membership $Nx \in G/N$ is equivalent to the membership $x \in G$.

4. The main results. Here are the answers to the fundamental problems.

THEOREM 4.1. *The membership problem for black box factor groups relative to the group oracle B belongs to* $(\mathrm{NP} \cap co\mathrm{AM})^B$.

THEOREM 4.2. *The problem of verifying the order of black box factor groups relative to the group oracle B belongs to* $(\mathrm{AM} \cap co\mathrm{AM})^B$.

The difficult part of this result, restricted to black box groups, will play a central role in the paper and will be stated separately as Theorem 4.2A after Theorem 4.8 below.

THEOREM 4.3. *The isomorphism problem for black box factor groups relative to the group oracle B belongs to* AM^B.

We have results on some more advanced questions as well.

THEOREM 4.4. *The verification of generators for the Sylow subgroups of black box factor groups belongs to* $(\mathrm{AM} \cap co\mathrm{AM})^B$.

Our last result concerns the verification of composition factors. In contrast to the elementary nature of the rest of the paper, the proof of this one requires the list of finite simple groups.

A group is *simple* if it has order greater than one and it has no nontrivial normal subgroups. The finite simple groups have been classified; each of them is associated with a *standard name* in the literature (see, e.g., [Car]). These names are character strings often involving two numerical parameters n and q, where n is a more or less arbitrary positive integer and q is a prime power. We adopt the convention to write n in *unary* and q in binary. This rule will also be applied to cyclic and alternating groups: the length of the name of the simple group $\mathbf{Z}_p$ (p prime) is $\log p + O(1)$, and the length of the name of the simple group $\mathrm{Alt}(n)$ ($n \geq 5$) is $n + O(1)$. The standard name of the linear group $PSL(n, q)$ is $A_{n-1}(q)$; the length of this name is $n + \log q + O(1)$.

The following fact is of importance. (Its validity depends on the convention just described.)

FACT 4.5. *Given the standard name X of a finite simple group L, a matrix representation of L over the corresponding finite field can be constructed in polynomial time.*

(Here we assume that the field itself is given explicitly; i.e., if the field in question is $\mathbf{F}_{p^m}$, then an irreducible polynomial of degree m over $\mathbf{F}_p$ is available (p a prime). Such a polynomial is known to be constructible in Las Vegas polynomial time.)

Remark 4.6. By a *matrix representation*, we mean a list of matrices that generate a group isomorphic to L.

We note that a somewhat smaller *projective* matrix representation can also be constructed in polynomial time.

A *projective* matrix representation is a representation in the form $L \cong G/(Z \cap G)$, where $G \leq GL(n, q)$ is a group of $n \times n$ matrices over the q-element field $\mathbf{F}_q$, and $Z = \{\lambda I : \lambda \in \mathbf{F}_q^\times\}$ is the group of *scalar matrices*. ($\mathbf{F}_q^\times = \mathbf{F}_q \setminus \{0\}$.) While it may not always be straightforward to find generators for $Z \cap G$, the fact that $G/(Z \cap G) \cong GZ/Z$ reduces the problem of representing these groups as factor groups of matrix groups to exhibiting generators for the cyclic group $\mathbf{F}_q^\times$. The verification of such generators is clearly in $\mathrm{NP} \cap co\mathrm{NP}$.

Fact 4.5 asserts, in particular, that neither the field nor the dimension of the matrices are large, nor is the number of generators. A more detailed statement is this: Let the standard name X of the group L consist of k characters. Then L *can be represented as a group of matrices of dimension* $O(k^2)$, *as well as a projective group of matrices of dimension* $O(k)$ *over a field of order less than or equal to* 2^k. The number of generators required is two.

In particular, L cannot be very large compared to its standard name:

$$|L| < \exp(O(k^3)).$$

The next fact shows it cannot be very small, either.

FACT 4.7. *If L is a finite simple group with standard name X, then*

$$\log |L| \geq c|X|$$

for some absolute constant $c > 0$.

There is no explicit reference for the above facts; they follow by inspection from the description and properties of the simple groups of Lie type (cf. [Car]).

A chain of subgroups $G = G_1 \geq G_2 \geq \cdots \geq G_s = 1$ is *subnormal* if each G_i is normal in G_{i-1}. A *composition series* is a subnormal chain such that each factor group G_{i-1}/G_i is simple. These factor groups are the *composition factors* of G. Up to isomorphism and order, they are uniquely associated with G.

There are now several tasks to perform. First, we may wish to verify simplicity, or, more generally, verify a composition series (each member of which is, as always, given by a list of generators). In addition, given another list of black box factor groups, we may wish to verify that those groups (in an appropriate order) are isomorphic to the composition factors of G. Third, we may want to verify the standard names of the composition factors of G. These tasks may be viewed as to include guessing the appropriate lists of groups and group names. We prove that the lists obtained are not too long (polynomially bounded in the length of the input), and all the tasks listed can be performed within $(\mathrm{AM} \cap co\mathrm{AM})^B$.

THEOREM 4.8. *Each of the following belongs to $(\mathrm{AM} \cap co\mathrm{AM})^B$, where B is a group oracle. The input is a black box factor group $G = G_0/N$ (relative to B):*

(a) *The verification of a composition series of G (possibly including guessing a composition chain),*

(b) *The verification of the standard names of the composition factors (possibly including guessing these names),*

(c) *Given G and a list of other black box factor groups, the verification that the other groups are (isomorphic to) the composition factors of G.*

(When guessing the result is part of Merlin's job, the content of the statements includes that the lengths of the guesses must be polynomially bounded.)

We show that Theorems 4.1–4.8 are close relatives. Indeed, half of Theorem 4.2, restricted to black box groups, together with Theorem 3.1, imply all the rest.

The strong part of Theorem 4.2 follows.

THEOREM 4.2A. *The problem of verifying the order of black box groups relative to the group oracle B belongs to AM^B.*

PROPOSITION 4.9. *Theorems 4.1–4.8 follow from Theorems 4.2A and 3.1.*

Proof. To verify the order of the black box factor group $G = G_0/N$, we verify the orders of G_0 and N and compute $|G| = |G_0|/|N|$. The remaining part of Theorem 4.2 is the following: to verify that the order of G is not a given number, just guess and verify the correct order.

Nonmembership verification (Theorem 4.1) follows instantly: $g \notin G$ if and only if $|G| < |\langle G, g\rangle|$.

The verification of isomorphism of two black box groups G and H (Theorem 4.3) can be reduced to order verification as follows. Let $g_1, \cdots, g_s$ be the given list of generators of G. Merlin guesses a corresponding list $h_1, \cdots, h_s$ of elements of H and

proves (by Theorem 3.1) that the h_i generate H. Let $K \leq G \times H$ denote the subgroup of the direct product generated by $(g_1, h_1), \cdots, (g_s, h_s)$. Finally, an AM^B-protocol verifies that $|G| = |H| = |K|$.

The correctness of this protocol is immediate by the following simple observation.

PROPOSITION 4.10. *The correspondence $g_i \mapsto h_i$ of generators of the finite groups G and H can be extended to an isomorphism $G \to H$ if and only if $|G| = |H| = |K|$.*

Proof. The necessity is clear. For the sufficiency, let π_1 and π_2 be the projections of K to G and H, respectively. Each projection is onto. If $|G| = |H| = |K|$, we infer that each projection is an isomorphism, and therefore $\pi_1^{-1}\pi_2$ is a $G \to H$ isomorphism. □

The following result, with a different proof, was pointed out to us by Luks.

COROLLARY 4.11 (Luks). *Isomorphism of permutation groups belongs to* NP.

Proof. Suppose that we are given a correspondence between generators of two permutation groups. The verification that this correspondence extends to an isomorphism is reduced by Proposition 4.10 to determining the order of permutation groups, a task known to belong to P [Si1], [FHL], [Kn]. □

We state the extension of Proposition 4.10 to black box factor groups as a separate lemma.

LEMMA 4.12. *Let N_i be a normal subgroup of the black box group G_i $(i = 1, 2)$. The verification of the isomorphism of the factor groups G_1/N_1 and G_2/N_2 belongs to* AM^B.

Proof. We proceed along the lines of the previous proof. Merlin guesses corresponding sets $\{g_j\}$ and $\{h_j\}$ of generators of G_i/N_i ($i = 1, 2$, respectively) (these are elements of G_i that, together with N_i, generate G_i). Now $N_1 \times N_2$ is normal in $G_1 \times G_2$. Let K denote the subgroup of $G_1 \times G_2$ generated by the pairs (g_j, h_j), together with $N_1 \times N_2$. Again, all that must be verified is that the groups G_1/N_1, G_2/N_2, and $K/(N_1 \times N_2)$ have equal order. □

Assume now that a prime number p and a subgroup H of G are given. To verify that H is a Sylow p-subgroup of G (Theorem 4.4), Merlin guesses and verifies the orders of G and H and leaves it to the judge to check that $|H|$ is the largest power of p dividing $|G|$.

The proof of Theorem 4.8 (verification of a composition chain and the composition factors) is a bit less immediate.

To verify a composition series and the composition factors of a black box group G, we first guess a composition chain $G = G_1 \triangleright G_2 \triangleright \cdots \triangleright G_d = 1$ (unless such a chain is part of the input), and verify that each subgroup is normal in its predecessor. Next, we guess the name of each factor group G_i/G_{i+1} from among the standard names of finite simple groups.

If $\{L_i\}$ are the composition factors of G, then the length of the code words for the generators of G is greater than or equal to $\log |G| = \sum_i \log |L_i|$. Consequently, we can guess the standard name X_i of each L_i and add it to the input; by Fact 4.7, this augmentation will increase the length of the input by at most a constant factor. In addition, using Fact 4.5, we may add matrix representations M_i of each L_i at a cost of polynomially increasing the length of the input.

Finally, by Lemma 4.12, we can verify (in AM^B) that M_i is isomorphic to G_i/G_{i+1}. This, in particular, will prove that these factors are simple, thus proving the AM^B claims of parts (a) and (b) of Theorem 4.8. The AM^B claim for part (c) of that theorem now follows directly from Theorem 4.3.

To extend this result to a black box factor group $G = G_0/N$, we guess a composition series of G_0, which passes through N, i.e., $N = G_j$ for some $j \leq d$. We then

perform the above protocols for $i \leq j-1$.

The $co\mathrm{AM}^B$ claims in Theorem 4.8 are now immediate. To verify that a certain strictly descending subnormal chain is *not* a composition series, Merlin inserts an additional normal subgroup and checks that all inclusions are still proper (membership and nonmembership verification). To see that the purported composition factors (listed either by their standard names or as black box factor groups) are wrong, guess and verify the right ones and compare.

Most of the remainder of the paper will be devoted to proving Theorem 4.2A. (The exceptions are §8, where a lower bound, yielding relativized separation, is derived, and §§12 and 13, where corollaries are listed.) Henceforth (with the exception of §§12 and 13), we only work with black box groups as opposed to black box factor groups.

Remark 4.13. For matrix groups over finite fields, and for factor groups of subgroups of direct products thereof, the results stated follow without relativization. (Queries to the group oracle are replaced by matrix operations.) For such groups, however, stronger results may follow, using very deep tools of group theory. Indeed, by the results of [BSz], a fairly plausible conjecture on short presentations of all finite simple groups (cf. [BKLP]) implies that *for matrix groups over finite fields*, AM *can be replaced by* NP in the theorems stated in this section.

Remark 4.14. For groups of integral matrices, the membership problem is already undecidable for 4×4 matrices [Mi].

However, finiteness of groups of integral matrices can be tested in Las Vegas polynomial time [Ba3], and all results of this paper apply to finite groups of integral matrices.

Remark 4.15. Permutation groups form a class of finite groups for which the algorithmic problems have been thoroughly investigated, and polynomial time algorithms have been found for most of the problems listed above (membership and order: [Si1], [Si2] (cf. [FHL], [Kn], [Je], [BCFLS]), composition series [Lu], Sylow subgroups [Ka]). Permutation group isomorphism belongs to NP (Luks, see Corollary 4.11 above). We list some open problems at the end of the paper.

5. Approximation protocols. As in statistical analyses, we usually must decide between two hypotheses A and B, which together exhaust all possibilities but are not necessarily mutually exclusive; i.e., they may overlap. A typical example is the estimation of a quantity f: given two real numbers $\alpha \geq \beta$, we must conclude that either $f \leq \alpha$ or $f \geq \beta$. Our conclusion should be very likely correct. (If $\beta \leq f \leq \alpha$, then either conclusion is correct.)

A typical result of this kind will have the following format: "There exists an Arthur–Merlin protocol that Merlin is likely to win if B is false, and Merlin is likely to lose if A is false." ("Likely" should mean with probability $> \frac{2}{3}$; but the certainty can then be amplified to $1-\delta$ by repeated tests and majority vote on the outcomes, where δ decreases exponentially as a function of the number of tests.)

An Arthur–Merlin protocol is an *ϵ-approximate lower-bound protocol* for the quantity $f(x) \geq 1$, if there exists $\alpha \geq 1$ such that Merlin is the likely winner if $f(x) \geq (1+\epsilon)\alpha$, and the likely loser if $f(x) \leq \alpha$.

If Merlin is able to win the majority of a large number of games defined by such a protocol, this can be viewed as an overwhelming statistical evidence in favor of the hypothesis $f(x) > \alpha$. On the other hand, if $f(x) \geq (1+\epsilon)\alpha$, then Merlin will indeed be very likely to win the majority of the games, thereby convincing Arthur of the more modest claim $f(x) > \alpha$.

We define *ϵ-approximate upper-bound protocols* analogously.

We say that *approximate lower-(upper)-bound* $\mathrm{AM}(t(n))$*-protocols exist* for a class of functions $f(x)$ if some Arthur–Merlin protocol takes an additional input μ, where μ is a tiny positive integer (*tiny* = written in unary), and turns into a $(1/\mu)$-approximate lower-(upper)-bound protocol. (As before, $t(n)$ denotes the number of moves. We speak of AM-protocols if the number of moves is two and Arthur moves first.) The *relativized versions* of these protocols allow the judge to make oracle queries.

6. Approximate lower-bound protocols. Let L be a language over the three-letter alphabet $\{0,1,\#\}$, and let c be a fixed positive integer. We assume that every word in L contains precisely one $\#$. For a string $x \in \{0,1\}^*$, let $f_{L,c}(x)$ be the number of those $y \in \{0,1\}^*$ such that $|x| \le |y| \le |x|^c$ and $x\#y \in L$.

THEOREM 6.1 (Sipser). *For any oracle B, any $c > 0$, and any $L \in \mathrm{NP}^B$, approximate lower-bound AM^B-protocols for the function $f_{L,c}(x)$ exist.*

The proof of this employs Sipser's technique [Sip] based on universal classes of hash functions described by Carter and Wegman [CW]. The same method is the main tool in the work of Goldwasser and Sipser [GS]. For completeness, we include a proof.

We note that, as long as c is fixed, by possibly padding the argument x, we may, and do, assume that $|x| = |y|$ for all (x,y) such that $x\#y \in L$. Let $L(x) = \{y : x\#y \in L\}$ and $f(x) = f_{L,1}(x)$, the purported lower bound for $|L(x)|$.

First, we consider the case when Merlin claims $L(x)$ to be *dense* in $\{0,1\}^n$ ($n = |x|$); i.e., his asserted lower bound is $f(x) > \alpha 2^n$ for some fixed positive α. In this case Arthur selects m random $(0,1)$-strings y_i of length n, and Merlin supplies a witness whenever one exists to prove that $x\#y_i \in L$. Merlin's expected number of successes will be $m|L(x)|2^{-n}$. If Merlin succeeds in at least $(1+\epsilon/2)mf(x)2^{-n}$ cases, we declare him the winner; otherwise, Arthur wins.

Let $\beta = |L(x)|2^{-n}$ and $\gamma = f(x)2^{-n} > \alpha$. Merlin is doing Bernoulli trials with probability β of success. If $f(x) \ge (1+\epsilon)|L(x)|$ then $\beta \ge (1+\epsilon)\gamma$, so the probability that the success rate is $< (1+\epsilon/2)\gamma$ decreases exponentially as a function of m and, indeed, for $m > c(\alpha)/\epsilon$, the probability will be $< \frac{1}{3}$. (Here, $c(\alpha)$ is a positive constant, depending on α.) This means that Merlin is likely to win in this case.

Similarly, if $f(x) < |L(x)|$, then $\beta < \gamma$, so the probability that the success rate is greater than or equal to $(1+\epsilon/2)\gamma$ decreases exponentially, and for $m > c(\alpha)/\epsilon$, Merlin is likely to lose.

This settles the dense case. The following lemma allows the general case to be reduced to the dense case. We identify the set $\{0,1\}^n$ with the n-dimensional space over the two-element field $\mathbf{F}_2$. Linear maps from $\{0,1\}^n$ to $\{0,1\}^k$ are represented by $k \times n$ $(0,1)$-matrices.

LEMMA 6.2. *Let $S \subseteq \{0,1\}^n$. Let $\alpha > 0$ and $2^k \ge |S|/\alpha$. Then there exists a $k \times n$ $(0,1)$-matrix C such that $|C(S)| \ge (1-\alpha)|S|$, where $C(S) \subseteq \{0,1\}^k$ is the image of S under the linear map C.*

Proof. Let us choose C randomly with uniform distribution over the 2^{kn} matrices. For $z \in \{0,1\}^n$, if $z \ne 0$ then $\mathrm{Prob}(Cz = 0) = 2^{-k}$. Therefore, for any two distinct $u, v \in \{0,1\}^n$, we have $\mathrm{Prob}(Cu = Cv) = \mathrm{Prob}(C(u-v) = 0) = 2^{-k}$. Let us call $u, v \in S$ *mates*, if $u \ne v$ and $Cu = Cv$. We conclude that for any $v \in S$, the probability that v has a mate (in S) is $\le |S|2^{-k} \le \alpha$. Therefore the expected number of mateless members of S is $\ge (1-\alpha)|S|$, and this, clearly, is a lower bound on the expected size of $C(S)$. Consequently, for some C we have $|C(S)| \ge (1-\alpha)|S|$. □

Now we return to the proof of Theorem 6.1. Given $0 < \epsilon < \frac{1}{2}$, let $\alpha = \epsilon/3$, and select k such that $2^{k-1} < (1+\epsilon)f(x)/\alpha \le 2^k$. Let $S = L(x)$.

The protocol runs as follows. Merlin exhibits a $k \times n$ $(0,1)$-matrix C that max-

imizes $|C(S)|$. Let $\delta = (1+\epsilon)(1-\alpha) - 1$. Next, a $(\delta/2)$-approximate lower-bound AM-protocol, as above, tests the claim that $|C(S)| \geq f(x)$: if $|C(S)| \geq (1+\delta/2)f(x)$, then Merlin is likely to win; if $|C(S)| < f(x)$, then Merlin is likely to lose. (Merlin demonstrates $v \in C(S)$ by exhibiting $u \in S$ such that $v = Cu$.)

If $|S| \geq (1+\epsilon)f(x)$, then S has a subset S_1 with $|S_1| = \lfloor (1+\epsilon)f(x) \rfloor$. Now $2^k \geq |S_1|/\alpha > 2^{k-2}$, hence $C(S_1)$ is dense in $\{0,1\}^k$, and we can apply Lemma 6.2 to S_1. The conclusion is that $|C(S)| \geq |C(S_1)| \geq (1-\alpha)|S_1| > (1+\delta/2)f(x)$, and Merlin is likely to win.

If $|S| < f(x)$, then $|C(S)| \leq |S| < f(x)$, so Merlin is likely to lose. □

Remark 6.3. This proof is identical to the one described in [BM, §4], except for a small bug there, corrected by the introduction of the set S_1 above. The author is indebted to the referee for pointing out the inaccuracy.

Remark 6.4. The MAM-protocol described in the proof of Theorem 6.1 can be transformed into an AM-protocol because the collapse AM = MAM [Ba1] applies to approximation protocols as well.

7. Verification of divisors of the order of a group.

THEOREM 7.1. *Let B be a group oracle and*

$$D(B) = \{(q, G, m) : G \leq B(q) \text{ and } m \text{ divides } |G|\}.$$

Then $D(B)$ belongs to AM^B.

Proof. Here is the protocol.

Merlin guesses and proves the prime factorization $m = p_1^{\alpha_1} \cdots p_s^{\alpha_s}$. For each i, Merlin guesses a subgroup $P_i \leq G$ and proves that P_i is a p_i-group (i.e., its order is a power of p_i). (By an elementary result [BSz, Prop. 5.20], this property is in NP^B.)

Finally, Arthur and Merlin perform an AM^B-protocol which Merlin is likely to win if $|P_i| \geq p_i^{\alpha_i}$; and he is likely to lose if $|P_i| \leq \frac{1}{2}p_i^{\alpha_i}$. (This requires an ϵ-approximate lower-bound protocol, with $\alpha = \frac{1}{2}p_i^{\alpha_i}$ and $\epsilon = 1$ with the notation of §5.)

The correctness of the protocol is clear: if m divides $|G|$, Merlin can find subgroups P_i of order $p_i^{\alpha_i}$ and is likely to win in the last step. If m does not divide $|G|$, then, for some i, the order of any p_i-subgroup P_i of G will be $\leq p_i^{\alpha_i - 1} \leq \frac{1}{2}p_i^{\alpha_i}$, and Merlin is likely to lose. □

8. Separation of NP and AM under a group oracle. Using the simplest case of approximate lower-bound protocols (just for dense sets, by direct sampling) it is easy to construct an oracle to separate AM from NP. (NP-machines cannot tell sets of size 2^{n-1} from sets of size 2^{n-2} among subsets of $\{0,1\}^n$, while AM-machines can.) We extend this idea by showing that group oracles are capable of achieving this separation. In fact, we show that NP-machines cannot even tell cyclic groups of order p from noncyclic groups of order p^2. An *abelian group oracle* is a group oracle B such that all groups $B(q)$ are abelian.

THEOREM 8.1. *There exists an abelian group oracle B such that the nonmembership problem for black box groups defined by B does not belong to* NP^B, *while, under any group oracle, it belongs to* $\mathrm{AM}^B \cap co\mathrm{NP}^B$. *Consequently,* NP^B *does not contain* $\mathrm{AM}^B \cap co\mathrm{NP}^B$. *In particular,* $\mathrm{NP}^B \neq \mathrm{AM}^B$.

Proof. An *elementary abelian p-group* is a product of cyclic groups of order p. The groups to be defined by the oracle will all be of a very simple kind, for each prime p, the group $B(p)$ will be an elementary abelian group of order p^2, i.e., $B(p) \cong \mathbf{Z}_p \times \mathbf{Z}_p$. Group elements will be encoded by strings of length $k = \lceil 2 \log p \rceil$. Each $B(p)$ will have a subgroup $G(p)$ given by two generators encoded 0^k and 1^k, neither of which is the identity.

Let $L(B) = \{p \mid G(p) \text{ is not cyclic}\}$. In other words, $p \in L(B)$ if and only if $|G(p)| = p^2$, if and only if 1^k is not a member of the subgroup generated by 0^k.

Since $|G(p)| \leq p^2$ in any case, all we must verify is that p^2 divides $|G(p)|$. This puts $L(B)$ in AM^B by the result of the preceding section.

We also observe that $L(B) \in co\mathrm{NP}^B$ since the nondeterministic verification of cyclicity is straightforward.

We construct B such that $L(B) \notin \mathrm{NP}^B$. The following observation will help us outwit NP-machines.

PROPOSITION 8.2. *Let $E(p)$ be an elementary abelian group of order p^2 and let S be a subset of $E(p)$. If $|S| \leq \sqrt{2p}$, then there exists a group homomorphism $\varphi : E(p) \to \mathbf{Z}_p$ which is one-to-one on S.*

Proof. The number of subgroups of order p in $E(p)$ is $p+1$, and these subgroups are pairwise disjoint (apart from the identity). Since this number is greater than $\binom{|S|}{2}$, one of these subgroups must miss all quotients ab^{-1}, where $a, b \in S$, $a \neq b$. Let P be such a subgroup. The natural homomorphism $\varphi : E(p) \to E(p)/P$ does the trick. □

COROLLARY 8.3. *Under the assumptions of Proposition 8.2, there exists a bijective map $f : E(p) \to E(p)$ such that the subgroup generated by $f(S)$ is cyclic and for any $u, v, w \in S$, if $uv = w$, then $f(u)f(v) = f(w)$.*

Proof. Let φ be as in Proposition 8.2. Let us identify $\mathrm{Im}\varphi$ with a subgroup $D \leq E(p)$. Let us extend the one-to-one map $\varphi : S \to D$ to a bijection $f : E(p) \to E(p)$. Evidently, the subgroup generated by $f(S)$ is D (cyclic of order p). Since the map f and the homomorphism φ agree on S, the second claim also follows. □

In constructing B, we proceed by the usual diagonal technique. Let $M_1, M_2, \cdots$ be an enumeration of the polynomial time bounded nondeterministic oracle Turing machines, and suppose that we have already constructed a finite segment of B such as to guarantee $L(B) \neq L(M_j^B)$ for $j < j_0$. We proceed to extending the definition of B such as to rule out $j = j_0$ in the same sense.

Set $M = M_{j_0}$. Let p be a prime large enough that the corresponding segment of B is yet totally undefined. Moreover, let

$$p > (5 \log p)^{2c}, \tag{1}$$

where n^c is the polynomial bound limiting the number of oracle queries that M is allowed to make on inputs of length n.

Let $E(p)$ be an elementary abelian group of order p^2, encoded by strings of length $k = 3 \log p$, generated by two elements encoded 0^k and 1^k. Let us pretend for a moment that $B(p) = E(p)$ and, consequently, $B(p) = G(p)$. If in this situation M rejects p, let us stick to it (set $B(p) = E(p)$), thus causing M to make the wrong decision. (If M queries other as yet undefined segments of the oracle, we fix arbitrary answers.)

Assume now that under the assumption $B(p) = E(p)$, M accepts. Let $Q_1, \cdots, Q_s$ be all the oracle queries made along an accepting computation path. Again, we can ignore all queries other than those regarding $B(p)$. The information gained from the queries can be summarized as follows:

For certain family of less than or equal to $3s$ strings e, x_i, u_j, v_j, w_j,

(i) e encodes the identity element of $B(p)$;

(ii) The strings x_i encode no element of $B(p)$;

(iii) The strings u_j, v_j, w_j encode elements of $B(p)$;

(iv) $\mathrm{prod}(u_j, v_j) = w_j$.

We may also assume that 0^k and 1^k are among the elements u_j, v_j.

CLAIM 8.4. *We can replace the group* $B(p) = (E(p),\text{prod})$ *by another (new) group* $B(p) = (E(p), \text{newprod})$ *such as to make* $G(p)$ *cyclic and still retain the validity of all statements* (i) *through* (iv).

Proof. Let S denote the set of elements $S = \{e, u_j, v_j, w_j | j = 1, \cdots\}$. By inequality (1) and Corollary 8.3, there exists a bijection $f : E(p) \to E(p)$ such that the subgroup generated by $f(S)$ is cyclic and $\text{prod}(f(u), f(v)) = f(w)$ for any $u, v, w \in S$ such that $\text{prod}(u, v) = w$. Let us now define $B(p)$ on the same underlying set $E(p)$ by setting

$$\text{newprod}(a, b) = c \text{ iff } \text{prod}(f(a), f(b)) = f(c).$$

The effect is that, on one hand, the group generated by 0^k and 1^k is now cyclic; on the other hand, all the product relations (iv) (and (i)) are inherited to "newprod." Since (ii) and (iii) remain valid by definition, the claim follows.

Redefining $B(p)$ this way will now cause M to erroneously accept p since the previous accepting path remains intact. This completes the proof of Theorem 8.1. □

9. Approximate upper bound on the number of equivalence classes. We must make some general observations before proceeding to the upper-bound protocol for the order of black box groups.

Assume that the language L consists of triples (x, y, z) such that $y, z \in \{0,1\}^*$ have length $|y| = |z| = r(x)$ for some polynomially bounded function $r(x) \leq |x|^C$, and for each x, the set $R_L(x) = \{(y, z) : (x, y, z) \in L\}$ is an equivalence relation on $\{0,1\}^{r(x)}$. We call such an L an *equivalence language*. Our aim is to give an approximate upper-bound AM^B-protocol to estimate the number $\nu_L(x)$ of equivalence classes of $R_L(x)$, assuming that $L \in \text{NP}^B$. We only succeed if the number of equivalence classes is very large: $\nu_L(x)2^{-r(x)} \geq r(x)^{-c}$ for every x and some constant c.

The fact that we do not succeed in general is not surprising since, even in the subcase when $L \in P$ and $\nu_L(x) \leq 2$, deciding that $\nu_L(x) < 2$ can be *co*NP-complete and thus a general upper bound within a factor of 2 would imply that *co*NP $\subseteq$ AM. This is an unlikely conclusion since it would imply the collapse of the polynomial time hierarchy to $\Pi_2^P = \Sigma_2^P = AM$ [BHZ] (cf. [BM, §1.9]). Approximate upper-bound protocols within an $O(2^{|x|^c})$ factor for some absolute constant c would yield the same unlikely consequence. (This follows by a direct product argument: Given an equivalence language L, consider the equivalence language L^m defined by $R_L^m(x) = \{(y_1, \cdots, y_m, z_1, \cdots, z_m) : (x, y_i, z_i) \in L\}$. Now $\nu_L^m(x) = (\nu_L(x))^m$. Hence $\nu_L(x) < 2$ if and only if $\nu_L^m(x) < 2^m$. Set $m = |x|^c$.)

PROPOSITION 9.1. (a) *For every equivalence language* $L \in P$, *the set* $\{x : \nu_L(x) = 1\}$ *belongs to coNP.*

(b) *There exists an equivalence language* $L \in P$ *such that the set* $\{x : \nu_L(x) = 1\}$ *is coNP-complete.*

Proof. (a) $\nu_L(x) \neq 1$ is witnessed by a pair (y, z) such that $(x, y, z) \notin L$.

(b) Define L by letting $(x, y, z) \in L$ if and only if x is a CNF formula with at least one nonempty clause and $x(y) = x(z)$, where $x(y)$ denotes the Boolean value obtained by substituting y (i.e., an initial segment of y) for the variables in x. Now, $\nu_L(x) = 2$ if x is satisfiable; otherwise, $\nu_L(x) = 1$. □

THEOREM 9.2. *Let* B *be any oracle,* $L \in NP^B$ *an equivalence language, and* c, ϵ *positive constants. Then there exists an* AM^B*-protocol for estimating* $\nu_L(x)$ *(the number of equivalence classes) from above in the following sense: on input* (x, k) *such that* $k2^{-r(x)} > r(x)^{-c}$, *the protocol allows Merlin to win with probability greater than* $\frac{2}{3}$ *if* $k \geq (1+\epsilon)\nu_L(x)$ *but forces Merlin to lose with probability greater than* $\frac{2}{3}$ *if* $k < \nu_L(x)$.

For the proof, we need the following observation.

LEMMA 9.3. *Let R be an equivalence relation on the finite set A. For $u \in A$, let $s(u)$ denote the size of the R-class of u. For random u (uniformly distributed over A), the random variable $\sigma = \sigma(u) = |A|/s(u)$ is an unbiased estimator of the number of R-classes.*

Proof. Clearly, each equivalence class contributes 1 to the sum

$$E(\sigma) = \sum_{u \in A} 1/s(u). \qquad \square$$

Proof of Theorem 9.2. We may assume that $\epsilon < \frac{1}{5}$. Let us fix L and let $\nu = \nu_L(x)$. Let $n = r(x)$ and $A = \{0,1\}^n$. The protocol goes as follows. For $m = \lceil 48\epsilon^{-2}n^{2c} \rceil$, Arthur picks m random members a_i of A. For each $i \leq m$, Merlin exhibits as many (say s_i) elements of A, $R(x)$-equivalent to a_i, as he can, but not more than $d = \lceil 4n^c/\epsilon \rceil$. Let $\kappa = 1/m(1/s_1 + \cdots + 1/s_m)$. Merlin wins if $\kappa < (1 - \epsilon/4)k2^{-n}$.

To prove the correctness of this protocol, let us first assume that $k \geq (1+\epsilon)\nu$. Merlin's optimal strategy is to make the s_i as large as possible. Let $s(u)$ denote the size of the $R(x)$-class of u and let $\mu = 1/m(1/s(a_1) + \cdots + 1/s(a_m))$. Then, clearly,

$$\frac{1}{s_i} = \max\left\{\frac{1}{s(a_i)}, \frac{1}{d}\right\} < \frac{1}{s(a_i)} + \frac{1}{d}.$$

Consequently, $\kappa < (1/d) + \mu$.

By Lemma 9.3, we have $E(\mu) = \nu 2^{-n}$. By assumption, $k2^{-n} > n^{-c}$.

Now the variance of $1/s(u)$ (for uniformly distributed random $u \in A$) is less than 1 (since $0 < 1/s(u) \leq 1$) and, therefore, the variance of μ is $< 1/m$. Consequently, by the Chebyshev inequality,

$$\text{(2)} \qquad \text{Prob}(\mu \geq \nu 2^{-n} + (\epsilon/3)k2^{-n}) = \text{Prob}(\mu - E(\mu) \geq (\epsilon/3)k2^{-n})$$

$$\leq \frac{\text{Var}(\mu)}{((\epsilon/3)k2^{-n})^2} < \frac{9}{m\epsilon^2}\frac{1}{(k2^{-n})^2} \leq \frac{9}{m\epsilon^2}n^{2c} \leq \frac{1}{5}.$$

Now, the probability that Merlin loses is

$$\text{(3)} \qquad \text{Prob}(\kappa \geq (1 - \epsilon/4)k2^{-n}) \leq \text{Prob}(\mu > (1 - \epsilon/4)k2^{-n} - 1/d).$$

In view of the inequality $1/d \leq (\epsilon/4)k2^{-n}$, a consequence of our choice of d and of the condition $k2^{-n} > n^{-c}$, the right-hand side of (3) is

$$\leq \text{Prob}(\mu \geq (1 - \epsilon/2)k2^{-n}) = \text{Prob}(\mu \geq (1 - 5\epsilon/6)k2^{-n} + (\epsilon/3)k2^{-n})$$

$$\leq \text{Prob}(\mu \geq (1 - 5\epsilon/6)(1+\epsilon)\nu 2^{-n} + (\epsilon/3)k2^{-n}) \leq \text{Prob}(\mu \geq \nu 2^{-n} + (\epsilon/3)k2^{-n}),$$

which, by (2), is less than $\frac{1}{5}$.

Let us now assume that $k < \nu$. Since $\kappa \geq \mu$, the probability that Merlin wins is at most

$$\text{Prob}(\mu < (1 - \epsilon/4)k2^{-n}) \leq \text{Prob}(\mu < (1 - \epsilon/4)E(\mu))$$

$$\leq \frac{\text{Var}(\mu)}{(\epsilon/4)^2 E(\mu)^2} < \frac{16}{m\epsilon^2}\frac{1}{(k2^{-n})^2} \leq \frac{16}{m\epsilon^2}n^{2c} \leq \frac{1}{3},$$

again by the Chebyshev inequality combined with the fact that $k2^{-n} < \nu 2^{-n} = E(\mu)$. □

10. Two elementary lemmas on finite groups. In this section, we describe the two elementary results in group theory on which the approximate upper-bound protocol of the next section depends.

Let G be a group and $g_1, \cdots, g_s$ be a sequence of elements of G. The *cube* generated by this sequence is the set

$$C(g_1, \cdots, g_s) = \{g_1^{\epsilon_1} \cdots g_s^{\epsilon_s} : \epsilon_i = 0, 1\}.$$

Note that this set depends on the ordering of the g_i. The sequence $g_1, \cdots, g_s$ is a sequence of *cube-generators* for G if $G = C(g_1, \cdots, g_s)$. In this case, clearly, $s \geq \log |G|$. Somewhat surprisingly, this naive bound is very nearly tight, as the following elementary result of Erdős and Rényi shows. (For a simple proof, see [BE].)

LEMMA 10.1 (see [ER]). *Let G be a finite group of order m. Then G has a sequence of s cube-generators, where $s < \log m + \log \ln m + 3$.*

(log and ln denote base 2 and base e logarithms, respectively.)

We note that the exponent strings $\epsilon_1 \cdots \epsilon_s$ provide a fairly economical encoding of the elements of G; the "average redundancy" will be $2^s/m < 8 \ln m$. The main results of this paper critically depend on the fact that the right-hand side is bounded by a polynomial of $\ln m$.

The following lemma establishes a *local expansion property* of groups. Since writing this paper, several algorithmic applications of this lemma have been found [Ba2], [BCFS]. The lemma generalizes to vertex-transitive graphs, i.e., graphs with a transitive group of automorphisms [Ba2].

LEMMA 10.2 (Local expansion of groups). *Let S denote a set of generators of the group G, and set $T = S \cup S^{-1} \cup \{1\}$. Let D be any subset of T^t, the set of t-term products of members of T (in any order). Let, finally, $0 < \alpha \leq 1/(2t+1)$ be such that*

$$|D| \leq (1 - 2\alpha t)|G|. \tag{4}$$

Then for at least one generator $g \in S$,

$$|D - Dg| \geq \alpha |D|. \tag{5}$$

Proof. For a contradiction, suppose that (5) fails for every $g \in S$. The fact that S generates G means that $G = \bigcup_{k \geq 0} T^k$.

Let us observe that for each $g \in S$, $|D - Dg^{-1}| = |Dg - D| = |D - Dg| \leq \alpha |D|$. Observing, in addition, that

$$D - Dxy \subseteq (D - Dy) \cup (D - Dx)y,$$

it follows by induction on k that for any $u \in T^k$, we have $|D - Du| < k\alpha |D|$.

As long as $k\alpha < 1$, this implies that $u \in D^{-1}D$. Since $\alpha \leq 1/(2t+1)$, it follows that $T^{2t+1} \subseteq D^{-1}D \subseteq T^{2t}$ and therefore $T^{2t} = T^{2t+1} = \cdots = G$.

Next we observe that for any $u \in G$, the number of $x \in D$ such that $xu \in D$ is greater than $(1 - 2\alpha t)|D|$. This is the case because $u \in T^{2t}$ and thus $|D - Du| < 2\alpha t |D|$.

Consequently, the number of pairs (x, u) such that $x \in D$ and $xu \in D$ is greater than $(1 - 2\alpha t)|G||D|$. On the other hand, the number of such pairs is precisely $|D|^2$. Hence $(1 - 2\alpha t)|G||D| < |D|^2$, contradicting assumption (4). □

11. The approximate upper-bound protocol.

THEOREM 11.1. *Approximate upper-bound AM^B-protocols for the order of black box groups exist.*

Proof. Let $0 < \epsilon < 1$. We construct an MAM^B-protocol that, given a black box group G and an integer k_0, allows Merlin to win with probability greater than $\frac{2}{3}$ if $k_0 \geq (1+\epsilon)|G|$, but forces him to lose with probability greater than $\frac{2}{3}$ if $k_0 < |G|$.

(As always, G is given by a list of generators, belonging to the group $B(q)$ defined by the oracle B. The reference to the identifier q and to the group $B(q)$ will henceforth be omitted.)

The protocol.

1. Merlin selects positive integers k and t such that $k \leq k_0$ and $\log k \leq t < \log k + \log \ln k + 3$ and elements $g_1, \cdots, g_t \in G$ (supposedly a short sequence of cube-generators of G), along with a short proof (via the Reachability Lemma) that $g_1, \cdots, g_t$ generate the group G (in the traditional sense). We note that $k \leq 2^t < 8k \ln k$.

Let C denote the cube generated by the g_i. The following three steps are performed concurrently.

2. Arthur and Merlin perform an AM^B-protocol that Merlin is likely to win if $|C| \geq k/2$ and is likely to lose if $|C| < k/4$.

3. Arthur and Merlin perform an AM^B-protocol that Merlin is likely to win if $|C| \leq (1-\epsilon/2)k$ and is likely to lose if $|C| > (1-\epsilon/4)k$.

4. Arthur and Merlin perform an AM^B-protocol that Merlin is likely to win if for each $i = 1, \cdots, t$,

$$C = Cg_i,$$

and Merlin is likely to lose if $|C| \geq k/4$, and for some $i, 1 \leq i \leq t$,

$$|C - Cg_i| \geq (\epsilon/8t)|C|.$$

Merlin is the winner if he wins each step.

(By the phrase "likely to win," we mean a chance greater than 0.9, say.)

Before discussing the implementation of steps 2–4, we show that the protocol does yield the desired approximate upper-bound verification.

CLAIM. *The above MAM^B-protocol accomplishes the objective stated in the first paragraph of the proof.*

Proof. Assume first that $k_0 \geq (1+\epsilon)|G|$. Let $(1+\epsilon)|G| \leq k \leq \min\{k_0, 2|G|\}$. In this case, Merlin can correctly select cube-generators for G in step 1, yielding a cube C such that $k/2 \leq |C| = |G| \leq k/(1+\epsilon) < (1-\epsilon/2)k$. Merlin is thus likely to win steps 2 and 3. Since now $C = Cg_i = G$ for each i, Merlin is likely to win step 4 as well.

Assume now that $k_0 < |G|$ and therefore $k < |G|$. In step 1, Merlin selects some cube $C \subseteq G$. Assume Merlin is not "likely to lose" any one of steps 2–4; i.e., he has greater than or equal to 0.1 chance in each case. It follows that $k/4 \leq |C| < (1-\epsilon/4)k$ and $|C - Cg_i| < (\epsilon/8t)|C|$ for each $i = 1, \cdots, t$. This last condition allows us to apply Lemma 10.2 with $S = \{g_1, \cdots, g_t\}$, $D = C$, and $\alpha = \epsilon/(8t)$. We conclude that

$$|G| \leq |C|/(1-2\alpha t) \leq |C|/(1-\epsilon/4) < k < |G|,$$

a contradiction. This completes the proof of the claim. □

Now we turn to the implementation of steps 2–4.

Step 2 requires an approximate lower-bound protocol as discussed in §6.

Let $A = \{0,1\}^t$. The map $\varphi : A \to G$ defined by $\varphi(u) = g_1^{u_1} \cdots g_t^{u_t}$ ($u = (u_1, \cdots u_t)$) determines the equivalence relation $\ker\varphi$ on A (u and v are equivalent if $\varphi(u) = \varphi(v)$). The number of equivalence classes is $|C|$.

To complete step 3, we invoke Theorem 9.2. The language L will consist of the triples (x, u, v), where $\varphi(u) = \varphi(v)$. (The input string x specifies the instance of the black box group and the elements $g_1, \cdots, g_t$. Therefore $|x| > t = r(x) = |u| = |v|$.) Now $|C| = \nu_L(x)$. If t is chosen as required in step 1, then $k2^{-t} > 1/(8 \ln k) > 1/(6t)$. The density condition of Theorem 9.2 is therefore met with any $c > 1$. (Observe that we have made full use of the fact that the error term in Lemma 10.1 is of order $O(\log\log|G|)$.) We can thus apply Theorem 9.2 with our $\epsilon/4$ playing the role of ϵ and $(1 - \epsilon/4)k$ in the role of k.

To implement step 4, Arthur selects a large number of random strings $u \in \{0,1\}^t$ and requires Merlin to exhibit, for each i, some $v = v(u,i) \in \{0,1\}^t$ such that $\varphi(u) = \varphi(v)g_i$. Merlin wins if he is able to exhibit such $v(u,i)$ for every u generated by Arthur. "Large" means about $1000t^2/\epsilon$.

If $C = Cg_i$ for each i, Merlin is a sure winner. We must prove that Merlin is likely to lose if for some i,

$$|C - Cg_i| \geq (\epsilon/8t)|C|.$$

This is less straightforward than it may seem since the elements $\varphi(u)$ are not uniformly distributed in C. Let $\alpha = \epsilon/8t$. The probability that Merlin fails for a random u; i.e., $\varphi(u) \in C - Cg_i$, is at least

$$|C - Cg_i|2^{-t} \geq \alpha|C|2^{-t} \geq \alpha|C|/(6tk) \geq \alpha/(24t) = \epsilon/(192t^2).$$

(The necessity to use the inequality $|C| \geq k/4$ along the way explains the role of step 2.) It follows that $1000t^2/\epsilon$ random strings u suffice to make it likely for Merlin to fail at least once. □

Proof of the main results. The main results are stated in §4. As we have shown there, we only need to prove Theorem 4.2A, which asserts that the order of a black box group can be verified. To verify that the order of the black box group G is m we require two AM^B-protocols: one that verifies that m divides $|G|$ (Theorem 7.1), and another that makes Merlin likely to win if $|G| \leq m$ and likely to lose if $|G| \geq 2m$ (Theorem 11.1). □

12. Corollaries. In this section we show a number of basic constructions of group theory to be verifiable in AM^B, where B is the group oracle.

First, we recall from [BSz] that the verification of the *normal closure* of a subgroup of a black box factor group is in NP^B. It follows that the verification of the commutator chain, as well as the descending central series, and hence the verification of solvability and nilpotence are in $(\mathrm{NP} \cap co\mathrm{NP})^B$. We should mention that much stronger results in this direction are now available; the normal closure of a subgroup in a black box group can be computed in Monte Carlo polynomial time, and, consequently, solvability and nilpotence can be *disproved* in Monte Carlo polynomial time [BCFLS]; so these two properties belong to the complexity class coR^B.

COROLLARY 12.1. *The following questions regarding black box factor groups can be reduced in polynomial time to verification of the orders of black box groups:*

(i) *homomorphisms,*

(ii) *kernels,*

The verification of

(iii) *minimal normal subgroups can be reduced to the verification of orders by a nondeterministic polynomial time reduction.*

Therefore, these problems belong to the class $(\mathrm{AM} \cap co\mathrm{AM})^B$, *and they belong to* $(\mathrm{NP} \cap co\mathrm{NP})^B$, *assuming the conjecture that order verification belongs to* NP^B.

Proof. (i) Let G and H be black box factor groups, and let φ be a mapping of the set S of generators of G into H. Following the idea of the proof of Proposition 4.10, we observe that φ extends to a homomorphism precisely if $|G| = |\langle (g, \varphi(g)) : g \in S \rangle|$.

(ii) To verify the kernel of φ observe that $\mathrm{Ker}(\varphi) = N$ if and only if φ takes the generators of N to the identity, and $|N| = |G|/|\mathrm{Im}(\varphi)|$.

(iii) Let M be a normal subgroup of G. If it is not minimal, show a smaller normal subgroup. To verify that it is minimal, represent M as $T_1 \times \cdots \times T_k$, where the T_i are isomorphic simple groups. (Simplicity and isomorphism are reduced to "order": guess the standard name of the simple group, establish isomorphism via (i) above; cf. the discussion in §4.) Now we distinguish two cases according to whether T_1 is abelian. If T_1 is nonabelian, we establish that G acts transitively on the set $\{T_1, \cdots, T_k\}$. If T_1 is abelian, then M is a vector space that we can construct explicitly, along with the action of G on it. What we must show now is that this action is irreducible. This can actually be done in polynomial time, according to a result of Rónyai [Ro2].

The rest of this section will explicitly exploit the approximate lower-bound protocol, so the results to be stated will not be placed in NP (or *co*NP, respectively), even assuming that "order" belongs to NP. A particularly important case in point is coset disjointness, a problem not known to belong to NP even in the permutation group setting (where it generalizes GRAPH-NONISOMORPHISM). We put this problem in AM^B next. We remark that GRAPH-NONISOMORPHISM was put in AM by Goldreich, Micali, and Wigderson [GMW]; their result was generalized to coset disjointness in permutation groups in [BM]. Below, we do not use the [GMW] protocol.

COROLLARY 12.2. *Verification of the following objects associated with a black box factor group* G *is in* $(\mathrm{AM} \cap co\mathrm{AM})^B$, *where* B *is the group oracle:*

(a) *The intersection of two subgroups* $H, K \leq G$.
(b) *The order of a double coset* HsK, $H, K \leq G$, $s \in G$.
(c) *Disjointness of double cosets* HsK *and* HtK.
(d) *Disjointness of subcosets* Hs *and* Kt.
(e) *The core of a subgroup* H, *i.e., the largest normal subgroup of* G *contained in* H.

Proof. (a) Having guessed the subgroup $D = H \cap K$ and verified $D \leq H \cap K$, we only need to verify an approximate upper bound on the order of $H \cap K$. The identity

$$|H \cap K| \cdot |HK| = |H| \cdot |K|$$

reduces this to the verification of an approximate lower bound on the size of the set HK, which can be accomplished using the hashing technique (Theorem 6.1). We note that as a by-product, the exact order of the double coset HK has also been verified. This settles (b) in view of the identity

$$|HsK| = |s^{-1}Hs \cdot K|.$$

(c) It is known that the double cosets HsK and HtK either coincide or they are disjoint. If they coincide, all we need to verify is that $t \in HsK$, which can be done in NP^B. If they are disjoint, we either verify that $|HsK| \neq |HtK|$, or else an approximate lower-bound verification for the union of these two double cosets reveals that it is substantially larger than HsK, say.

(d) If the two subcosets Hs and Kt intersect, this can be verified in NP^B by exhibiting a common element. To verify disjointness, let $D = \{(g, g) : g \in G\}$ be the

diagonal subgroup of $G \times G$. Then the subcosets Hs and Kt are disjoint if and only if the following double cosets in $G \times G$ are disjoint: $(H \times K) \cdot D$ and $(H \times K) \cdot (s,t) \cdot D$.

(e) The core of H is the intersection of all conjugates of H. Suppose that we want to verify that the subgroup C is the core of H. First, we verify that $C \leq H$ and $C \lhd G$. Hence C is contained in the core. Next, we observe that every intersection of subgroups is an intersection of $\leq n$ subgroups (since n is an upper bound on the length of any subgroup chain). Therefore it suffices to represent C as the intersection of $\leq n$ conjugates of H. □

COROLLARY 12.3. *Verification of the following objects associated with a black box factor group G is in $(\mathrm{AM} \cap co\mathrm{AM})^B$, where B is the group oracle:*

(f) *The centralizer of an element.*

(g) *The centralizer of a subgroup.*

(h) *The center.*

(i) *Conjugacy of two elements.*

(j) *The maximal solvable normal subgroup of G.*

(k) *The nonabelian part of the socle of G (i.e., the product of all nonabelian minimal normal subgroups).*

Proof. (f) Let $C_G(x)$ be the centralizer of $x \in G$, and let x^G denote the set of conjugates of x in G. Then $|C_G(x)||x^G| = |G|$. Hence, to verify $L = C_G(x)$ for some $L \leq G$, we check (in polynomial time) that $L \leq C_G(x)$, and we get an approximate upper bound on $|C_G(x)|$ by obtaining an approximate lower bound on $|x^G|$, using the hashing technique.

(g) The centralizer of H is the intersection of the centralizers of its generators.

(h) A special case of (g).

(i) The proof is similar to item (c) in the previous corollary. Let $x, y \in G$. Then x^G and y^G either coincide or they are disjoint. By item (f) we know $|x^G|$ and $|y^G|$ and proceed to verifying their disjointness as for (c).

(j) and (k). Let M be the largest solvable normal subgroup of G and suppose that we want to verify $L = M$ for some $L \leq G$. First, we verify that L is solvable and normal. Next we exhibit the socle S of G/L and verify that it is the product of nonabelian minimal normal subgroups of G/L, using item (iii) of Corollary 12.1. This will simultaneously guarantee that $L = M$ and S is indeed the socle of G/M, as claimed. □

13. Automorphism groups: nondeterministic black box groups. In our nondeterministic setting, it would have been more natural to follow [BSz] in defining the black box as a device that does not itself perform the group operations, just serves to verify the results if some witness has been provided. For instance, instead of performing multiplication, it would accept or reject quadruples (x, y, z, w), where $|x| = |y| = |z| = n$ and $|w| = n^c$. If x, y, z encode valid group elements, then the quadruple (x, y, z, w) is accepted for some w if and only if $x \cdot y = z$. (So, w is a witness of the fact that $x \cdot y = z$. No witness exists if z is not the product.) Similarly, the verification of the inverse, and even the verification of the identity element requires a witness. (We have omitted the group identifier q.)

All the NP^B and AM^B results stated remain valid under this "*nondeterministic black box*" condition: the prover will always guess the product and provide a witness to support the guess.

Another difference in the [BSz] model is that there, several strings are allowed to encode the same group element.

We must exercise care when adopting this model. For our statistical considerations to remain valid, we need to postulate the encoding to be *uniform* in the sense that each group element is encoded by the same number of strings, and this number has to come certified with the group box.

Let the term "*uniformly encoded nondeterministic black box groups*" or "UNB groups" refer to the two circumstances in its name, as defined above.

All the AM^B and NP^B results of this paper extend to UNB groups.

This model would have enabled us to treat factor groups without mentioning them: the Reachability Lemma ensures that factors groups of UNB groups are themselves UNB groups. (To make this step, nondeterminism is necessary, even for the verification of the identity element, and uniformity of the encoding will automatically hold: the coset Ng will be encoded by any of its elements.)

The range of applicability of our result is significantly extended by the following observation.

PROPOSITION 13.1. *If G is a* UNB *group and H is a subgroup of* $\mathrm{Aut}(G)$, *then H is a* UNB *group.*

Proof. We must construct the group box for H. Elements of $\mathrm{Aut}(G)$ will be represented by their action on a given set S of generators of G. So if each element of G is represented by k strings, then each element of $\mathrm{Aut}(G)$ is represented by $k^{|S|}$ strings. The Reachability Lemma ensures that given an action on G in this way, the image of any group element can be verified. Therefore compositions and inverses can also be verified. □

We note that we are unable to verify generators for the *full* group of automorphisms. Proposition 13.1 makes claims about subgroups of $\mathrm{Aut}(G)$, given by generators. For instance, the center of such a subgroup, or the *kernel of the action of a group on another group,* can be verified by AM-protocols.

14. Discussion and open problems. We have seen that the order of a black box factor group can be verified by an AM-protocol, and this central result implies that a series of other objects associated with black box groups can also be similarly verified.

Perhaps the most important omission on our list of consequences is *normalizers.*

PROBLEM 14.1. *Does there exist an* AM*-protocol to verify the normalizer of a subgroup in a matrix group over a finite field?*

A positive answer would lead to another series of corollaries, including verification of the fitting subgroup.

AM-verification of maximal subgroups and full automorphism groups seem to be open even for permutation groups.

PROBLEM 14.2. *Do there exist* AM*-protocols to verify* (a) *maximality of a subgroup of a permutation group, or of a matrix group over a finite field;* (b) *the full automorphism group of a permutation group, or of a matrix group over a finite field?*

We have shown that *isomorphism* of black box factor groups belongs to AM. We can show that this problem belongs to *co*AM as well [BKL].

We should mention that some of the problems considered here have polynomial time solutions for permutation groups and their factor groups [KL]. Notable examples are composition series [Lu], Sylow subgroups [Ka], [KL], kernels of actions, and cores [KL]. Notable exceptions are centralizers and intersections.

In contrast, the list of polynomial time (Monte Carlo) algorithms for matrix groups over finite fields is very short.

An interesting example is Rónyai's polynomial time algorithm to determine the

centralizer of a matrix group within $SL(d,q)$ [Ro3]. That result was motivated by the problem of verifying centralizers, proposed in an earlier version of this paper.

Another example is a Monte Carlo test by Neumann and Praeger [NP] to decide whether a given set of matrices generates $SL(d,q)$.

The Neumann-Prager algorithm works under the heuristic assumption that uniformly distributed random elements of the group generated are available. In the polynomial time paradigm, such an assumption is justified in [Ba2] in great generality: a polynomial time Monte Carlo algorithm is given to construct nearly uniformly distributed random elements in *any black box group*.

Another sequence of algorithmic results encompassing all black box groups is given in [BCFLS]. In that paper, a Monte Carlo algorithm is given, which reduces in polynomial time the number of generators of a black box group to $O(n)$ generators. This allows repeated application of polynomial time methods of construction of certain subgroups without the danger of an exponential blowup of the number of generators (due to our inability to perform membership tests). Another polynomial time Monte Carlo algorithm from [BCFLS] constructs *normal closures*. Consequently, the commutator chain and the descending central series can be constructed in Monte Carlo polynomial time. This puts solvability and nilpotence of black box groups into *coR*. The results extend to black box factor groups G/N, except in some cases such as the solvability test and nilpotence tests, where membership test in N is also required.

Luks has recently announced deterministic polynomial time algorithms to test solvability and nilpotence of matrix groups over finite fields.

A problem of great significance is to extend these algorithmic results to include a wider class of matrix group problems.

Finally, we state a natural extension of the basic problems we have considered.

PROBLEM 14.3. *Does membership/nonmembership in matrix groups over finite fields have bounded round* zero knowledge *proof protocols* [GMR]?

Acknowledgments. The author is grateful to Gene Luks, Mike Sipser, Lajos Rónyai, and Avi Wigderson for insightful comments, and to an anonymous referee for pointing out some inaccuracies.

REFERENCES

[AGH] W. AIELLO, S. GOLDWASSER, AND J. HÅSTAD, *On the power of interaction*, Combinatorica, 10 (1990), pp. 3–26.

[Ba1] L. BABAI, *Trading group theory for randomness*, in Proc. 17th Annual ACM Sympos. on the Theory of Computing, Providence, RI, 1985, pp. 421–429.

[Ba2] ———, *Local expansion of vertex-transitive graphs and random generation in finite groups*, in Proc. 23rd ACM Sympos. on Theory of Computing, New Orleans, LA,1991, pp. 164–174.

[Ba3] ———*Deciding finiteness of matrix groups in Las Vegas polynomial time*, in Proc. 3rd ACM–SIAM Sympos. on Discrete Algorithms, Orlando, FL, 1992, to appear.

[BCFLS] L. BABAI, G. COOPERMAN, L. FINKELSTEIN, E. M. LUKS, AND A. SERESS, *Fast Monte Carlo algorithms for permutation groups*, in Proc. 23rd ACM Symp. on Theory of Computing, New Orleans, LA, 1991, pp. 90–100.

[BCFS] L. BABAI, G. COOPERMAN, L. FINKELSTEIN, AND A. SERESS, *Nearly linear time algorithms for permutation groups with a small base*, in Proc. ISSAC '91, Internat. Sympos. on Symbolic and Algebraic Computing, Bonn, Germany, 1991, pp. 200–209.

[BE] L. BABAI AND P. ERDŐS, *Representation of group elements as short products*, in Theory and Practice of Combinatorics, A. Rosa et al., eds., Ann. Discrete. Math., 12 (1982), pp. 27–30.

[BKL] L. BABAI, S. KANNAN, AND E. M. LUKS, *Bounded-round interactive proof for nonisomorphism of matrix groups*, in preparation.

[BKLP] L. Babai, W.M. Kantor, E.M. Luks, and P.P. Pálfy, *Short presentations for finite groups*, in preparation

[BM] L. Babai and S. Moran, *Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes*, J. Comput. Sys. Sci., 36 (1988), pp. 254–276.

[BSz] L. Babai and E. Szemerédi, *On the complexity of matrix group problems* I, in Proc. 25th IEEE Sympos. on the Foundation of Computer Science, 1984, pp. 229–240.

[BG] C. H. Bennett and J. Gill, *Relative to a random oracle A*, $P^A \neq NP^A \neq coNP^A$ *with probability* 1, SIAM J. Comput., 10 (1981), pp. 96–113.

[BHZ] R. Boppana, J. Håstad, and S. Zachos, *Does coNP have short interactive proofs?*, Inform. Process. Lett., 25 (1987), pp. 127–132.

[BŠ] Z. I. Borevič and I. R. Šafarevič, *Zahlentheorie*, Birkhäuser, Basel, 1966.

[Can] D. G. Cantor, *Computing in the Jacobian of a hyperelliptic curve*, Math. Comp., 48 (1987), pp. 95–101.

[Car] R. Carter, *Simple Groups of Lie Type*, John Wiley, London, 1972.

[CW] J. L. Carter and M. N. Wegman, *Universal classes of hash functions*, J. Comput. Sys. Sci., 18 (1979), pp. 143-154.

[CS] G. Cornell and J. H. Silverman, eds., *Arithmetic Geometry*, Springer-Verlag, Berlin, New York, 1986.

[ER] P. Erdős and A. Rényi, *Probabilistic methods in group theory*, J. Analyse Math., 14 (1965), pp. 127–138.

[Fa] G. Faltings, *Finiteness theorems for abelian varieties over number fields*, in Arithmetic Geometry, G. Cornell and J. H. Silverman, eds., Springer-Verlag, Berlin, New York, 1986, pp. 9–27.

[FS] L. Fortnow and M. Sipser, *Are there interactive protocols for co–NP languages?*, Inform. Process. Lett., 28 (1988), pp. 249–251.

[FR] K. Friedl and L. Rónyai, *Polynomial time solutions of some problems in computational algebra*, in Proc. 17th Annual ACM Sympos. on the Theory of Computing, Providence, RI, 1985, pp. 153–162.

[FHL] M. L. Furst, J. Hopcroft, and E. M. Luks, *Polynomial time algorithms for permutation groups*, in Proc. 25th IEEE Sympos. on the Foundation of Computer Science, 1980, pp. 36-41.

[Gi] J. Gill, *Computational complexity of probabilistic Turing machines*, SIAM J. Comput., 6 (1977), pp. 675–695.

[GMW] O. Goldreich, S. Micali, and A. Wigderson, *Proofs that yield nothing but their validity and a methodology of cryptographic protocol design*, in Proc. 27th IEEE Sympos. on the Foundation of Computer Science, 1986, pp. 174–187.

[GMR] S. Goldwasser, S. Micali, and C. Rackoff, *The knowledge complexity of interactive proofs*, in Proc. 17th ACM Sympos. on the Theory of Computing, Providence, RI, 1985, pp. 291-304. Expanded version: SIAM J. Comput., 18 (1989), pp. 186–208.

[GS] S. Goldwasser and M. Sipser, *Private coins versus public coins in interactive proof systems*, in Randomness and Computation, S. Micali, ed., Advances in Computing Research, Vol. 5, Jai Press, Greenwich, CT, 1989, pp. 73-90.

[GLS] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, North-Holland, Amsterdam, 1988.

[Ha] M. Hall, Jr. *The Theory of Groups*, Macmillan, New York, 1959.

[Hu] J. E. Humphreys, *Linear Algebraic Groups*, Graduate Texts in Math., 21, Springer-Verlag, Berlin, New York, 1975.

[Je] M. R. Jerrum, *A compact representation for permutation groups*, J. Algorithms, 7 (1986), pp. 60–78.

[Ka] W. M. Kantor, *Sylow's theorem in polynomial time*, J. Comput. Sys. Sci., 30 (1985), pp. 359-394.

[KL] W. M. Kantor and E. M. Luks, *Computing in quotient groups*, in Proc. 22nd Annual ACM Sympos. on the Theory of Computing, Baltimore, MA, 1990, pp. 524-534.

[KUW] R. M. Karp, E. Upfal, and A. Wigderson, *The complexity of parallel computation on matroids*, in Proc. 26th IEEE Sympos. on the Foundation of Computer Science, 1985, pp. 541–550.

[Kn] D. E. Knuth, *Efficient representation of perm groups*, Combinatorica, 11 (1991), pp. 33–43.

[Ku] S. A. Kurtz, *A note on randomized polynomial time*, SIAM J. Comput., 16 (1987), pp. 852–853.

[LL] A. K. LENSTRA AND H. W. LENSTRA, *Algorithms in number theory*, in Handbook of Theoretical Computer Science, Chap. 12, J. van Leeuwen, A. Meyer, M. Nivat, M. Paterson, and D. Perrin, eds., Elsevier, Amsterdam, and M. I. T. Press, Cambridge, MA, 1990

[Lu] E. M. LUKS, *Computing the composition factors of a permutation group in polynomial time*, Combinatorica, 7 (1987), pp. 87–99.

[LFKN] C. LUND, L. FORTNOW, H. KARLOFF, AND N. NISAN, *Algebraic methods for interactive proof systems*, in Proc. 31st IEEE Sympos. on the Foundation of Computer Science, St. Louis, MO, 1990, pp. 2–10.

[MS] A. R. MEYER AND L. J. STOCKMEYER, *The equivalence problem for regular expressions with squaring requires exponential time*, in Proc. 13th IEEE Ann. Sympos. Switching and Automata Theory, College Park, MD, 1972, pp. 125–129.

[Mi] K. A. MIHAILOVA, *The occurrence problem for direct products of groups*, Dokl. Acad. Nauk SSSR, 119 (1958), pp. 1103–1105 and Mat. Sbornik (N.S.), 70, 112 (1966), pp. 241–251.

[Mil] J. S. MILNE, *Jacobian varieties*, in Arithmetic Geometry, G. Cornell and J. H. Silverman, eds., Springer-Verlag, Berlin, New York, pp. 167–212.

[NP] P. M. NEUMANN AND C. E. PRAEGER, *A recognition algorithm for the special linear groups*, manuscript, 1990.

[N] N. NISAN, *Pseudorandom bits for constant depth circuits*, Combinatorica, 11 (1991), pp. 63–70.

[Pi] J. PILA, *Counting points on curves in polynomial time*, manuscript, 1987.

[Ro1] L. RÓNYAI, *Simple algebras are difficult*, in Proc. 19th ACM Sympos. on the Theory of Computing, New York, NY, 1987, pp. 398–408.

[Ro2] ———, *Computing the structure of finite algebras*, J. Symbolic Comput., 9 (1990), pp. 355–373.

[Ro3] ———, *Computing the order of centralizers in linear groups*, Inform. Comput., 91 (1991), pp. 172–176.

[Sa] M. SÁNTHA, *Relativized Arthur–Merlin versus Merlin–Arthur games*, Inform. Comput., 80, 1 (1989), pp. 44–49.

[Sch1] R. J. SCHOOF, *Quadratic fields and factorization*, in Computational Methods in Number Theory, H. W. Lenstra, Jr. and R. Tijdeman, eds., Math. Centre Tracts 154/155, Mathematisch Centrum, Amsterdam, 1982, pp. 235–286.

[Sch2] ———, *Elliptic curves over finite fields and the computation of square roots mod p*, Math. Computs., 44 (1985), pp. 483–494.

[Sh] A. SHAMIR, *IP=PSPACE*, in Proc. 31st IEEE Sympos. on the Foundation of Computer Science, St. Louis, MO, 1990, pp. 11–15.

[Sil] J. SILVERMAN, *The Arithmetic of Elliptic Curves*, Graduate Texts in Math., 106, Springer-Verlag, Berlin, New York, 1986.

[Si1] C. C. SIMS, *Computational methods in the study of permutation groups*, in Computational Problems in Abstract Algebra, J. Leech, ed., Pergamon Press, Elmsford, NY, 1970, pp. 169–183.

[Si2] ———, *Some group theoretic algorithms*, in Lecture Notes in Math., 697, Springer-Verlag, Berlin, New York, 1978, pp. 108–124.

[Sip] M. SIPSER, *A complexity theoretic approach to randomness*, in Proc. 15th Annual ACM Sympos. on the Theory of Computing, Boston, MA, 1983, pp. 330-335.

[St1] L. R. STOCKMEYER, *The polynomial time hierarchy*, Theoret. Comput. Sci., 3 (1976), pp. 1–22.

[St2] ———, *The complexity of approximate counting*, in Proc. 15th Annual ACM Sympos. on the Theory of Computing, Providence, RI, 1983, pp. 118–126.

SIAM J. DISCRETE MATH.
Vol. 5, No. 1, pp. 112–116, February 1992
© 1992 Society for Industrial and Applied Mathematics
009

DISJOINT PATHS IN A PLANAR GRAPH—A GENERAL THEOREM*

GUOLI DING†, A. SCHRIJVER‡, AND P. D. SEYMOUR§

Abstract. Let $D = (V, A)$ be a directed planar graph, let $(r_1, s_1), \cdots, (r_k, s_k)$ be pairs of vertices on the boundary of the unbounded face, let $A_1, \cdots, A_k$ be subsets of A, and let H be a collection of unordered pairs from $\{1, \cdots, k\}$. Given are necessary and sufficient conditions for the existence of a directed $r_i - s_i$ path P_i in (V, A_i) (for $i = 1, \cdots, k$), such that P_i and P_j are vertex-disjoint whenever $\{i, j\} \in H$.

Key words. disjoint paths, trees, planar graph

AMS(MOS) subject classifications. 05C35, 05C38, 05C70

1. Introduction.

Let $D = (V, A)$ be a directed graph, let $(r_1, s_1), \ldots, (r_k, s_k)$ be pairs of vertices of D, let $A_1, \cdots, A_k$ be subsets of A, and let H be a collection of unordered pairs from $\{1, \cdots, k\}$. We are interested in the conditions under which there exist directed paths $P_1, \cdots, P_k$ so that

(1)
(i) P_i is a directed $r_i - s_i$ path in (V, A_i) $(i = 1, \cdots, k)$;

(ii) P_i and P_j are vertex-disjoint for each $\{i, j\} \in H$.

In §3 we will discuss some special cases of this problem.

Since the problem is NP-complete, we may not expect a nice set of necessary and sufficient conditions characterizing the existence of paths satisfying (1). The problem is NP-complete even if we restrict the problem to instances with $k = 2, A_1 = A_2 = A$, and $H = \{\{1, 2\}\}$. Moreover, it is NP-complete when restricted to $A_1 = \cdots = A_k = A$, H is the collection of all pairs from $\{1, \cdots, k\}$, and D arises from an undirected planar graph by replacing each edge by two opposite arcs.

In this paper we give necessary and sufficient conditions for the problem when

(2) D is planar and the vertices $r_1, s_1, \cdots, r_k, s_k$ all belong to the boundary of one fixed face I.

The characterization extends the one given by Robertson and Seymour [1]. In fact, if (2) holds, there is an easy, greedy-type algorithm for finding the path P_i, as we discuss below.

Let D be embedded in the plane $\mathbb{R}^2$. We identify D with its image in the plane. Without loss of generality, we may assume I to be the unbounded face. (Each face is considered as an *open* region.) Moreover, we may assume that the boundary $\mathrm{bd}(I)$ of I is a simple closed curve. This is no restriction, since we can extend D by new arcs as long as we do not include them in any A_i and as long as we keep $r_1, s_1, \cdots, r_k, s_k$ on $\mathrm{bd}(I)$.

We say that two pairs (r, s) and (r', s') of vertices on $\mathrm{bd}(I)$ *cross* if each $r - s$ curve in $\mathbb{R}^2 \setminus I$ intersects each $r' - s'$ curve in $\mathbb{R}^2 \setminus I$. Clearly, the following is a

* Received by the editors June 4, 1990; accepted for publication (in revised form) November 8, 1990.

† Rutgers Center for Operations Research, Rutgers University, New Brunswick, New Jersey 08903.

‡ Mathematisch Centrum, Kruislaan 413, 1098 SJ Amsterdam, the Netherlands.

§ Bellcore, 445 South Street, Morristown, New Jersey 07960.

necessary condition for the existence of paths satisfying (1):

(3) *cross-freeness condition*: if $\{i,j\} \in H$ then (r_i, s_i) and (r_j, s_j) do not cross.

Now the following algorithm finds paths as in (1) if (2) holds. First, check if the cross-freeness condition is satisfied. If not, our problem has no solution. If the cross-freeness condition is satisfied, choose a pair (r_i, s_i) so that the shortest of the two $r_i - s_i$ paths along $\mathrm{bd}(I)$ is as short as possible (over all $i = 1, \cdots, k$). Without loss of generality, $i = k$. Let Q be this shortest $r_k - s_k$ path along $\mathrm{bd}(I)$. If (V, A_k) does not contain any $r_k - s_k$ path, then there are no paths satisfying (1). If (V, A_k) does contain an $r_k - s_k$ path, let P_k be the (unique) directed $r_k - s_k$ path in (V, A_k) that is nearest to Q. Next, repeat the algorithm for $D, (r_1, s_1), \cdots, (r_{k-1}, s_{k-1})$, removing from any A_i with $\{i,k\} \in H$ all those arcs incident with some vertex in P_k. After at most k iterations we either find paths as required, or we find that no such paths exist.

The correctness of the algorithm follows from the following observation. Suppose that there exist paths $Q_1, \cdots, Q_k$ as required. Then, if k is as above, we may assume without loss of generality that Q_k is equal to P_k. Indeed, $Q_1, \cdots, Q_{k-1}, P_k$ also form a solution, since if P_k intersects some Q_i, then also Q_k intersects Q_i.

We describe a second necessary condition. Let C be some curve in $\mathbb{R}^2$, starting in I and ending in some face F. Let $f(C)$ and $l(C)$ denote the first and last point of intersection of C with D. Let $i_1, \cdots, i_n$ be indices from $\{1, \cdots, k\}$ such that

(4)

(i) $f(C), r_{i_1}, s_{i_1}, \ldots, r_{i_n}, s_{i_n}$ are all distinct;

(ii) The $r_{i_j} - s_{i_j}$ part of $\mathrm{bd}(I)$ containing $f(C)$ is contained in the $r_{i_{j+1}} - s_{i_{j+1}}$ part of $\mathrm{bd}(I)$ containing $f(C)$, for $j = 1, \cdots, n-1$;

(iii) $\{i_j, i_{j+1}\} \in H$ for $j = 1, \cdots, n-1$.

For each $j = 1, \cdots, n$ we define a set W_j as follows. If $f(C), r_{i_j}, s_{i_j}$ occur clockwise around $\mathrm{bd}(I)$, W_j is the set of points p on D traversed by C such that some arc in A_{i_j} is entering C at p from the left and some arc in A_{i_j} is leaving C at p from the right. Similarly, if $f(C), r_{i_j}, s_{i_j}$ occur counterclockwise around $\mathrm{bd}(I)$, W_j is the set of points p on D traversed by C such that some arc in A_{i_j} is entering C at p from the right, and some arc in A_{i_j} is leaving C at p from the left.

We say that C *fits* $i_1, \cdots, i_n$ if there exist distinct points $p_1, \cdots, p_n$ so that $p_j \in W_j$ for $j = 1, \cdots, n$ and so that C traverses $p_1, \cdots, p_n$ in this order. Now we have the following condition:

(5) *cut condition*: each curve C starting and ending in I fits each choice of $i_1, \cdots, i_n$ satisfying (4), whenever $(f(C), l(C))$ crosses each (r_{i_j}, s_{i_j}) $(j = 1, \cdots, n)$.

2. The theorem. We now prove the following theorem.

THEOREM. *Let $D = (V, A)$ be a directed planar graph, embedded in the plane $\mathbb{R}^2$, let $(r_1, s_1), \cdots, (r_k, s_k)$ be pairs of vertices of D on $\mathrm{bd}(I)$, with $r_i \neq s_i$ for $i = 1, \cdots, n$, let $A_1, \cdots, A_k$ be subsets of A, and let H be a set of unordered pairs from $\{1, \cdots, k\}$.*

Then there exist paths $P_1, \cdots, P_k$ satisfying (1) *if and only if the cross-freeness condition* (3) *and the cut condition* (5) *hold.*

Proof. Necessity of the conditions is trivial. To see sufficiency, we assume without loss of generality that the arcs on $\mathrm{bd}(I)$ do not belong to any A_i. (We can add new arcs to D (but not to any A_i), without violating the cross-freeness and cut conditions.)

Choose an arbitrary point p_0 on $\mathrm{bd}(I)$, not being a vertex of D. For each $i = 1, \cdots, k$, let Q_i be that of the two $r_i - s_i$ parts of $\mathrm{bd}(I)$ that does not contain p_0. For each $i = 1, \cdots, k$, let $\mathcal{F}_i$ be the set of faces $F \neq I$ of D for which there exists a curve C starting in I and ending in F, such that $f(C) \in Q_i$, and such that C does not fit some choice of $i_1, \cdots, i_n$ satisfying (4) with $i_n = i$.

Note that, since no arc on $\mathrm{bd}(I)$ belongs to A_i, each arc in Q_i is on the boundary of $\bigcup \mathcal{F}_i$. Let B_i be the set of arcs on the boundary of $\bigcup \mathcal{F}_i$ but not in Q_i. We show that

(6) B_i is contained in A_i and contains a directed $r_i - s_i$ path.

Assume without loss of generality that r_i, p_0, s_i occur in this order clockwise around $\mathrm{bd}(I)$. Let a be an arc on the boundary of $\bigcup \mathcal{F}_i$ and not in Q_i. We show that a belongs to A_i and that a is oriented clockwise with respect to $\bigcup \mathcal{F}_i$.

Let a separate faces $F \in \mathcal{F}_i$ and $F' \notin \mathcal{F}_i$. By definition of $\mathcal{F}_i$, there exists a curve C starting in I and ending in F, such that $f(C) \in Q_i$ and such that C does not fit some choice $i_1, \cdots, i_n$ satisfying (4) with $i_n = i$. Now extend C to F' by crossing a, obtaining a curve C'.

If C' does not fit $i_1, \cdots, i_n$, then $F' = I$ (as $F' \notin \mathcal{F}_i$). Then, however, C' violates the cut condition.

So C' does fit $i_1, \cdots, i_n$. Since C itself does not fit $i_1, \cdots, i_n$, this implies that a belongs to A_i and that a is oriented clockwise with respect to $\bigcup \mathcal{F}_i$. This proves (6).

Choose for each $i = 1, \cdots, k$ a directed $r_i - s_i$ path P_i in B_i. We finally show that if $\{i, j\} \in H$, then P_i and P_j are vertex-disjoint. Assume without loss of generality that $i = 1, j = 2$, and let $\{1, 2\} \in H$. Suppose some vertex v is traversed both by P_1 and P_2. Hence v is incident with some face F_1 in $\mathcal{F}_1$ and with some face F_2 in $\mathcal{F}_2$. It follows that there exists a curve C from I to F_1 such that $f(C) \in Q_i$ and such that C does not fit indices $i_1, \cdots, i_n$ satisfying (4) with $i_n = 1$.

By the cross-freeness condition, we know that parts Q_1 and Q_2 of $\mathrm{bd}(I)$ are either contained in each other or are disjoint.

First, assume that they are contained in each other, say $Q_1 \subseteq Q_2$. Then each face $F' \neq I$ incident with v is contained in $\mathcal{F}_2$. To see this, we can extend curve C via v to F', yielding curve C'. As C does not fit $i_1, \ldots, i_n = 1$, it follows that C' does not fit $i_1, \cdots, i_n = 1, i_{n+1} = 2$. So $F' \in \mathcal{F}_2$. As this holds for each face $F' \neq I$ incident with v, no arc incident with v belongs to B_2, and hence P_2 does not traverse v.

Next, assume that Q_1 and Q_2 are disjoint. (So p_0 is in between Q_1 and Q_2.) Since F_2 belongs to $\mathcal{F}_2$, there exists a curve C' from I to F_2 not fitting indices $i'_1, \cdots, i'_{n'}$ satisfying (4) (adapted to $C', i'_1, \cdots, i'_{n'}$), such that $f(C') \in Q_2$ and such that $i'_{n'} = 2$.

Connect the curves C and C' by a $F_1 - F_2$ curve via v, yielding a curve C'' from I to I. Then C'' does not fit $i_1, \cdots, i_n, i'_{n'}, \cdots, i'_1$, as we can easily check. This violates the cut condition. □

The theorem can be seen to give a "good characterization."

3. Special cases. In this section we describe some special cases of the problem and the theorem.

First, let $G = (V, E)$ be an undirected planar graph, embedded in $\mathbb{R}^2$. Let $\{r_1, s_1\}, \cdots, \{r_k, s_k\}$ be pairs of vertices of G, each on the boundary of the unbounded face I of G. Robertson and Seymour [1] proved that there exist pairwise vertex-disjoint

paths $P_1, \cdots, P_k$ in G where P_i connects r_i and s_i for $i = 1, \cdots, k$, if and only if no two of the pairs $\{r_i, s_i\}$ cross and each vertex cut of G contains at least as many vertices as it separates pairs from $\{r_1, s_1\}, \cdots, \{r_k, s_k\}$.

This follows trivially from our theorem by replacing each arc by two opposite arcs, and taking for H the collection of all pairs from $\{1, \cdots, k\}$.

The second special case generalizes the first. Let $G = (V, E)$ be an undirected planar graph, embedded in $\mathbb{R}^2$. Let $R_1, \cdots, R_t$ be pairwise disjoint sets of vertices of G, all on the boundary of the unbounded face I of G.

We say that two sets R and R' of vertices on the boundary of I *cross* if some pair of vertices in R crosses some pair of vertices in R'. We say that a cut *separates* a set R of vertices if the cut separates $\{r, s\}$ for some r, s in R.

Robertson and Seymour [1] proved more generally that there exist pairwise vertex-disjoint trees $T_1, \cdots, T_t$ in G such that T_i covers R_i $(i = 1, \cdots, t)$ if and only if no two of the R_i cross, and each vertex cut of G contains at least as many vertices as it separates sets from $R_1, \cdots, R_t$.

This follows from the theorem by replacing each edge of G by two opposite edges, by taking as pairs $(r_1, s_1), \cdots, (r_k, s_k)$ all pairs (r, s) for which there exists an $i \in \{1, \cdots, t\}$ such that $r, s \in R_i$, and by taking for H all pairs $\{j, j'\}$ from $\{1, \cdots, k\}$ for which $r_j, s_j, r_{j'}$, and $s_{j'}$ do not all belong to the same set among $R_1, \cdots, R_t$. (We take each A_j to be equal to the full arc set.)

As a third special case, consider a planar directed graph $D = (V, A)$ and a collection of ordered pairs $(r_1, s_1), \cdots, (r_k, s_k)$ on the boundary of the unbounded face I (with $r_i \neq s_i$ for $i = 1, \cdots, k$). Then the theorem implies that there exists a directed $r_i - s_i$ path P_i for $i = 1, \cdots, k$ so that $P_1, \cdots, P_k$ are pairwise vertex-disjoint if and only if no two of the (r_i, s_i) cross, and for each cut C not intersecting any of $r_1, s_1, \cdots, r_k, s_k$, the following *cut condition* holds:

(7) If C separates $(r_{i_1}, s_{i_1}), \cdots, (r_{i_n}, s_{i_n})$, in this order, then C contains vertices $p_1, \cdots, p_n$, in this order so that for each $j = 1, \cdots, n$:

- if r_{i_j} is at the left-hand side of C, then at least one arc of D is entering C at p_j from the left and at least one arc of D is leaving C at p_j from the right;
- if r_{i_j} is at the right-hand side of C, then at least one arc of D is entering C at p_j from the right and at least one arc of D is leaving C at p_j from the left.

This follows by taking for H the set of all pairs from $\{1, \cdots, k\}$ and taking each A_i equal to A.

More generally, let $D = (V, A)$ be a planar directed graph, let $R_1, \cdots, R_t$ be sets of vertices on the boundary of the unbounded face I of D, and let, for each $i = 1, \cdots, k$, r_i be some vertex from R_i. The theorem gives necessary and sufficient conditions for the existence of pairwise vertex-disjoint rooted trees $T_1, \cdots, T_k$ in D, where T_i has root r_i and covers R_i $(i = 1, \cdots, k)$. Again this follows straightforwardly with reductions like the above.

Finally, let $D = (V, A)$ be a planar directed graph and let $R_1, \cdots, R_k$ be sets of vertices on the boundary of the unbounded face I of G. Again, it is straightforward to derive necessary and sufficient conditions for the existence of pairwise vertex-disjoint strongly connected subgraphs $D_1, \cdots, D_k$ such that D_i covers R_i (for $i = 1, \cdots, k$).

REFERENCES

[1] N. ROBERTSON AND P. D. SEYMOUR, *Graph minors* VI. *Disjoint paths across a disc*, J. Combin. Theory Ser. B, 41 (1986), pp. 115–138.

SIAM J. DISCRETE MATH.
Vol. 5, No. 1, pp. 117–126, February 1992
© 1992 Society for Industrial and Applied Mathematics
010

ON WELL-PARTIAL-ORDER THEORY AND ITS APPLICATION TO COMBINATORIAL PROBLEMS OF VLSI DESIGN*

MICHAEL R. FELLOWS† AND MICHAEL A. LANGSTON‡

Abstract. The existence of decision algorithms with low-degree polynomial running times for a number of well-studied graph layout, placement, and routing problems is nonconstructively proved. Some were not previously known to be in $\mathcal{P}$ at all; others were only known to be in $\mathcal{P}$ by way of brute force or dynamic programming formulations with unboundedly high-degree polynomial running times. The methods applied include the recent Robertson–Seymour theorems on the well-partial-ordering of graphs under both the minor and immersion orders. The complexity of search versions of these problems is also briefly addressed.

Key words. nonconstructive proofs, polynomial-time complexity, well-partially-ordered sets

AMS(MOS) subject classifications. 68C25, 68E10, 68K05

1. Introduction.

Practical problems are often characterized by fixed-parameter instances. In the VLSI domain, for example, the parameter may represent the number of tracks permitted on a chip, the number of processing elements to be employed, the number of channels required to connect circuit elements, or the load on communications links. In fixing the value of such parameters, we help focus on the physically realizable nature of the system rather than on the purely abstract aspects of the model.

In this paper, we employ and extend Robertson–Seymour poset techniques to prove low-degree polynomial-time decision complexity for a variety of fixed-parameter layout, placement, and routing problems, dramatically lowering known time-complexity upper bounds. Our main results are summarized in Table 1, where n denotes the number of vertices in an input graph and k denotes the appropriate fixed parameter. (At the referee's urging, we also list relevant, previously published results from [5], [8], as noted in the rightmost column of the table.)

In the next section, we survey the necessary background from graph theory and graph algorithms that makes these advances possible. Sections 3–5 describe our results on several representative types of decision problems, illustrating a range of techniques based on well-partially-ordered sets. In §6, we discuss how self-reducibility can be used to bound the complexity of search versions of these problems. A few open problems and related issues are briefly addressed in the final section.

2. Background.

Except where explicitly noted otherwise, all graphs that we consider are finite and undirected. A graph H is less than or equal to a graph G in the *minor* order, written $H \leq_m G$, if and only if a graph isomorphic to H can be obtained from G by a series of these two operations: taking a subgraph and contracting an edge. For example, the construction depicted in Fig. 1 shows that $W_4 \leq_m Q_3$.

* Received by the editors January 5, 1990; accepted for publication (in revised form) March 13, 1991. A preliminary version of a portion of this paper was presented at the Fifth MIT Conference on Advanced Research in VLSI held in Cambridge, Massachusetts in March 1988.

† Department of Computer Science, University of Victoria, Victoria, British Columbia, Canada V8W 2Y2. This author's research was supported in part by National Science Foundation grant MIP-8919312 and by Office of Naval Research contract N00014-88-K-0456.

‡ Department of Computer Science, University of Tennessee, Knoxville, Tennessee 37996-1301. This author's research was supported in part by National Science Foundation grant MIP-8919312 and by Office of Naval Research contract N00014-88-K-0343.

TABLE 1
Main results.

General problem area	Problem	Best previous upper bound	Our result
circuit layout	GATE MATRIX LAYOUT	open	$O(n^2)$ [5]
Linear arrangement	MIN CUT LINEAR ARRANGEMENT	$O(n^{k-1})$	$O(n^2)$
	MODIFIED MIN CUT	$O(n^k)$	$O(n^2)$
	TOPOLOGICAL BANDWIDTH*	$O(n^k)$	$O(n^2)$ [8]
	VERTEX SEPARATION	$O(n^{k^2+2k+4})$	$O(n^2)$
Circuit design and utilization	CROSSING NUMBER*	open	$O(n^3)$ [8]
	MAX LEAF SPANNING TREE	$O(n^{2k+1})$	$O(n^2)$
	SEARCH NUMBER	$O(n^{2k^2+4k+8})$	$O(n^2)$
Embedding and routing	2-D GRID LOAD FACTOR	open	$O(n^2)$
	BINARY TREE LOAD FACTOR	open	$O(n^2)$
	DISK DIMENSION	open	$O(n^3)$ [5]
	EMULATION	open	$O(n^3)$ [8]

* Input restricted to graphs of maximum degree three.

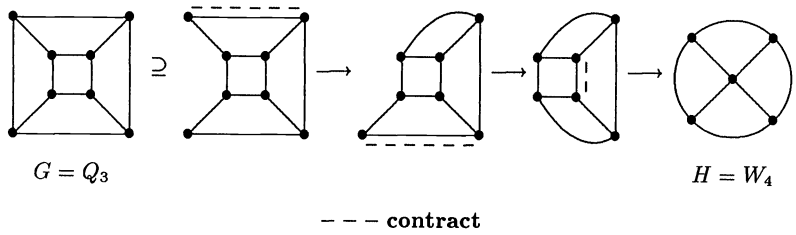


FIG. 1. *Construction demonstrating that W_4 is a minor of Q_3.*

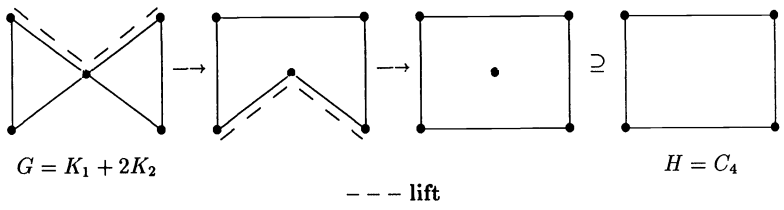


FIG. 2. *Construction demonstrating that C_4 is immersed in $K_1 + 2K_2$.*

Note that the relation $\leq_m$ defines a partial ordering on graphs. A family F of graphs is said to be *closed* under the minor ordering if the facts that G is in F and that $H \leq_m G$ together imply that H must be in F. The *obstruction set* for a family F of graphs is the set of graphs in the complement of F that are minimal in the minor ordering. Therefore, if F is closed under the minor ordering, it has the following characterization: G is in F if and only if there is no H in the obstruction set for F such that $H \leq_m G$.

THEOREM 2.1 (see [25]). *Graphs are well-partially-ordered*[1] *by $\leq_m$.*

THEOREM 2.2 (see [24]). *For every fixed graph H, the problem that takes as input a graph G and determines whether $H \leq_m G$ is solvable in polynomial time.*

Theorems 2.1 and 2.2 guarantee the *existence* of a polynomial-time decision algorithm for any minor-closed family of graphs, but do not provide any details of what that algorithm might be. Moreover, no proof of Theorem 2.1 can be entirely constructive. For example, there can be no systematic method of computing the finite obstruction set for an arbitrary minor-closed family F from the description of a Turing machine that precisely accepts the graphs in F [9].

An interesting feature of Theorems 2.1 and 2.2 is the low degree of the polynomials bounding the decision algorithms' running times (although the constants of proportionality are enormous). Letting n denote the number of vertices in G, the time required to recognize F is $O(n^3)$. If F excludes a planar graph, then F has bounded tree-width [22] and the time complexity decreases to $O(n^2)$.

A graph H is less than or equal to a graph G in the *immersion* order, written $H \leq_i G$, if and only if a graph isomorphic to H can be obtained from G by a series of these two operations: taking a subgraph and lifting[2] a pair of adjacent edges. For example, the construction depicted in Fig. 2 shows that $C_4 \leq_i K_1 + 2K_2$ (although $C_4 \not\leq_m K_1 + 2K_2$).

The relation $\leq_i$, like $\leq_m$, defines a partial ordering on graphs with the associated notions of closure and obstruction sets.

THEOREM 2.3 (see [21]). *Graphs are well-partially-ordered by $\leq_i$.*

[1] A partially-ordered set $(X, \leq)$ is *well-partially-ordered* if (1) any subset of X has finitely many minimal elements and (2) X contains no infinite descending chain $x_1 \geq x_2 \geq x_3 \geq \ldots$ of distinct elements.

[2] A pair of adjacent edges uv and vw, with $u \neq v \neq w$, is *lifted* by deleting the edges uv and vw and adding the edge uw.

The proof of the following result is original, although it has been independently observed by others as well [20].

THEOREM 2.4. *For every fixed graph H, the problem that takes as input a graph G and determines whether $H \leq_i G$ is solvable in polynomial time.*

Proof. Letting k denote the number of edges in H, we replace $G = \langle V, E \rangle$ with $G' = \langle V', E' \rangle$, where $|V'| = k|V| + |E|$ and $|E'| = 2k|E|$. Each vertex in V is replaced in G' with k vertices. Each edge e in E is replaced in G' with a vertex and $2k$ edges connecting this vertex to all of the vertices that replace e's endpoints. We can now apply the disjoint-connecting paths algorithm of [24], since it follows that $H \leq_i G$ if and only if there exists an injection from the vertices of H to the vertices of G' such that each vertex of H is mapped to some vertex in G' that replaces a distinct vertex from G, and such that G' contains a set of k vertex-disjoint paths, each one connecting the images of the endpoints of a distinct edge in H. □

Theorems 2.3 and 2.4, like Theorems 2.1 and 2.2, only guarantee the existence of a polynomial-time decision algorithm for any immersion-closed family F of graphs. The method we use in proving Theorem 2.4 yields an obvious time bound of $O(n^{h+6})$, where h denotes the order of the largest graph in F's obstruction set. (There are $O(n^h)$ different injections to consider; the disjoint-paths algorithm takes cubic time on G', a graph of order at most n^2.) Thanks to the next theorem of Mader, however, we find that the bound immediately reduces to $O(n^{h+3})$ because the problem graphs of interest permit only a linear number of distinct edges.

THEOREM 2.5 (sec [14]). *For any graph H there exists a constant c_H such that every simple graph $G = \langle V, E \rangle$ with $|E| > c_H|V|$ satisfies $G \geq_i H$.*

We show in §4 that by exploiting excluded-minor knowledge on immersion-closed families the time complexity for determining membership can, in many cases, be reduced to $O(n^2)$.

3. Exploiting the minor order. Given a graph G of order n, a *linear layout* of G is a bijection ℓ from V to $\{1, 2, \cdots, n\}$. For such a layout ℓ, the *vertex separation* at location i, $s_\ell(i)$, is $|\{u : u \in V, \ell(u) \leq i$, and there is some $v \in V$ such that $uv \in E$ and $\ell(v) > i\}|$. The vertex separation of the entire layout is $s_\ell = \max\{s_\ell(i) : 1 \leq i \leq n\}$, and the vertex separation of G is $vs(G) = \min\{s_\ell : \ell$ is a linear layout of $G\}$.

Given both G and a positive integer k, the $\mathcal{NP}$-complete VERTEX SEPARATION problem [13] asks whether $vs(G)$ is less than or equal to k. It has previously been claimed that VERTEX SEPARATION can be decided in $O(n^{k^2+2k+4})$ time [4], and is thus in $\mathcal{P}$ for any fixed value of k. We now prove that the problem can be solved in time bounded by a polynomial in n, the degree of which does not depend on k.

THEOREM 3.1. *For any fixed k,* VERTEX SEPARATION *can be decided in $O(n^2)$ time.*

Proof. Let k denote any fixed positive integer. We show that the family F of "yes" instances is closed under the minor ordering. To do this, we must prove that if $vs(G) \leq k$ then $vs(H) \leq k$ for every $H \leq_m G$. Without loss of generality, we assume that H is obtained from G by exactly one of these three actions: deleting an edge, deleting an isolated vertex, or contracting an edge.

If H is obtained from G by deleting an edge, then $vs(H) \leq vs(G) \leq k$ because the vertex separation of any layout of G either remains the same or decreases by 1 with the removal of an edge. If H is obtained from G by deleting an isolated vertex, then, also clearly, $vs(H) \leq k$.

Suppose that H is obtained from G by contracting the edge uv. Let ℓ denote a

layout of G whose vertex separation does not exceed k and assume that $\ell(u) < \ell(v)$. We contract uv to u in the layout ℓ' of H as follows: we set $\ell'(x) = \ell(x)$ if $\ell(x) < \ell(v)$ and set $\ell'(x) = \ell(x) - 1$ if $\ell(x) > \ell(v)$. Let us consider the effect of this action on the vertex separation at each location of the layout. Clearly, $s_{\ell'}(i) = s_\ell(i)$ for $1 \leq i < \ell(u)$. If there exists a vertex w with $\ell(w) > \ell(u)$ and either $uw \in E$ or $vw \in E$, then $s_{\ell'}(\ell(u)) \leq s_\ell(\ell(u))$. Otherwise, $s_{\ell'}(\ell(u)) \leq s_\ell(\ell(u)) - 1$. Similar arguments establish that $s_{\ell'}(i) \leq s_\ell(i)$ for the ranges $\ell(u) < i < \ell(v)$ and $\ell(v) \leq i < n$. Therefore, the vertex separation of ℓ' does not exceed k and $vs(H) \leq k$.

We conclude that, in any case, H is in F, and hence F is minor-closed. It remains only to note that there are trees with arbitrarily large vertex separation (such an excluded planar graph ensures bounded tree-width, and thus a time complexity of $O(n^2)$). □

Given a graph G and a positive integer k, the $\mathcal{NP}$-complete SEARCH NUMBER problem [19] asks whether k searchers are sufficient to ensure the capture of a fugitive who is free to move with arbitrary speed about the edges of G, with complete knowledge of the location of the searchers. More precisely, we say that every edge of G is initially *contaminated*. An edge $e = uv$ becomes *clear* either when a searcher is moved from u to v (v to u) while another searcher remains at u (v), or when all edges incident on u (v) except e are clear and a searcher at u (v) is moved to v (u). (A clear edge e becomes recontaminated if the movement of a searcher produces a path without searchers between a contaminated edge and e.) The goal is to determine if there exists a sequence of *search steps* that results in all edges being clear simultaneously, where each such step is one of the following three operations: (1) place a searcher on a vertex, (2) move a searcher along an edge, or (3) remove a searcher from a vertex. It has been reported that SEARCH NUMBER is decidable in $O(n^{2k^2+4k+8})$ time [4]. As has been independently noted by Papadimitriou [18], however, minor-closure can be applied to reduce this bound.

THEOREM 3.2. *For any fixed k,* SEARCH NUMBER *can be decided in $O(n^2)$ time.*

Proof. The proof is straightforward by showing that, for fixed k, the family of "yes" instances is closed under the minor ordering and by observing that there are excluded trees. □

Consider next the $\mathcal{NP}$-complete MAX LEAF SPANNING TREE problem [11]. Given a connected graph G and a positive integer k, this problem asks whether G possesses a spanning tree in which k or more vertices have degree one. This problem can be solved by brute force in $O(n^{2k+1})$ time. (There are $\binom{n}{k}$ ways to select k leaves and $O(n)$ possible adjacencies to consider at each leaf. For each of these $O(n^{2k})$ candidate solutions, the connectivity of the remainder of G can be determined in linear time because there can be at most a linear number of edges.) Although this means that MAX LEAF SPANNING TREE is in $\mathcal{P}$ for any fixed k, we seek to exploit minor-closure so as to ensure a low-degree polynomial running time.

THEOREM 3.3. *For any fixed k,* MAX LEAF SPANNING TREE *can be decided in $O(n^2)$ time.*

Proof. Let k denote any fixed positive integer. Consider the proper subset of the "no" instances, the family F of graphs, none of whose connected components has a spanning tree with k or more leaves. F is clearly closed under the minor ordering, from which the theorem follows because we need only test an input graph for connectedness and nonmembership in F. □

4. Exploiting the immersion order. An *embedding* of an arbitrary graph G into a fixed *constraint graph* C is an injection $f\colon V(G) \to V(C)$ together with an assignment, to each edge uv of G, of a path from $f(u)$ to $f(v)$ in C. The *minimum load factor* of G relative to C is the minimum, over all embeddings of G in C, of the maximum number of paths in the embedding that share a common edge in C.

For example, for the case in which C is the infinite-length one-dimensional grid, the minimum load factor of G with respect to C is called the *cutwidth* of G. In the $\mathcal{NP}$-complete MIN CUT LINEAR ARRANGEMENT problem [11], we are given a graph G and an integer k, and are asked whether the cutwidth of G is no more than k. Related $\mathcal{NP}$-complete problems address the cutwidth of G relative to C when C is the infinite-length, fixed-width two-dimensional grid (2-D GRID LOAD FACTOR) or when C is the infinite-height binary tree (BINARY TREE LOAD FACTOR).

THEOREM 4.1. *For any fixed k and any fixed C, the family of graphs for which the minimum load factor relative to C is less than or equal to k is closed under the immersion ordering.*

Proof. Let an embedding f of G in C with load factor no more than k be given. Suppose that $H \leq_i G$. If $H \subseteq G$, then the embedding that restricts f to H clearly has load factor no more than k. If H is obtained from G by lifting the edges uv and vw incident at vertex v, then an embedding for H can be defined by assigning to the resulting edge uw the composition of the paths from u to v and from v to w in C. This cannot increase the load factor. □

COROLLARY 4.2. *For any fixed k,* MIN CUT LINEAR ARRANGEMENT, 2-D GRID LOAD FACTOR, *and* BINARY TREE LOAD FACTOR *can be decided in polynomial time.*

This result has previously been reported for MIN CUT LINEAR ARRANGEMENT, using an algorithm with time complexity $O(n^{k-1})$ [16]. We now prove that it is sometimes possible to employ excluded-minor knowledge on immersion-closed families to guarantee quadratic-time decision complexity.

THEOREM 4.3. *For any fixed k,* MIN CUT LINEAR ARRANGEMENT, 2-D GRID LOAD FACTOR, *and* BINARY TREE LOAD FACTOR *can be decided in $O(n^2)$ time.*

Proof. For MIN CUT LINEAR ARRANGEMENT, it is known that there are binary trees with cutwidth exceeding k for any fixed k [2]. Let T denote such a tree. Because T has maximum degree three, it follows that $G \geq_m T$ implies $G \geq_i T$. Thus no $G \geq_m T$ can be a "yes" instance (recall that the "yes" family is immersion closed) and we know from [22] that all "yes" instances have bounded tree-width. (Tree-width and the associated metric branch-width are defined and related to each other in [23].) Now we need only search for a satisfactory tree-decomposition, using the $O(n^2)$ method of [24]. Testing for obstruction containment in the immersion order can be done in linear time on graphs of bounded tree-width in this setting [24], given such a tree-decomposition.

Sufficiently large binary trees are excluded for 2-D GRID LOAD FACTOR as well (recall that both k and the grid-width are fixed).

For BINARY TREE LOAD FACTOR, it is a simple exercise to see that all "yes" instances have bounded tree-width by building a tree-decomposition with width at most $3k$ from a binary tree embedding with load factor at most k. (The decomposition tree T can be taken to be the finite subtree of C that spans the image of G. For vertex $u \in V(T)$, the associated set of vertices of G contains the inverse image of u if one exists, and every vertex $v \in V(G)$ with an incident edge that is assigned a path in C

that includes u.) □

5. Other methods. The application of Theorems 2.1–2.4 directly ensures polynomial-time decidability. A less direct approach relies on the well-known notion of polynomial-time transformation, as we now illustrate with an example. The $\mathcal{NP}$-complete MODIFIED MIN CUT problem was first introduced in [13]. Given a linear layout ℓ of a simple graph G, the *modified cutwidth* at location i, $c_\ell(i)$, is $|\{e : e = uv \in E$ such that $\ell(u) < i$ and $\ell(v) > i\}|$. The modified cutwidth of the entire layout is $c_\ell = \max\{c_\ell(i) : 1 \leq i \leq n\}$, and the modified cutwidth of G is $mc(G) = \min\{c_\ell : \ell$ is a linear layout of $G\}$. Given both G and a positive integer k, the MODIFIED MIN CUT problem asks whether $mc(G)$ is less than or equal to k. Observe that, while the MIN CUT LINEAR ARRANGEMENT problem addresses the number of edges that cross any cut *between* adjacent vertices in a linear layout, the MODIFIED MIN CUT problem addresses the number of edges that cross (and do not end at) any cut *on* a vertex in the layout.

When k is fixed, neither the family of "yes" instances nor the family of "no" instances for MODIFIED MIN CUT is closed under either of the available orders. Nevertheless, we can employ a useful consequence of well-partially-ordered sets.

CONSEQUENCE (see [8]). *If $(S, \leq)$ is a well-partially-ordered set that supports polynomial-time order tests for every fixed element of S, and if there is a polynomial-time computable map t: $D \to S$ such that for $F \subseteq D$,* (a) *$t(F) \subseteq S$ is closed under $\leq$ and* (b) *$t(F) \cap t(D - F) = \emptyset$, then there is a polynomial-time decision algorithm to determine for input z in D whether z is in F.*

To use this result on fixed-k MODIFIED MIN CUT, observe that if any vertex of a simple graph G has degree greater than $2k + 2$, then G is automatically a "no" instance. Given a simple graph G with maximum degree less than or equal to $2k + 2$, we first augment G with loops as follows: if a vertex v has degree $d < 2k + 2$, then it receives $(2k + 2) - d$ new loops. Letting G' denote this augmented version of G, we now replace G' with the Boolean matrix M, in which each row of M corresponds to an edge of G' and each column of M corresponds to a vertex of G'. That is, M has $|E'|$ rows and n columns, with $M_{ij} = 1$ if and only if edge i is incident on vertex j. M and $k' = 3k + 2$ are now viewed as input to the GATE MATRIX LAYOUT problem [3], in which we are asked whether the columns of M can be permuted so that, if in each row we change to $*$ every 0 lying between the row's leftmost and rightmost 1, then no column contains more than k 1s and $*$s. Thus a permutation of the columns of M corresponds to a linear layout of G. For such a permutation, each $*$ in column i, $1 < i < n$, represents a distinct edge crossing a cut at vertex i in the corresponding layout of G.

THEOREM 5.1. *For any fixed k,* MODIFIED MIN CUT *can be decided in $O(n^2)$ time.*

Proof. We apply the consequence, using the set of all graphs for S, $\leq_m$ for $\leq$, the set of simple graphs of maximum degree $2k + 2$ for D, the family of "yes" instances in D for F, and the composition of the map just defined from graphs to matrices with the map of [5] from matrices to graphs for t. Testing for membership in D and computing t are easily accomplished in $O(n^2)$ time. That $t(F)$ is closed under $\leq_m$ and excludes a planar graph for any fixed k is established in [5]. Finally, condition (b) holds because, for any G in D, $t(G)$ is a "yes" instance for GATE MATRIX LAYOUT with parameter $3k + 2$ if and only if G is a "yes" instance for MODIFIED MIN CUT with parameter k. □

6. Search problems. Given a decision problem Π_D and its search version Π_S, any method that pinpoints a solution to Π_S by repeated calls to an algorithm that answers Π_D is termed a *self-reduction*. This simple notion has been formalized with various refinements in the literature, but the goal remains the same: to use the existence of a decision algorithm to prove the existence of a search algorithm. Note the crucial importance of self-reducibility in the current context, given that Theorems 2.1–2.4 only yield decision algorithms, not search procedures.

It sometimes suffices to *fatten up* a graph by adding edges to isolate a solution. For example, this strategy can be employed to construct solutions to (fixed-k) GATE MATRIX LAYOUT, when any exist, in $O(n^4)$ time [1]. It follows from the proof of Theorem 5.1 that the same can be said for MODIFIED MIN CUT as well. We leave it to the reader to verify that such a scheme works for the search version of (fixed-k) VERTEX SEPARATION, by attempting to add each edge in $V \times V - E$ in arbitrary order, retaining in turn only those whose addition maintains a "yes" instance, and at the end reading off a satisfactory layout (from right to left) by successively removing a vertex of smallest degree. This self-reduction automatically solves the search version of SEARCH NUMBER, also (see the discussion of "2-expansions" in [4]).

Conversely, it is sometimes possible to *trim down* a graph by deleting edges so as to isolate a solution. It is easy to see that this simple strategy yields an $O(n^4)$-time algorithm for the search version of (fixed-k) MAX LEAF SPANNING TREE, by attempting to delete each edge in E in arbitrary order, retaining in turn only those whose deletion does not maintain a "yes" instance.

Another technique involves the use of graph *gadgets*. A simple gadget, consisting of two new vertices with k edges between them, is useful in constructing a solution to (fixed-k) MIN CUT LINEAR ARRANGEMENT, when any exist, in $O(n^4)$ time [1]. A similar use of gadgets enables efficient self-reductions for load factor problems. (On BINARY TREE LOAD FACTOR, for example, we can begin by using two k-edge gadgets uv and wx to locate a vertex y of the input graph that can be mapped to a leaf of the constraint tree by identifying u, w, and y.)

Indeed, polynomial-time self-reductions exist for all of the problems that we study in this paper. In addition to the straightforward methods just mentioned, faster but more elaborate techniques are described in [6], [9].

7. Concluding remarks. The range of problems amenable to an approach based on well-partially-ordered sets is remarkable. Although the problems that we have addressed in this paper are all fixed-parameter versions of problems that are $\mathcal{NP}$-hard in general, we remind the reader that by fixing parameters we do not automatically trivialize problems, and thereby obtain polynomial-time decidability (consider, for example, GRAPH k–COLORABILITY [11]). Moreover, the techniques that we have employed can be used to guarantee membership in $\mathcal{P}$ for problems that have no associated (fixed) parameter [8].

Table 1 suffers from one notable omission, namely, BANDWIDTH [11]. The only success reported to date has concerned restricted instances of TOPOLOGICAL BANDWIDTH. Both BANDWIDTH and the related EDGE BANDWIDTH problem [7] have resisted this general line of attack so far. Clearly, BANDWIDTH is at least superficially similar to other layout permutation problems we have addressed, and fixed-k BANDWIDTH, like the others, is solvable in (high-degree) polynomial-time with dynamic programming [12]. Perhaps BANDWIDTH, however, is really different; it is one of the very few problems that remain $\mathcal{NP}$-complete when restricted to trees [10].

The results that we have derived here immediately extend to hypergraph problem variants as long as hypergraph instances can be efficiently reduced to graph instances. For example, such reductions are known for HYPERGRAPH VERTEX SEPARATION and HYPERGRAPH MODIFIED MIN CUT [17], [27].

Finally, we observe that even partial-orders that fail to be well-partial-orders (on the set of all graphs) may be useful. For example, although it is well known that graphs are not well-partially-ordered under the topological order, it has been shown [15] that, for every fixed h, all graphs without h vertex-disjoint cycles *are* well-partially-ordered under topological containment. Also, polynomial-time order tests exist [24]. Problems such as (fixed-k) TOPOLOGICAL BANDWIDTH, therefore, are decidable in polynomial time as long as the input is restricted to graphs with no more than h disjoint cycles (for fixed h). Similarly, we might employ the result [26] that graphs without a path of length h, for h fixed, are well-partially-ordered under subgraph containment.

Acknowledgment. We wish to express our gratitude to the anonymous referee whose extremely careful review of the original version of this paper greatly helped to improve and streamline the final presentation.

REFERENCES

[1] D. J. Brown, M. R. Fellows, and M. A. Langston, *Polynomial-time self-reducibility: Theoretical motivations and practical results*, Internat. J. Comput. Math., 31 (1989), pp. 1–9.

[2] M.-J. Chung, F. Makedon, I. H. Sudborough, and J. S. Turner, *Polynomial time algorithms for the min cut problem on degree restricted trees*, SIAM J. Comput., 14 (1985), pp. 158–177.

[3] N. Deo, M. S. Krishnamoorthy, and M. A. Langston, *Exact and approximate solutions for the gate matrix layout problem*, IEEE Trans. Computer-Aided Design, 6 (1987), pp. 79–84.

[4] J. A. Ellis, I. H. Sudborough, and J. S. Turner, *Graph separation and search number*, in Proc. 21st Allerton Conf. on Communication, Control and Computing, Urbana, Illinois, 1983, pp. 224–233.

[5] M. R. Fellows and M. A. Langston, *Nonconstructive advances in polynomial-time complexity*, Inform. Process. Lett., 26 (1987), pp. 157–162.

[6] ———, *Fast self-reduction algorithms for combinatorial problems of* VLSI *design*, in Proc. 3rd Aegean Workshop on Computing, Corfu, Greece, 1988, pp. 278–287.

[7] ———, *Layout permutation problems and well-partially-ordered sets*, in Proc. 5th MIT Conf. on Advanced Research in VLSI, Cambridge, Massachusetts, 1988, pp. 315–327.

[8] ———, *Nonconstructive tools for proving polynomial-time decidability*, J. Assoc. Comput. Mach., 35 (1988), pp. 727–739.

[9] ———, *On search, decision and the efficiency of polynomial-time algorithms*, in Proc. 21st ACM Symp. on Theory of Computing, Seattle, Washington, 1989, pp. 501–512.

[10] M. R. Garey, R. L. Graham, D. S. Johnson, and D. E. Knuth, *Complexity results for bandwidth minimization*, SIAM J. Appl. Math., 34 (1978), pp. 477–495.

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness*, Freeman, San Francisco, CA, 1979.

[12] E. M. Gurari and I. H. Sudborough, *Improved dynamic programming algorithms for bandwidth minimization and the min cut linear arrangement problem*, J. Algorithms, 5 (1984), pp. 531–546.

[13] T. Lengauer, *Black-white pebbles and graph separation*, Acta Inform., 16 (1981), pp. 465–475.

[14] W. Mader, *Hinreichende Bedingungen für die Existenz von Teilgraphen, die zu einem vollständigen Graphen homöomorph sind*, Math. Nachr., 53 (1972), pp. 145–150.

[15] ———, *Wohlquasigeordnete Klassen endlicher Graphen*, J. Combin. Theory Ser. B, 12 (1972), pp. 105–122.

[16] F. S. MAKEDON AND I. H. SUDBOROUGH, *On minimizing width in linear layouts*, Lecture Notes in Comput. Sci., 154 (1983), pp. 478–490.
[17] Z. MILLER AND I. H. SUDBOROUGH, *Polynomial algorithms for recognizing small cutwidth in hypergraphs*, in Proc. 2nd Aegean Workshop on Computing, Loutraki, Greece, 1986, pp. 252–260.
[18] C. H. PAPADIMITRIOU, private communication, 1988.
[19] T. D. PARSONS, *Pursuit-evasion in a graph*, in Theory and Application of Graphs, Y. Alavi and D. R. Lick, eds., Springer-Verlag, Berlin, New York, 1976, pp. 426–441.
[20] N. ROBERTSON, private communication, 1987.
[21] N. ROBERTSON AND P. D. SEYMOUR, *Graph minors* IV. *Tree-width and well-quasi-ordering*, J. Combin. Theory Ser. B, 48 (1990), pp. 227–254.
[22] ———, *Graph minors* V. *Excluding a planar graph*, J. Combin. Theory Ser. B, 41 (1986), pp. 92–114.
[23] ———, *Graph minors* X. *Obstructions to tree-decomposition*, J. Combin. Theory Ser. B, 52 (1991), pp. 153–190.
[24] ———, *Graph minors* XIII. *The disjoint paths problem*, to appear.
[25] ———, *Graph minors* XVI. *Wagner's conjecture*, to appear.
[26] P. D. SEYMOUR, private communication, 1989.
[27] I. H. SUDBOROUGH, private communication, 1988.

SIAM J. DISCRETE MATH.
Vol. 5, No. 1, pp. 127–143, February 1992
© 1992 Society for Industrial and Applied Mathematics
011

ON TENSOR POWERS OF INTEGER PROGRAMS*

ROBIN PEMANTLE†, JAMES PROPP‡, AND DANIEL ULLMAN§

Abstract. A natural product on integer programming problems with nonnegative coefficients is defined. Hypergraph covering problems are a special case of such integer programs, and the product defined is a generalization of the usual hypergraph product. The main theorem of this paper gives a sufficient condition under which the solution to the nth power of an integer program is asymptotically as good as the solution to the same nth power when the variables are not necessarily integral but may be arbitrary nonnegative real numbers.

Key words. integer program, linear program, hypergraph, covering

AMS(MOS) subject classifications. 90C10; secondary, 05C65

1. Definitions and notations. The minimization problems that we consider here are of the form "Minimize the quantity

$$c_1x_1 + c_2x_2 + \cdots + c_dx_d$$

subject to the constraints

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1d}x_d &\geq b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2d}x_d &\geq b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{md}x_d &\geq b_m, \end{aligned}$$

where a_{ij}, b_i, and c_j are fixed nonnegative real numbers, and x_j are unknown nonnegative integers." Label the constraints $C(1), \cdots, C(m)$. We would lose no generality by throwing out those variables x_j for which $c_j = 0$ (together with every constraint $C(i)$ for which $a_{ij} > 0$) and those constraints for which $b_i = 0$, thus making all b_i and c_j positive. For the time being, however, we do not require positivity.

We may write our integer program more compactly as "Minimize c^Tx subject to $Ax \geq b$ with $x \geq 0$ and integral," where A is a nonnegative m-by-d matrix, b is a nonnegative column vector of length m, c is a nonnegative column vector of length d, and x ranges over the set of nonnegative integer column vectors of length d. Assume further that $b_i > 0$ implies the existence of a j for which $a_{ij} > 0$. We denote this integer program by the triple $P = (A, b, c)$. Our positivity assumptions on A, b, and c imply that feasible solution vectors x exist; the minimum possible value of c^Tx as x ranges over all solution vectors is called the *value* of P, denoted $v(P)$.

Associated with the integer program P is its *LP-relaxation*, obtained by dropping the requirement that the entries in the solution vector be integers. We let $v^*(P)$ (the

* Received by the editors May 22, 1989; accepted for publication (in revised form) January 10, 1991.

† University of California at Berkeley, Berkeley, California 94720. Present address, Oregon State University, Corvallis, Oregon 97331. This author was supported by a National Science Foundation post-doctoral fellowship.

‡ University of California at Berkeley, Berkeley, California 94720. Present address, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

§ The George Washington University, Washington, DC 20052. This author was supported by a Junior Scholar Incentive Award from The George Washington University.

LP-relaxed value of P) signify the optimum of this relaxed linear program. Note that $v^*(P)$ is a real number between 0 and $v(P)$.

Also associated with the minimization program P is the program $P^\perp$, "Maximize $b^T y$ subject to $A^T y \leq c$, with $y \geq 0$ and integral." The program $P^\perp$ is called the *dual* of P. The well-known duality theorem asserts that the optimum values of the respective LP-relaxations of P and $P^\perp$ are equal; that is, if we extend our definitions of v and v^* to cover maximization programs in the natural way, we have $v^*(P^\perp) = v^*(P)$. However, it is by no means true that $v(P^\perp) = v(P)$; for, in general, we have

$$0 \leq v(P^\perp) \leq v^*(P^\perp) = v^*(P) \leq v(P),$$

so that if b and c have integer entries, but $v^*(P)$ is not an integer, there is no chance of the integers $v(P^\perp)$ and $v(P)$ being equal.

Given two minimization programs P and P', there is natural way to define two other programs called their *sum* and *tensor product.* (For wholly analogous constructions in information theory, see pp. 65–66 of [7]. See also [1], where the analogous sum of two network flow problems is seen to correspond to parallel-composition of the two networks.) Suppose $P = (A, b, c)$, $P' = (A', b', c')$, where A is m-by-d and A' is m'-by-d'. We define $A \oplus A'$ as the $(m + m')$-by-$(d + d')$ matrix

$$\begin{pmatrix} A & 0 \\ 0 & A' \end{pmatrix},$$

$b \oplus b'$ as the vector of length $m + m'$ obtained by concatenating the vectors b and b', and $c \oplus c'$ as a similar concatenation; we then define the sum $P \oplus P'$ of the programs P and P' to be the program $(A \oplus A', b \oplus b', c \oplus c')$. To define multiplication of programs, it is notationally convenient to allow indices for vectors and matrices to be not just natural numbers, but also pairs of natural numbers; then, the tensor product of A and A' may be defined as the matrix whose $((i,j),(k,l))$th entry is $a_{ik}a'_{jl}$. (If, as is often done, we re-index the product so that the indices are natural numbers, then the matrix $A \otimes A'$ may be depicted as

$$\begin{pmatrix} a_{1\,1}A' & a_{1\,2}A' & \cdots & a_{1\,n}A' \\ a_{2\,1}A' & a_{2\,2}A' & \cdots & a_{2\,n}A' \\ \vdots & \vdots & & \vdots \\ a_{m\,1}A' & a_{m\,2}A' & \cdots & a_{m\,n}A' \end{pmatrix};$$

however, this representation is not necessary for our purposes.) We define $b \otimes b'$ as the column vector of length mm' whose (i,j)th entry is $b_i b'_j$, and $c \otimes c'$ as the column vector of length nn' whose (k,l)th entry is $c_k c'_l$. We conclude by defining the product $P \otimes P'$ of the programs P and P' to be the program $(A \otimes A', b \otimes b', c \otimes c')$.

We leave it to the reader to verify that $\oplus$ and $\otimes$ satisfy the natural commutativity, associativity, and distributivity properties; moreover, we can define the "empty program" (no variables, no constraints) and the "identity program" (with A as the 1-by-1 identity matrix, and b and c as vectors whose lone entry is 1) to serve as identity elements for $\oplus$ and $\otimes$, respectively. We further remark that, defining $\oplus$ and $\otimes$ for maximization programs in the obvious way, we have $(P \oplus P')^\perp = P^\perp \oplus P'^\perp$ and $(P \otimes P')^\perp = P^\perp \otimes P'^\perp$. Finally, we point out that if P is a minimization program in which some of the entries of the b-vector or c-vector equal 0, there is a canonical program P' obtained by throwing out the corresponding variables and constraints;

moreover, the mapping $P \mapsto P'$ preserves $v(\cdot)$ and $v^*(\cdot)$ and commutes with the operations $\oplus$ and $\otimes$. Hence, in the following we may without loss of generality assume that b_i and c_j are positive for all i, j and that $v(P) > 0$.

An easy fact from the next section is that $v^*(P \otimes P') = v^*(P)v^*(P')$; however, an example there will show that it is not true in general that $v(P \otimes P') = v(P)v(P')$, and that we must content ourselves with the weaker statement $v(P \otimes P') \leq v(P)v(P')$. If we define $P^{\otimes n}$ as $P \otimes P \otimes \cdots \otimes P$ with n occurrences of P, then this inequality implies that $v(P^{\otimes i+j}) \leq v(P^{\otimes i})v(P^{\otimes j})$ for all i, j; by Fekete's lemma [2], we conclude that as n gets large, the quantity

$$\sqrt[n]{v(P^{\otimes n})}$$

approaches its infimum, which we call the *asymptotic optimum value* of P. The following theorem gives conditions on P that force the asymptotic optimum value to equal the value of the LP-relaxation of P.

THEOREM 1. *Let $P = (A, b, c)$ be an integer minimization program in which b_i and c_j are strictly positive for all i and j. Suppose there exists an optimum solution-vector $(\alpha(1), \alpha(2), \cdots, \alpha(d))$ for the* LP*-relaxation of P such that*

$$\prod_j \left(\frac{a_{ij}}{b_i}\right)^{a_{ij}\alpha(j)/v^*(P)} \leq 1 \text{ for all } i. \tag{1}$$

Then $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ as $n \to \infty$.

(In condition (1), we are to take $0^0 = 1$, as usual. If we multiply the exponent $V^*(P)$, the resulting inequality is equivalent to (1) and looks simpler, but the form we have given will be more useful.)

It has already been mentioned (see the first paragraph of §1) that once we have assumed that our program P satisfies $a_{ij}, b_i, c_j \geq 0$ for all i, j, we may as well assume that $b_i > 0$ for all i and $c_j > 0$ for all j. An explanation of the role played by condition (1) will be given later. For now, let us mention the following result.

THEOREM 2. *Suppose $P = (A, b, c)$ is an integer minimization program in which $0 \leq a_{ij} \leq b_i$ and $c_j > 0$ for all i and j. Then $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ as $n \to \infty$.*

This is a special case of Theorem 1. The hypothesis of Theorem 2 gives us $a_{ij}/b_i \leq 1$ for all i, j, so that condition (1) is automatically satisfied by any optimum solution-vector α.

Theorem 2 is strictly weaker than Theorem 1, because condition (1) is strictly weaker than $a_{ij} \leq b_i$. For example, the set of (x, y) in the positive quadrant for which the matrix $\left(\begin{smallmatrix} x & y \\ y & x \end{smallmatrix}\right)$ satisfies (1) when $b = c = (1, 1)^T$ is the region $\{(x, y) : x^x y^y \leq 1\}$, which strictly includes the unit square. It can be shown that the condition $x^x y^y \leq 1$ is sharp for programs of this kind; that is, $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ if and only if $x^x y^y \leq 1$. It would be interesting to know if condition (1) is sharp in general.

Section 2 of this paper outlines the relationship between integer programming and hypergraph theory and gives the basic results on tensor powers of integer programs. Section 3 contains a probabilistic proof of Theorem 1. Section 4 contains a constructive (in fact, greedy) proof of Theorem 2.

2. Background and preliminary results. First, we briefly recapitulate the discussion of hypergraphs and integer programs contained in [3]. A *hypergraph* $\mathcal{H} = (V, E)$ is a finite vertex set V together with a collection $E \subset 2^V$ of nonempty subsets of V, called (hyper)edges. A *cover* of $\mathcal{H}$ is a set of vertices C that intersects every

edge of $\mathcal{H}$; that is, for all $e \in E$, $C \cap e \neq \phi$. The *covering number* $\tau(\mathcal{H})$ is the smallest cardinality of a cover of $\mathcal{H}$. Suppose $\mathcal{H}$ has d vertices and m edges; then the *incidence matrix* of $\mathcal{H}$ is the m-by-d matrix A with (i,j)th entry equal to 1 if the ith edge contains the jth vertex, and equal to 0 otherwise. Furthermore, if we let b and c be vectors of length m and d, respectively, consisting entirely of 1's, and associate with each cover C of $\mathcal{H}$ a d-vector x whose jth entry is 1 or 0, according to whether or not C contains the jth vertex of $\mathcal{H}$, then $\tau(\mathcal{H})$ is seen to equal the value of the integer minimization program (A, b, c). This integer programming viewpoint naturally leads us to consider the relaxed version of the program in which the integrality constraint has been dropped; the value of the relaxed program is called the *fractional covering number* $\tau^*(\mathcal{H})$ of $\mathcal{H}$.

The definitions that appear in §1 all correspond to notions that have already been used in the theory of hypergraphs; for example, if P_i is the program that corresponds to the problem of determining $\tau(\mathcal{H}_i)$ $(i = 1, 2)$, then $P_1 \otimes P_2$ corresponds to the problem of determining $\tau(\mathcal{H}_1 \times \mathcal{H}_2)$, where $\mathcal{H}_1 \times \mathcal{H}_2$ is the hypergraph defined as follows (with $\times$ denoting Cartesian product on the right):

$$V(\mathcal{H}_1 \times \mathcal{H}_2) = V(\mathcal{H}_1) \times V(\mathcal{H}_2),$$
$$E(\mathcal{H}_1 \times \mathcal{H}_2) = \{e_1 \times e_2 : e_1 \in E(\mathcal{H}_1), e_2 \in E(\mathcal{H}_2)\}.$$

In [8], McEliece and Posner proved (in different notation) a special case of Theorem 1, namely,

$$\lim_{n \to \infty} \sqrt[n]{\tau(\mathcal{H}^n)} = \tau^*(\mathcal{H}) \ .$$

This amounts to our Theorem 1 in the special case that the matrix A consists entirely of 0's and 1's, and the vectors b and c consist entirely of 1's. This analogy suggests the following definition.

DEFINITION 3. A program P is a *fuzzy hypergraph covering* (FHC) program if all b_i and c_j are equal to 1 and $0 \leq a_{i,j} \leq 1$ for all i, j. (The terminology arises by analogy with fuzzy sets.)

This paper extends McEliece and Posner's result to a more general class, including FHC programs.

Remark. It is not immediately clear that the condition $c_j = 1$ for all j is inessential, but an argument for this is given to finish the proof of Theorem 1 after it has been proved in the case where $c_j = 1$ for all j. Note, however, that the normalization of b to a vector of ones breaks the symmetry between P and $P^\perp$ and may thus change whether $v(P) = v(P^\perp)$. Finally, the condition $0 \leq a_{ij} \leq 1$ may not be entirely relaxed without invalidating the theorem (see the second-to-last paragraph of this section).

Example. A typical FHC program is the following: "Minimize $x_1 + x_2$ subject to

$$x_1 + \frac{1}{2}x_2 \geq 1, \tag{2}$$

$$\frac{1}{3}x_1 + x_2 \geq 1, \tag{3}$$

with $x_1, x_2 \geq 0$." This program P is associated with the matrix

$$A = \begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{3} & 1 \end{pmatrix}.$$

Clearly, $v(P) = 2$, with optimal solution vectors $x = (1,1)$ and $(0,2)$. To determine $v^*(P)$, note that the feasibility of $x = (\frac{3}{5}, \frac{4}{5})$ implies that $v^*(P) \leq \frac{7}{5}$, while the feasibility of $y = (\frac{4}{5}, \frac{3}{5})$ for the dual program $P^\perp$ implies that $v^*(P) = v^*(P^\perp) \geq \frac{7}{5}$. The tensor square of this program, $P^{\otimes 2}$, has coefficient matrix

$$A^{\otimes 2} = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{3} & 1 & \frac{1}{6} & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{6} & 1 & \frac{1}{2} \\ \frac{1}{9} & \frac{1}{3} & \frac{1}{3} & 1 \end{pmatrix},$$

and we readily see that $x = (0,1,1,1)$ is a solution vector, so that $v(P^{\otimes 2}) \leq 3$. This illustrates that $v(P^{\otimes 2})$ may be strictly less than $v(P)^2$. Here $v(P^{\otimes 2})^{1/2} = \sqrt{3}$ and, in fact, by Theorem 1, $v(P^{\otimes n})^{1/n} \to \frac{7}{5}$.

The following proposition does not make use of the FHC property, but the fuzzy hypergraph point of view may still be helpful to the reader in interpreting the statements and their proofs.

PROPOSITION 4. *The following hold:*

(i) $v^*(P \oplus P') = v^*(P) + v^*(P')$;

(ii) $v(P \oplus P') = v(P) + v(P')$;

(iii) $v^*(P \otimes P') = v^*(P)v^*(P')$;

(iv) $v(P \otimes P') \leq v(P)v(P')$.

Proof. To prove (i) and (ii) note that if x and x' are solution vectors for P and P', then their concatenation is a solution vector for $P \oplus P'$; and, conversely, every solution vector for $P \oplus P'$ is such a concatenation. To prove (iii), suppose x and x' are optimal solution vectors to the respective linear programs P and P'. Then since

$$(A \otimes A')(x \otimes x') = (Ax) \otimes (A'x') \geq b \otimes b'$$

(note the use of nonnegativity), $x \otimes x'$ is a feasible vector for the product program $P \otimes P'$, with

$$\begin{aligned}(c \otimes c')^T (x \otimes x') &= (c^T x)(c'^T x') \\ &= v^*(P)v^*(P'),\end{aligned}$$

so that $v^*(P \otimes P') \leq v^*(P)v^*(P')$. On the other hand, suppose that y and y' are optimal solution vectors to the dual programs $P^\perp$ and $P'^\perp$; then $y \otimes y'$ is a feasible vector for the program $P^\perp \otimes P'^\perp = (P \otimes P')^\perp$ with

$$\begin{aligned}(b \otimes b')^T (y \otimes y') &= (b^T y)(b'^T y') \\ &= v^*(P)v^*(P'),\end{aligned}$$

so that $v^*(P \otimes P') \geq v^*(P)v^*(P')$. (Note that we have applied the duality theorem three times: to P, to P', and to $P \otimes P'$.) We conclude that $v^*(P \otimes P') = v^*(P)v^*(P')$. The proof of (iv) is the same as the first half of the proof of (iii) (but we no longer have a duality principle to provide us with the reverse inequality). □

The preceding proposition gives us an upper bound on $v(P \otimes P')$. The following less obvious result (an extension of the first inequality in Füredi's formula [3, (5.14)]) gives us a lower bound.

PROPOSITION 5. *It holds that* $v(P \otimes P') \geq \max\{v^*(P)v(P'), v(P)v^*(P')\}$.

Proof. Put $P = (A, b, c)$, $P' = (A', b', c')$. By symmetry, it is enough to show that

$$v^*(P) \leq \frac{v(P \otimes P')}{v(P')}$$

when $v(P') > 0$. Given an optimal solution vector z to $P \otimes P'$, indexed by pairs (k, l), where $1 \leq k \leq d$ and $1 \leq \ell \leq d'$, define

$$x_k = \frac{1}{v(P')} \sum_l c'_l z_{(k,l)} \ .$$

We show that x is a feasible solution to the LP-relaxation of P. Fix i and note that

$$\sum_k a_{ik} x_k = \sum_k a_{ik} \frac{1}{v(P')} \sum_l c'_l z_{(k,l)} = \frac{b_i}{v(P')} \sum_l c'_l \sum_k \frac{a_{ik}}{b_i} z_{(k,l)} \ .$$

Setting

$$y_l = \sum_k \frac{a_{ik}}{b_i} z_{(k,l)},$$

we get

$$\sum_k a_{ik} x_k = \frac{b_i}{v(P')} \sum_l c'_l y_l \ .$$

However, since

$$\sum_l a'_{jl} y_l = \frac{1}{b_i} \sum_{k,l} a_{ik} a'_{jl} z_{(k,l)} = \frac{1}{b_i} \left((A \otimes A')(z) \right)_{(i,j)} \geq \frac{1}{b_i} (b \otimes b')_{(i,j)} = b'_j$$

for all j, y is a feasible solution to P', and hence satisfies

$$\sum_l c'_l y_l \geq v(P') \ .$$

Thus

$$\sum_k a_{ik} x_k \geq \frac{b_i}{v(P')} v(P') = b_i,$$

establishing that x is indeed a feasible solution to P. We conclude that

$$v^*(P) \leq \sum_k c_k x_k = \sum_k c_k \frac{1}{v(P')} \sum_l c'_l z_{(k,l)} = \frac{1}{v(P')} \sum_{k,l} c_k c'_l z_{(k,l)} = \frac{v(P \otimes P')}{v(P')},$$

which was to be shown. □

Proposition 4(iii) implies that $\sqrt[n]{v(P^{\otimes n})} \geq \sqrt[n]{v^*(P^{\otimes n})} = v^*(P)$. Our main theorem states that if P is an FHC program or, more generally, if P satisfies condition (1), then, in fact, $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ as $n \to \infty$. Our first proof of this fact relies heavily on the ideas of McEliece and Posner and, in particular, uses the same sort of probabilistic construction as theirs did; however, our argument is necessarily more complicated, since optimal solution vectors x will typically need to have entries

much larger than 1 to satisfy the constraints. In our second proof, we use a greedy construction as in Lovász's proof of the McEliece and Posner theorem [6].

It should be mentioned that the convergence $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ does not hold for integer minimization programs in general. As an illustration of this, let P be the program "Minimize $x+y+z$ subject to $2x \geq 1$, $2y \geq 1$, $2z \geq 1$ with $x, y, z \geq 0$ and integral." Then $v(P^{\otimes n}) = 3^n$ for all n, whereas $v^*(P) = \frac{3}{2}$. Hence, we see that for convergence to $v^*(P)$ to hold, something like the FHC property is required.

It should also be mentioned that the convergence $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ typically does not hold for integer maximization programs, even when all of the a_{ij} are 0's and 1's. For example, consider the problem P of maximizing $x_1+x_2+x_3+x_4+x_5$ subject to the constraints that x_1+x_2, x_2+x_3, x_3+x_4, x_4+x_5, and x_5+x_1 all be at most 1. Viewed as an integer program, this is equivalent to finding the largest independent set of vertices in the pentagon graph C_5. More generally, the nth power of P is equivalent to finding the largest independent set of vertices in the nth strong power of C_5 (see [4] for graph-product and graph-power terminology). The limit $\sqrt[n]{v(P)}$ is known as the Shannon capacity of the graph C_5 [9]. It has been shown [5] that the Shannon capacity of the graph C_5 is $\sqrt{5}$; on the other hand, $v^*(P)$ is $\frac{5}{2}$, since $(1/2, 1/2, 1/2, 1/2, 1/2)$ is a solution to both P and $P^\perp$. This example shows that Theorem 1 does not dualize to a theorem about maximization programs; that is, $\sqrt[n]{v((P^\perp)^{\otimes n})}$ need not approach $v^*(P^\perp) = v^*(P)$.

3. Proof of Theorem 1. The proof of Theorem 1 requires some ideas from the theory of two-player zero-sum games. Treat the matrix A as the payoff matrix in a two-player zero-sum game between Alpha, who names a variable (column of A), and Beta, who names a constraint (row of A), where Alpha tries to maximize the payoff and Beta tries to minimize the payoff; the payoff is a_{ij} when constraint i and variable j are chosen. (To prepare for the multi-indices that are to follow, write a_{ij} as $a(i,j)$.) Alpha has an optimal mixed strategy that chooses each variable $x(j)$ with some probability $u(j)$. The expected value of the payoff under this strategy is called the value of the game (see [10]) and will be denoted by $\mathcal{S}$. There is a very simple relationship between this game and the LP-relaxation of the program P with matrix A and with b and c consisting of ones, namely, that if $(\alpha(1), \cdots, \alpha(d))$ is a feasible solution for the linear program, then $u(j) = \alpha(j)/\sum_j \alpha(j)$ gives a strategy for Alpha with a guaranteed payoff of at least $1/\sum_j \alpha(j)$. Moreover, $v^*(P)$ (the value of the linear program P) is equal to $1/\mathcal{S}$ (the reciprocal of the value of the matrix game), with each optimal solution-vector α giving rise to an optimal strategy u. In the case where the c vector is not all ones, it will still be convenient to let $u(j)$ denote $\alpha(j)/\sum_j \alpha(j)$, where $(\alpha(1), \cdots, \alpha(d))$ (a feasible solution with minimal cost) is given by the hypothesis of Theorem 1.

To illustrate this, consider the previous example of minimizing $x_1 + x_2$ subject to the constraints $x_1 + \frac{1}{2}x_2 \geq 1$ and $\frac{1}{3}x_1 + x_2 \geq 1$; $x_1 + x_2$ is minimized by choosing $x_1 = \frac{3}{5}$ and $x_2 = \frac{4}{5}$, and $v^*(P) = \frac{7}{5}$. The best strategy for Alpha in the game with payoff matrix

$$A = \begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{3} & 1 \end{pmatrix}$$

is to choose $u(1) = \frac{3}{5}/\frac{7}{5} = \frac{3}{7}$ and $u(2) = \frac{4}{7}$. Then $\mathcal{S} = \frac{5}{7}$, which is clear from the fact that the expected payoff against this strategy is $\frac{5}{7}$, whether Beta chooses constraint 1 or constraint 2 or any probabilistic combination of the two. In other words, $\sum_j a(i,j)u(j) = \frac{5}{7}$ for $i = 1, 2$.

Remark. For ease of exposition, we will assume hereafter that, as in the preceding example, all of Beta's strategies are equivalent against Alpha's chosen optimal strategy; i.e.,

$$\sum_j a(i,j)u(j) = \mathcal{S} \tag{4}$$

for all i, where $u(j) = \alpha(j)/\sum_j \alpha(j)$ with α as in the statement of Theorem 1. There is no loss of generality in doing so, since if this is not the case, there is always a way to make it be true by diminishing some of the $a(i,j)$ without changing the value of the game (in other words, without making the integer programming problem or its LP-relaxation any easier). Informally, this amounts to reining in the slack in all the constraints where the inequality is strict for the optimal solution vector.

The proof requires a probabilistic construction. Assume without loss of generality that b_i are all equal to one, since v, v^*, and the truth of condition (1) for P and its tensor powers are all preserved by the normalization that divides each a_{ij} by b_i and sets b_i equal to one. Then (1) becomes

$$\prod_j a(i,j)^{a(i,j)u(j)} \leq 1 \tag{5}$$

for all i. For ease of exposition, assume also that the c vector is all 1's (the last paragraph of the proof handles the case of general positive c vectors). Let v_0 be any constant greater than $v^*(P)$ and let $V = \lceil v_0^n \rceil$ for n large (just how large, we will decide later). It will be shown via (12) below that $v^*(P) \geq 1$ and hence that $V/v_0^n \to 1$. To determine a set of values for the d^n variables in the n-fold tensor product of P such that the sum of the variables is V, begin with all the variables equal to zero, and then select one of them according to a certain probability distribution and increment it by 1. Repeat this V times with the choices being independent and identically distributed. It will be shown that for the correct choice of probability distribution, this procedure has a positive probability (in fact, a probability close to 1) of producing a feasible integer vector. The probability distribution is exactly the same as the probability distribution used by McEliece and Posner [8]. That is to say, the probability of choosing the variable $x(j_1, j_2, \cdots, j_n)$ is given by $u(j_1)u(j_2)\cdots u(j_n)$ where u is the optimal strategy for Alpha. The proof that this construction works, however, is more involved than the one in the paper by McEliece and Posner.

Proof of Theorem 1. Choose a vector at random according to the scheme described in the previous paragraph. The random vector will be feasible if for every constraint $C(i_1, \cdots, i_n)$ the sum of the coefficients of the V randomly chosen variables in that constraint is at least 1. For each variable $x = x(j_1, \cdots, j_n)$, the coefficient in the constraint $C = C(i_1, \cdots, i_n)$ is just the product

$$\prod_{k=1}^{n} a(i_k, j_k).$$

Note that the value of this product depends on the number of times each pair (i,j) occurs in the list of (i_k, j_k), but not on the order of these pairs, and with that in mind define the *type* of the constraint-and-variable pair (C, x) to be the matrix Y, where $Y(i,j)$ is $(1/n)$ times the number of times the pair (i,j) occurs in the list of (i_k, j_k). Also, define

$$r = r(Y) = \prod_{i,j} a(i,j)^{Y(i,j)}$$

so that the coefficient of x in C is just $r(Y)^n$.

The proof will proceed by finding for each constraint C a matrix Y for which, with high probability, the number of times we select a variable x such that (C, x) is of type Y is at least $r(Y)^{-n}$. In other words, the coefficients in C of the randomly chosen variables sum to at least $r^{-n}r^n = 1$, even if we ignore all but those variables x for which (C, x) is of one particular type. (This is less surprising than it might at first seem, since the number of types is polynomial in n, whereas all other quantities are growing exponentially; hence, in restricting to those x for which (C, x) is of a certain type, we are not losing an exponentially significant contribution.)

Fix a particular constraint $C = C(i_1, \cdots, i_n)$ and define its type β to be the vector of length m such that $\beta(i)$ is equal to $(1/n)$ times the number of times i appears in the list of the i_k. Define the m-by-d matrix

$$Z(i,j) = a(i,j)\beta(i)u(j)/\mathcal{S}. \tag{6}$$

Note that

$$\sum_j Z(i,j) = \beta(i)\left(\sum_j a(i,j)u(j)/\mathcal{S}\right) = \beta(i), \tag{7}$$

by (4). Also, note that $r = r(Z) = \prod_{i,j} a(i,j)^{a(i,j)u(j)\beta(i)/\mathcal{S}}$, which is the product over i of positive powers of quantities $\prod_j a(i,j)^{a(i,j)u(j)}$, each of which is at most 1 (by (5)); thus

$$r \leq 1. \tag{8}$$

Now define an approximation $\tilde{Z}$ to Z recursively in j by

$$\tilde{Z}(i,j) = \frac{1}{n}\lceil nZ(i,j)\rceil \quad \text{if } \sum_{j'=1}^{j-1} \tilde{Z}(i,j') \leq \sum_{j'=1}^{j-1} Z(i,j'), \quad \text{and}$$

$$\tilde{Z}(i,j) = \frac{1}{n}\lfloor nZ(i,j)\rfloor \quad \text{if} \sum_{j'=1}^{j-1} \tilde{Z}(i,j') > \sum_{j'=1}^{j-1} Z(i,j').$$

The important properties of $\tilde{Z}$ are that

(i) $Z(i,j) = 0$ implies $\tilde{Z}(i,j) = 0$;
(ii) $n\tilde{Z}(i,j)$ is an integer;
(iii) $|Z(i,j) - \tilde{Z}(i,j)| < 1/n$; and
(iv) $\sum_j \tilde{Z}(i,j) = \sum_j Z(i,j) = \beta(i)$.

They follow immediately from the definition. The reason we want conditions (i) and (iii) to hold is so that calculations involving $\tilde{Z}$ can be approximated by calculations involving Z; the reason we want conditions (ii) and (iv) is so that $\tilde{Z}$ will actually be the type of a constraint-and-variable pair (C, x) for some x.

Define

$$\tilde{r} = \tilde{r}(Z) = \prod_{i,j} a(i,j)^{\tilde{Z}(i,j)}.$$

The immediate object is to estimate the number of variables x of the V that are chosen (with repetition) for which the pair (C, x) is of the type $\tilde{Z}$, and show that this

number is very likely to be at least $\tilde{r}^{-n}$. Each time a variable $x = x(j_1, \cdots, j_n)$ is chosen, the chance that (C, x) is of the type $\tilde{Z}$ is just the chance that for each i, the values of the j_k for which $i_k = i$ form the multiset that has $n\tilde{Z}(i,1)$ ones, $n\tilde{Z}(i,2)$ twos, and so on. Denote this probability by $P(\tilde{Z})$. Then

$$P(\tilde{Z}) = \prod_i \left[\text{multi}\left(n\beta(i); n\tilde{Z}(i,1), \cdots, n\tilde{Z}(i,m)\right) \prod_j u(j)^{n\tilde{Z}(i,j)} \right],$$

where $\text{multi}(x; y_1, y_2, \cdots)$ denotes the multinomial coefficient with x on top and $y_1, y_2, \cdot$ on the bottom. Evaluate these multinomial coefficients by assuming $n \geq 3$ and by using the inequalities

$$x! > x^x e^{-x} \quad \text{and} \quad x! < nx^x e^{-x}$$

for the numerator and denominator, respectively. Here $0^0 = 1$ by convention. After all the e^x and n^x factors cancel, we obtain

$$P(\tilde{Z}) \geq n^{-md} \left[\prod_i \beta(i)^{\beta(i)} \prod_{i,j} \tilde{Z}(i,j)^{-\tilde{Z}(i,j)} \prod_{i,j} u(j)^{\tilde{Z}(i,j)} \right]^n. \tag{9}$$

Note that if we replace $\tilde{Z}$ by Z everywhere in the bracketed expression, it becomes

$$\begin{aligned} &\prod_i \beta(i)^{\beta(i)} \prod_{i,j} Z(i,j)^{-Z(i,j)} \prod_{i,j} u(j)^{Z(i,j)} \\ &= \prod_{i,j} \left(u(j)\beta(i)Z(i,j)^{-1}\right)^{Z(i,j)} \\ &= \prod_{i,j} \left(\mathcal{S}a(i,j)^{-1}\right)^{Z(i,j)} \qquad (10) \\ &= \mathcal{S} \prod_{i,j} a(i,j)^{-Z(i,j)} \\ &= \mathcal{S}/r, \end{aligned}$$

where the first equality follows from (7) and the second follows from the definition of Z in (6). We proceed to rewrite (9) in terms of $\mathcal{S}/r$. Specifically, we will approximate (9) by a version with Z replacing $\tilde{Z}$, thereby introducing an additional error factor of the form $(1 - \delta(n))^n$ with $\delta(n) \to 0$ as $n \to \infty$. By property (iii) of $\tilde{Z}$,

$$\frac{\tilde{Z}(i,j)^{-\tilde{Z}(i,j)}}{Z(i,j)^{-Z(i,j)}} \geq \inf \frac{x^x}{y^y},$$

where the infimum is taken over nonnegative x and y satisfying $|x - y| < 1/n$. Denote this infimum by $1 - \theta(n)$; since the function $x \ln(x)$ (with $0 \ln 0$ defined to be 0) is uniformly continuous on $[0,1]$, $\theta(n) \to 0$ as $n \to \infty$. Thus, putting $1 - \delta_1(n) = (1 - \theta(n))^{md}$, we get

$$\frac{\prod_{i,j} \tilde{Z}(i,j)^{-\tilde{Z}(i,j)}}{\prod_{i,j} Z(i,j)^{-Z(i,j)}} \geq (1 - \theta(n))^{md} = 1 - \delta_1(n)$$

with $\delta_1(n) \to 0$ as $n \to \infty$. Also note that

$$\frac{u(j)^{\tilde{Z}(i,j)}}{u(j)^{Z(i,j)}} = u(j)^{\tilde{Z}(i,j)-Z(i,j)},$$

which is at least $u(j)^{1/n}$ when $u(j) > 0$ and is 1 when $u(j) = 0$; either way, the fraction is at least ${u_{\min}}^{1/n}$, where $u_{\min}$ is the minimum of the positive entries of u. Thus

$$\frac{\prod_{i,j} u(j)^{\tilde{Z}(i,j)}}{\prod_{i,j} u(j)^{Z(i,j)}} \geq \left({u_{\min}}^{1/n}\right)^{md}.$$

Letting $\delta_2(n)$ be defined by $1 - \delta_2(n) = {u_{\min}}^{md/n}(1 - \delta_1(n))$, we conclude that

$$\begin{aligned} P(\tilde{Z}) &\geq n^{-md}(1-\delta_2(n))^n \left[\prod_i \beta(i)^{\beta(i)} \prod_{i,j} Z(i,j)^{-Z(i,j)} \prod_{i,j} u(j)^{Z(i,j)}\right]^n \\ &= n^{-md}(1-\delta_2(n))^n (\mathcal{S}/r)^n \ , \end{aligned} \tag{11}$$

where $\delta_2(n) \to 0$ as $n \to \infty$.

The other estimate of this sort that we will need is a bound on $\tilde{r}$ in terms of r. Take $\eta > 0$ with $\eta \leq a(i,j)$ and $\eta \leq 1/a(i,j)$ for all $a(i,j) \neq 0$. Then

$$\frac{a(i,j)^{\tilde{Z}(i,j)}}{a(i,j)^{Z(i,j)}} = a(i,j)^{\tilde{Z}(i,j)-Z(i,j)} \geq \eta^{1/n}$$

for all i, j, and

$$\begin{aligned} \tilde{r} &= \prod_{i,j} a(i,j)^{\tilde{Z}(i,j)} \\ &\geq \prod_{i,j} \eta^{1/n} \prod_{i,j} a(i,j)^{Z(i,j)} \\ &= \eta^{md/n}\ r \ . \end{aligned}$$

Then from (11) it follows that

$$P(\tilde{Z}) \geq Q(n)(\mathcal{S}/\tilde{r})^n, \tag{12}$$

where $Q(n) = (1-\delta_2(n))^n \eta^{md} n^{-md}$. (Note that $\theta, \delta_1, \delta_2$, and Q depend only on n, not on which constraint was chosen. What will end up being important about $Q(n)$ is that $\sqrt[n]{Q(n)} \to 1$ since $\delta(n) \to 0$.) Since $\mathcal{S}/r = \lim_n \mathcal{S}/\tilde{r} \leq \lim_n Q(n)^{1/n} P(\tilde{Z})^{1/n} \leq \lim_n Q(n)^{1/n} = 1$, it follows that $v^*(P) = 1/\mathcal{S} \geq 1/r \geq 1$, and the debt we incurred in the paragraph before the proof of Theorem 1 by claiming $V/v_0^n \to 1$ is paid. A more meaningful interpretation of (12) is that if we choose a variable $x = x(j_1, \cdots j_n)$ at random with probability $u(j_1) \cdots u(j_n)$, the probability that (C, x) is of type $\tilde{Z}$ is at least $Q(n)(\mathcal{S}/\tilde{r})^n$. Also, recall that if (C, x) is of type $\tilde{Z}$, then the coefficient of x in C is $\tilde{r}$.

The last step in the proof of Theorem 1 is an application of the following lemma.

LEMMA 6. *Let a, b, c, ϵ be positive real numbers with $ab/(1+\epsilon) \geq c \geq 1 \geq b$. Consider a family $\{X_i\}$ of at least a^n independently and identically distributed Bernoulli random variables with $\mathbf{P}(X_i = 1) \geq b^n$. Then there is some positive constant δ and some positive integer N for which $\mathbf{P}(\sum X_i < c^n) < e^{-e^{\delta n}}$ whenever $n > N$. Furthermore, N and δ can be chosen to depend only on ϵ.*

Assuming the lemma for the moment, the rest of the proof of Theorem 1 is as follows. We have selected $V = \lceil v_0{}^n \rceil$ variables for some $v_0 > v^*(P) = 1/\mathcal{S}$. Then for any fixed constraint $C = C(i_1, \cdots, i_n)$, we have chosen a matrix $\tilde{Z}$ and its associated value $\tilde{r}$ so that each variable chosen by our random scheme has coefficient at least $\tilde{r}^n$ with probability at least $P(\tilde{Z})$. Let $a = v_0$, $b = P(\tilde{Z})^{1/n}$, and $c = \max(1, 1/\tilde{r})$. Let X_i be the Bernoulli random variable that equals 1 if the ith variable chosen, x, has the property that (C, x) is of type $\tilde{Z}$ and equals 0 otherwise. Since $\tilde{r}$ converges to r as n gets large, and since $r \leq 1$ by (8), it follows that for any $\delta > 0$, $1/\tilde{r} > 1 - \delta$ for sufficiently large n. Then (12) implies that the first inequality in the hypothesis of Lemma 6 is satisfied with any $\epsilon < v_0\mathcal{S} - 1$ for sufficiently large n, since $ab/c \geq v_0\mathcal{S}Q(n)^{1/n}/c\tilde{r}$ and $Q(n)^{1/n}/c\tilde{r} \to 1$. The second inequality is guaranteed by the choice of c and the last is true because b is a positive power of a probability. The conclusion of the lemma is that the probability of there being enough variables of type $\tilde{Z}$ to satisfy the constraint (namely, $\tilde{r}^{-n}$ of them) is at least $1 - e^{-e^{\delta n}}$. This is true uniformly over all constraint types for sufficiently large n, and since there are only exponentially many constraints C, the sum of the failure probabilities over all constraints goes to zero as n goes to infinity. In particular, the constraints are all satisfied with nonzero probability for n sufficiently large, and that proves the theorem.

The case where the c vector is not all 1's. Suppose that $(\alpha(1), \cdots, \alpha(d))$ is an optimal solution to the program. Then letting $u(j) = \alpha(j)/\sum_j \alpha(j)$ gives a strategy in the two-player game that achieves a payoff of $1/\sum_j \alpha(j)$. Letting $\mathcal{S} = 1/\sum_j \alpha(j)$, the calculation after (7) still shows that condition (1) implies $r \leq 1$, and (12) still gives $\mathcal{S} \leq 1$. Here we have borrowed another page from matrix-game theory to assert that the optimal solution with the new c vector will, in general, have a different set of dominated strategies for Alpha (i.e., a different set of j for which $u(j) = 0$), but that each $\sum_j a(i,j)u(j)$ will still be $1/\mathcal{S}$ for all i such that strategy i is not a dominated strategy for Beta. Note that since the set of dominated strategies changes with the c vector, the modification of dominated strategies as in the paragraph before condition (1) must come after looking at the c vector. Ignore the cost vector c for the moment and use the same randomized algorithm as before to choose $\lceil (\epsilon + \sum_j \alpha(j))^n \rceil$ variables to increment, where ϵ is a new, arbitrarily small, positive number. Since the number of variables is going to infinity, the constraints are now satisfied with a probability that goes to one as $n \to \infty$. The expected total cost of the variables chosen is $\lceil (\epsilon + \sum_j \alpha(j))^n \rceil$, so the probability that the cost exceeds $(2\epsilon + \sum_j \alpha(j))^n (c^T u)^n$ goes to zero as n goes to infinity; hence, in particular, the probability that the cost is at most $(2\epsilon + \sum_j \alpha(j))^n (c^T u)^n$, and that the constraints are all satisfied that, is positive for large n. However, $c^T u = c^T \alpha / \sum_j \alpha(j)$, so, for large enough n, there are feasible integer vectors with cost at most $((2c^T\alpha/\sum_j \alpha_j)\epsilon + c^T\alpha)^n$ for arbitrarily small ϵ. The theorem is proved. □

Proof of Lemma 6. This is a standard large deviation estimate, but, to get δ to depend only on ϵ, the usual moment estimate will be redone from scratch. The following fact can easily be seen by looking at chords of the graph of $\ln(1-x)$ near

$x = 0$, below:

$$\frac{\ln(1 - zu)}{z \ln(1 - u)} \to 1 \text{ uniformly over } z \in [0, 1] \text{ as } u \downarrow 0. \tag{13}$$

(The expression is taken to be 1 when z or u is 0.) Letting $t \geq 0$ be a free parameter, the moment calculation is

$$\begin{aligned}
\mathbf{P}(\textstyle\sum X_i < c^n) &= \mathbf{P}(e^{t\sum -X_i} > e^{-tc^n}) \\
&\leq \mathbf{E}e^{t(-\sum X_i)} \,/\, e^{-tc^n} \\
&\leq (\mathbf{E}e^{-tX_1})^{a^n} / e^{-tc^n} \\
&\leq (1 - b^n(1 - e^{-t}))^{a^n} / e^{-tc^n}.
\end{aligned}$$

Exponentiating (13), we see that for all $\gamma \in (0, 1)$, u can be chosen small enough so that for any $z \in [0, 1]$, $1 - zu \leq ((1 - u)^z)^\gamma$. Then with $z = b^n$ and $u = 1 - e^{-t}$ we have that for any $\gamma \in (0, 1)$ and sufficiently small t, the following inequality holds for any $b \in [0, 1]$:

$$1 - b^n(1 - e^{-t}) \leq \left([1 - (1 - e^{-t})]^{b^n}\right)^\gamma = e^{-t\gamma b^n}.$$

Then

$$\mathbf{P}(\textstyle\sum X_i < c^n) \leq e^{-t(\gamma a^n b^n - c^n)}. \tag{14}$$

Fix any γ such that $\gamma ab > c$. The right-hand side of (14) increases when b and c are decreased by the same factor, and also when b is decreased, so assume without loss of generality that $ab/(1 + \epsilon) = c = 1$. Then the exponent in the right-hand side grows like $(1 + \epsilon)^n$, so for any $\delta \in (0, \ln(1 + \epsilon))$, there is an N for which the left-hand side of (14) is bounded by $e^{-e^{n\delta}}$ whenever $n > N$. It is clear that δ and N can be chosen to depend only on ϵ. □

4. Proof of Theorem 2. The proof of Theorem 1 made delicate use of the structure of the nth power of an integer program. In contrast, the proof presented in this section is based on very general lemmas about semi-FHC programs (defined below), and only at the very end does the notion of a product of integer programs make an appearance. Even then, we appeal only to the most basic facts about $P^{\otimes n}$—namely, that $v^*(P^{\otimes n}) = v^*(P)^n$, and that the number of constraints in $P^{\otimes n}$ grows exponentially, not faster.

We will prove the following restatement of Theorem 2 (obtained by rescaling, as in the proof of Theorem 1).

THEOREM 7. *Suppose that $P = (A, b, c)$ is an integer minimization program in which $0 \leq a_{ij} \leq 1$, $b_i = 1$, and $c_j > 0$ for all i and j. Then $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$ as $n \to \infty$.*

We will prove Theorem 7 by analyzing a somewhat broader class of programs than those that satisfy its hypothesis. Say that an integer program $P = (A, b, c)$ is of *semi-FHC type* if all of the entries of A are in $[0, 1]$ and all of the entries of c are positive (the entries of b may be arbitrary real numbers). Call a constraint trivial if it is satisfied by all real vectors x (i.e., all its coefficients are 0 and its right-hand

side is less than or equal to 0), and call a program trivial if all its constraints are trivial. Given an integer program $P = (A, b, c)$ of semi-FHC type, let m denote the number of nontrivial constraints, $S(P)$ denote the sum of the entries of b, and $D(P)$ denote $\max_j\{\sum_i a_{ij}/c_j\}$. When P is of FHC type, $S(P) = m$; when P is trivial, $S(P) \leq 0$. (The notation "$D(P)$" originates from the fact that when P is a hypergraph-covering program, $D(P)$ coincides with the maximum degree of the hypergraph, i.e., the maximum number of edges sharing a vertex.)

Our argument begins with the observation that $v^*(P) \geq S(P)/D(P)$ for any nontrivial semi-FHC program P. To see this, let y be the vector of length m, all of whose components equal $1/D(P)$. Since the jth row-sum of A^T is at most $D(P)c_j$, the jth component of the vector $A^T y$ is less than or equal to c_j for every j. Hence y is a feasible solution to the dual program $P^\perp$, whence $v^*(P) = v^*(P^\perp) \geq b^T y = S(P)/D(P)$.

In particular, suppose that P is as in Theorem 7, and Q is a nontrivial semi-FHC program such that every feasible solution to P is also feasible for Q. Then $S(Q)/D(Q) \leq v^*(Q) \leq v^*(P)$. This upper bound on the ratio $S(Q)/D(Q)$ is the key ingredient in the proof of the following fact.

LEMMA 8. *If P is as in Theorem 7 and $c_j \leq v^*(P)$ for all j, then there is a nonnegative integer vector x^* such that $c^T x^* \leq 2(\ln 10 + 1)v^*(P)$ and at least one-fourth of the entries of the column vector Ax^* exceed 1.*

Proof. We define a sequence of semi-FHC programs $P^{(0)} = P$, $P^{(1)}$, $P^{(2)}$, $\cdots$, $P^{(N)}$, where N will be specified later, together with a sequence of d-component integer vectors $u^{(0)} = 0$, $u^{(1)}$, $u^{(2)}$, $\cdots$, $u^{(N)}$ in the following iterative way. We assume that $P^{(k)} = (A^{(k)}, b^{(k)}, c)$ has already been defined, and wish to define $P^{(k+1)}$. Take j (more properly speaking, j_k) such that $\sum_i a_{ij}^{(k)}/c_j = D(P^{(k)})$, and let $u^{(k+1)}$ be the vector obtained from $u^{(k)}$ by incrementing its jth component by 1. Let $b^{(k+1)}$ equal $b^{(k)}$ minus the jth column of $A^{(k)}$. Lastly, to define $A^{(k+1)}$, call a row of $A^{(k)}$ *satisfied* if the corresponding entry of $b^{(k+1)}$ is negative; replace all the entries in all the satisfied rows of $A^{(k)}$ by 0's and call the resulting matrix $A^{(k+1)}$. Terminate this greedy procedure after N steps, where N is chosen to be the smallest integer such that $\sum_{k=0}^{N-1} c_{j_k} \geq v^*(P) \ln 10$.

Note that under this scheme, if we fix i between 1 and m and look at the ith entries of the successive vectors $b^{(0)}, b^{(1)}, b^{(2)}, \cdots, b^{(N)}$, we see a sequence of numbers that decreases by at most 1 at each stage until a negative term appears, at which point the sequence is constant (since the corresponding row of A gets "zeroed out"). Hence all the entries of all the b-vectors lie in the interval $[-1, 1]$.

Also note that a feasible solution for $P^{(k)}$ remains feasible for $P^{(k+1)}$, since the only change made in passing from the former to the latter is that certain constraints have been relaxed (some of the entries in the b-vector have decreased), while other constraints have been effectively dropped (some of the rows of the A-matrix have been zeroed out). Therefore any feasible solution for $P^{(0)} = P$ is feasible for each $P^{(k)}$. If $P^{(k)}$ is nontrivial, this implies that $S(P^{(k)})/D(P^{(k)}) \leq v^*(P)$ and hence

$$S(P^{(k+1)})/S(P^{(k)}) = \frac{S(P^{(k)}) - c_{j_k} D(P^{(k)})}{S(P^{(k)})}$$

$$= 1 - c_{j_k} \frac{D(P^{(k)})}{S(P^{(k)})}$$

$$\leq 1 - \frac{c_{j_k}}{v^*(P)}$$

$$\leq e^{-c_{j_k}/v^*(P)}$$

for all k between 0 and $N-1$. Multiplying these N inequalities together, we obtain

$$S(P^{(N)})/S(P) \leq e^{-\sum_{k=0}^{N-1} c_{j_k}/v^*(P)}$$

$$\leq e^{-\ln 10}$$

$$= \frac{1}{10} .$$

If any of the $P^{(k)}$ are trivial then so is $P^{(N)}$, so that $S(P^{(N)}) = 0$, from which we see that the foregoing inequality holds in this case as well. We have shown that the sum of the entries of $b^{(N)}$ (all of which lie between -1 and 1) is at most $\frac{1}{10}S(P) = \frac{1}{10}\sum_i b_i^{(0)} = m/10$. This means that at least a quarter of its entries are less than $\frac{1}{2}$ (since, otherwise, the average of the entries of $b^{(N)}$ would be at least $\frac{3}{4}\left(\frac{1}{2}\right) + \frac{1}{4}(-1) = \frac{1}{8} > \frac{1}{10}$, a contradiction). On the other hand, we also know that all of the entries of $b^{(0)}$ were 1's, so at least a quarter of the entries of $b^{(0)} - b^{(N)}$ must exceed $\frac{1}{2}$; since $b^{(N)} \geq b^{(0)} - Au^{(N)}$, we have $Au^{(N)} \geq b^{(0)} - b^{(N)}$, so that at least a quarter of the entries of $Au^{(N)}$ exceed $\frac{1}{2}$. Also, by the minimality of N and our assumption on c, we conclude that $c^T u^{(N)} = \sum_{k=0}^{N-1} c_{j_k} + c_{j_N} \leq v^*(P)\ln 10 + v^*(P) = (\ln 10 + 1)v^*(P)$. Setting $x^* = 2u^{(N)}$, we obtain a vector with the desired properties. □

LEMMA 9. *If P is as in Theorem 7, $S(P) > 1$, and $c_j \leq v^*(P)$ for all j, then*

$$v(P) \leq 100\, v^*(P)\, \ln S(P) .$$

Proof. Let x^* be the vector of Lemma 8 with $c^T x^* \leq 2(\ln 10 + 1)v^*(P)$ and with the property that at least one-fourth of the entries of Ax^* exceed 1. Let P' be the integer program obtained from P by dropping all the constraints that correspond to the values of i for which $(Ax^*)_i \geq 1$. Note that any feasible solution to P', if increased by adding x^*, yields a feasible solution to P; hence

$$v(P) \leq v(P') + 2(\ln 10 + 1)v^*(P) .$$

Note that P' has at most three-fourths as many constraints as P. Hence, iterating this reduction process K times, where

$$K = \lceil \log_{4/3} S(P) \rceil > \log_{4/3} S(P) ,$$

we obtain a program Q such that

$$v(P) \leq v(Q) + 2K(\ln 10 + 1)v^*(P) .$$

The number of constraints in Q, however, is at most

$$\left(\tfrac{3}{4}\right)^K S(P) < \left(\tfrac{3}{4}\right)^{\log_{4/3} S(P)} S(P) = 1;$$

that is, Q is the empty program, with no constraints and with value 0. Hence

$$v(P) \leq 2K(\ln 10 + 1)v^*(P)$$

$$= 2 \left\lceil \frac{\ln S(P)}{\ln 4/3} \right\rceil (\ln 10 + 1) v^*(P)$$

$$\leq 100\, v^*(P)\, \ln S(P),$$

as claimed, the last inequality following from the fact that $\lceil x \rceil / x$ cannot be too large when x is bounded away from zero. □

The following lemma will permit us to assume that in Theorem 7 no component of the vector c is greater than $v^*(P)$.

LEMMA 10. *If P is as in Theorem 7 and $c_k > v^*(P)$, then every optimal solution x^* to the LP-relaxation of P has $x_k^* = 0$.*

Proof. If x^* is a feasible solution for P then so is z^*, given by

$$z_j^* = \begin{cases} x_j^*(1 + x_k^*) & \text{for } j \neq k, \\ x_k^{*2} & \text{for } j = k, \end{cases}$$

since

$$\sum_j a_{ij} z_j^* = (1 + x_k^*) \sum_j a_{ij} x_j^* - a_{ik} x_k^*$$

$$\geq (1 + x_k^*) - x_k^* = 1.$$

If x^* is optimal as well, then

$$v^*(P) = \sum_j c_j x_j^*$$

$$\leq \sum_j c_j z_j^*$$

$$= \sum_{j \neq k} c_j x_j^*(1 + x_k^*) \; + \; c_k x_k^{*2}$$

$$= (1 + x_k^*) \sum_j c_j x_j^* \; + \; -c_k x_k^*$$

$$= (1 + x_k^*) v^*(P) - c_k x_k^*$$

$$= v^*(P) - (c_k - v^*(P)) x_k^*,$$

which implies that $x_k^* = 0$. □

Proof of Theorem 7. Let m be the number of constraints of P; we may suppose that $m > 1$ since the result is trivial for $m = 1$. Note that $P^{\otimes n}$ has only m^n constraints. Assume first that $c_j \leq v^*(P)$ for all j. Then all of the entries of $c^{\otimes n}$ are at most $v^*(P)^n = v^*(P^{\otimes n})$ for all n, so that by Lemma 9 we have

$$v(P^{\otimes n}) \leq 100 v^*(P^{\otimes n}) \ln(m^n) = 100 n \ln(m) v^*(P)^n,$$

from which we can conclude that $\sqrt[n]{v(P^{\otimes n})} \to v^*(P)$. On the other hand, if $c_k > v^*(P)$ for some k, then we may drop those variables x_k from the program P. This can only make $v(P^{\otimes n})$ larger, but does not affect $v^*(P)$ because of Lemma 10, so we have reduced the problem to the first case, which has just been solved. □

Acknowledgments. We would like to thank both referees for their detailed reports, which suggested many useful changes. In particular, Mark Hartmann explained to us how to modify our original proof of Theorem 7 to remove the unnecessary restriction that $c_j = 1$ for all j.

REFERENCES

[1] W. Bein and P. Brucker, *Greedy concepts for network flow problems*, Discrete Appl. Math., 15 (1986), pp. 135–144.

[2] F. M. Fekete, *Über die Verteilung der Wurzeln bei gewissen algebraischen Gleichungen mit ganzzahligen Koeffizienten*, Math. Z., 17 (1923), pp. 228–249.

[3] Z. Füredi, *Matchings and covers in hypergraphs*, Graphs Combin., 4 (1988), pp. 115–206.

[4] P. Hell and F. Roberts, *Analogues of the Shannon capacity of a graph*, Ann. Discrete Math., 12 (1982), pp. 155–162.

[5] L. Lovász, *On the Shannon capacity of a graph*, IEEE Trans. Inform. Theory, 25 (1979), pp. 1–7.

[6] ———, *On the ratio of optimal integral and fractional covers*, Discrete Math., 13 (1975), pp. 383–390.

[7] R. McEliece, *The Theory of Information and Coding*, Addison–Wesley, Reading, MA, 1977.

[8] R. McEliece and E. Posner, *Hide and seek, data storage, and entropy*, Ann. Math. Statist., 42 (1971), pp. 1706–1716.

[9] C. E. Shannon, *The zero-error capacity of a noisy channel*, IRE Trans. Inform. Theory, IT-2 (1956), pp. 8–19.

[10] J. Von Neumann and O. Morgenstern, *Theory of games and economic decisions*, 3rd ed., Princeton University Press, Princeton, NJ, 1953.

SIAM J. DISCRETE MATH.
Vol. 5, No. 1, pp. 144–149, February 1992
©1992 Society for Industrial and Applied Mathematics
012

PAIR LABELINGS OF GRAPHS *

DAVID R. GUICHARD† AND JOHN W. KRUSSEL‡

Abstract. Given a graph G and positive integer d, the pair-labeling number $r^\star(G,d)$ is the minimum n such that each vertex in G can be assigned a pair of numbers from $\{0,1,\cdots,n-1\}$ so that any two numbers used at adjacent vertices differ by at least d modulo n. All possible values of $r^\star(G,d)$, given the chromatic number of G, are determined.

Key words. pair labeling, star chromatic number

AMS(MOS) subject classifications. 05C15

1. Introduction. Füredi, Griggs, and Kleitman [FGK] have recently examined the *pair labeling number* $r(G,d)$, which is the minimum n such that each vertex in G can be assigned a pair of numbers from $\{0,\cdots,n-1\}$ so that any two numbers used at adjacent vertices differ by at least d. They discovered all possible values for $r(G,d)$ when the chromatic number of G is fixed at k, namely, $r(G,d) \in [d(k-1)+3, d(k-1)+k+1]$ when $k \geq 2$, with all values in this interval realized. The problem was motivated by an application involving assignment of channel frequencies without interference. From a purely mathematical point of view, we think it somewhat more natural to consider the minimum n such that each vertex in G can be assigned a pair of numbers from $\{0,\cdots,n-1\}$ so that any two numbers used at adjacent vertices differ by at least d modulo n. We denote this minimum by $r^\star(G,d)$. Our main result is analogous to that of [FGK] but the interval is $[d(k-1)+3, dk+k]$.

Our notation is largely standard and similar to that of [FGK]. Sets given in interval notation $[a,b]$ are subsets of the integers. For a set S and nonnegative integer t, $\binom{S}{t}$ is the collection of t-subsets of S. All graphs $G=(V,E)$ are simple and undirected. We use $\mathbf{N}$ to denote the nonnegative integers.

A *pair labeling* of a graph G is a function $f: V \to \binom{\mathbf{N}}{2}$ from the vertices of G to pairs of integers. We refer to two measures of the distance between pairs of integers A, B. The distance $d(A,B)$ is the minimum value of $|a-b|$ over all $a \in A$ and $b \in B$; the distance $d_n^\star(A,B)$ is the minimum value of $|a-b|_n$ over all $a \in A$ and $b \in B$, where $|a-b|_n$ is the minimum of $|a-b|$ and $n-|a-b|$ (in this case, we always assume that $a,b \in [0,n-1]$). We usually write $d^\star$ with n to be understood. For a pair labeling f of G we define $d(f)$ to be the minimum value of $d(f(v),f(w))$ over all adjacent vertices v and w, and similarly $d_n^\star(f)$. Finally, we define $r(G,d)$ to be the minimum n such that there is an $f: V \to \binom{[0,n-1]}{2}$ with $d(f) \geq d$ and $r^\star(G,d)$ to be the minimum n such that there is an $f: V \to \binom{[0,n-1]}{2}$ with $d_n^\star(f) \geq d$. Note that when $d=1$ there is no difference between the two notions of distance (two pairs have distance at least one if and only if they are disjoint), so we assume henceforth that $d \geq 2$.

Our main result is the following theorem.

THEOREM. *Suppose that G has chromatic number k. If $k=1$, then $r^\star(G,d)=2$. If $k=2$, then $r^\star(G,d)=2d+2$. If $k=4$, then $r^\star(G,d) \in [3d+4, 4d+4]$. Otherwise,*

*Received by the editors April 10, 1990; accepted for publication (in revised form) December 14, 1990.

†Mathematics Department, Whitman College, Walla Walla, Washington 99362.

‡Department of Mathematics, Lewis and Clark College, Portland, Oregon 97219.

$r^\star(G,d) \in [(k-1)d+3, kd+k]$. *When $k \geq 3$, all values in the stated intervals are realized by suitable graphs G.*

2. Star chromatic number. We summarize here some results of Vince [V] for later use. Suppose that f is an n-coloring of a connected graph G, that is, a function from the vertices of G to $[0, n-1]$, and define $c_n(f)$ to be the minimum over all adjacent v and w of $|f(v)-f(w)|_n$. Define the star chromatic number of G, $\chi^\star(G)$, to be the minimum over all n and all n-colorings f of $n/c_n(f)$. Vince showed that the minimum exists and indeed is attained for some n at most the number of vertices in G.

We will use two results proved in [V]. First, for all connected G, $\chi(G)-1 < \chi^\star(G) \leq \chi(G)$. Define $G_{n,m}$ to be the graph whose vertices are the integers in $[0, n-1]$ and with v adjacent to w if and only if $|v-w|_n \geq m$. Vince proved that for $1 \leq m < n/2$, $\chi^\star(G_{n,m}) = n/m$.

3. Easy lemmas. The cases $k = 1$ and 2 are easy, and we discuss them no further. For $k \geq 3$, the upper bound $kd+k$ is easy to establish. Suppose that the k-chromatic graph G is colored with k colors $\{0, \cdots, k-1\}$. Replace color i by the pair $\{i(d+1), i(d+1)+1\}$ thus labelling the graph with the pairs $\{0,1\}, \{d+1, d+2\}, \cdots, \{(k-1)(d+1), (k-1)(d+1)+1\}$, all at distance at least d mod $(kd+k)$ from each other. For $G = K_k$, it is easy to see that no $n < kd+k$ suffices, so the upper bound is sharp.

Recall that a graph homomorphism from $G = (V, E)$ to $H = (W, F)$ is a function $h: V \to W$, which preserves adjacency. Vertex labeling problems can often be stated in terms of homomorphisms; for example, G has chromatic number at most k if and only if there is a homomorphism from G to K_k, and if G is homomorphic to H then $\chi(G) \leq \chi(H)$. Note also that if G is homomorphic to H then $\chi^\star(G) \leq \chi^\star(H)$. Define the graph $G^\star(n,d)$ to have vertex set $\binom{[0,n-1]}{2}$ and an edge connecting two vertices v, w if and only if $d_n^\star(v,w) \geq d$. It follows immediately from the definitions that $r^\star(G,d) \leq n$ if and only if G is homomorphic to $G^\star(n,d)$.

To establish the lower bound when $k \neq 4$, we must show that any graph G with $r^\star(G,d) \leq d(k-1)+2$ has chromatic number at most $k-1$. For this, it is sufficient to show that the chromatic number of $G^\star(d(k-1)+2, d)$ is at most $k-1$, for if $r^\star(G,d) \leq d(k-1)+2$ then G is homomorphic to $G^\star(d(k-1)+2, d)$ and so G has chromatic number at most $k-1$. When $k = 4$ we do the same with $G^\star(d(k-1)+3, d)$.

LEMMA. *The chromatic number of $G^\star(d(k-1)+2, d)$ is at most $k-1$.*

Proof. We indicate how to color $G^\star(d(k-1)+2, d)$ with $k-1$ colors. Consider the vertex $v = \{i, j\}$ with $0 \leq i < j \leq d(k-1)+1$. Color v with color $m \leq k-2$ if $md \leq i < (m+1)d$ and color vertex $\{d(k-1), d(k-1)+1\}$ with color $k-2$. Any two vertices with the same color are at distance $d-1$ or less, so this is a proper coloring of the graph. □

LEMMA. *The chromatic number of $G^\star(3d+3, d)$ is at most 3.*

Proof. Consider the vertex $v = \{i, j\}$. If $v \cap [1, d]$ is nonempty, color v with color 0. If $v \cap [d+2, 2d+1]$ is nonempty, and v is not color 0, color it with color 1. If $v \cap [2d+3, 3d+2]$ is nonempty and v is not colored, color it with color 2. Color $\{0, d+1\}$ color 0, $\{d+1, 2d+2\}$ color 1 and $\{2d+2, 0\}$ color 2. It is easy to check that this is a proper coloring. □

To show that the lower bound is sharp, we will prove that the chromatic number of $G^\star(d(k-1)+3, d)$ is at least k for $k \neq 4$ and that the chromatic number of $G^\star(3d+4, d)$ is at least 4. This will be our major task. Note that this implies also that $r^\star(G^\star(d(k-1)+3, d), d) = d(k-1)+3$ for $k \neq 4$, since otherwise $G^\star(d(k-1)+3, d)$

is homomorphic to $G^\star(d(k-1)+2,d)$ and so has chromatic number less than or equal to $k-1$; similarly for $k=4$.

It remains to be seen that all values in the intervals are realized by some graphs G. For $k \geq 3$ and $i \in [d(k-1)+4, d(k-1)+k+1]$, [FGK] found graphs G_i with chromatic number k and $r(G_i,d)=i$ and containing a vertex adjacent to all the other vertices in the graph. It is easy to see that $r^\star(G_i,d)=i+d-1$, so that all values in $[dk+3, dk+k]$ are realized.

This leaves the remainder of the interval, $[d(k-1)+4, dk+2]$ or $[d(k-1)+5, dk+2]$ when $k=4$. By the main lemma of §4 $\chi(G^\star((k-1)d+3,d)) \geq k$ when $k \neq 4$ and by the first lemma in §3 $\chi(G^\star(kd+2,d)) \leq k$. Since $G^\star(n-1,d)$ is a subgraph of $G^\star(n,d)$ this implies that $\chi(G^\star(n,d))=k$ when n is in $[d(k-1)+3, dk+2]$. Similarly, $\chi(G^\star(n,d))=4$ when n is in $[3d+4, 4d+2]$. Thus it seems natural to try to show that $r^\star(G^\star(n,d),d)=n$. For this, it suffices to show that some graph G (regardless of chromatic number) has $r^\star(G,d)=n$, for then G is homomorphic to $G^\star(n,d)$ but not to $G^\star(n-1,d)$ and so $G^\star(n,d)$ cannot be homomorphic to $G^\star(n-1,d)$. In fact, since all values in $[d(k-1)+3, d(k-1)+k-1]$ are attained (by the argument of the preceding paragraph) we are only missing the intervals $[d(k-1)+k, dk+2]$ (which are empty when $k > d+2$).

LEMMA. *If $n > 2d+2$ is not $d \pmod{d+1}$, then $r^\star(G_{n,d+1},d)=n$.*

Proof. To see that $r^\star(G_{n,d+1},d) \leq n$, label vertex v with the pair $\{v, v+1\}$. For the other direction, suppose (for a contradiction) that $G_{n,d+1}$ is homomorphic to $G^\star(n-1,d)$. Every vertex in $G_{n,d+1}$ is contained in an $\lfloor n/(d+1) \rfloor$ clique. We claim that a vertex $\{i,j\}$ in $G^\star(n-1,d)$ is not contained in such a clique if $|i-j|_{n-1} \geq d$. For if $|i-j|_{n-1} \geq d$, then $\{i,j\}$ can be contained in no clique larger than $1+\lfloor (n-1-2d)/(d+1) \rfloor$, which is less than $\lfloor n/(d+1) \rfloor$ if n is not $d \bmod d+1$. This implies that the image of each vertex of $G_{n,d+1}$ is a vertex $\{i,j\}$ with $|i-j|_{n-1} < d$.

Denote the image of vertex v by $\{i_v, j_v\}$ with $0 < j_v - i_v < d$ or $n-1-d < i_v - j_v < n-1$. If v and w are adjacent in $G_{n,d+1}$, then the pairs $\{i_v, j_v\}$ and $\{i_w, j_w\}$ are at distance at least $d \bmod n-1$ and so the integers i_v and i_w are at distance at least $d+1 \bmod n-1$. This means that the map $v \to i_v$ is a homomorphism from $G_{n,d+1}$ to $G_{n-1,d+1}$, contradicting the fact that $\chi^\star(G_{n,d+1}) > \chi^\star(G_{n-1,d+1})$. □

This lemma takes care of all the missing values for n except $3d+2$—this is the only n which is both $d \pmod{d+1}$ and contained in an interval $[d(k-1)+k, dk+2]$. Careful examination of the proof of the lemma reveals that $r^\star(G_{3d+2,d+1},d)=3d+2$. The proof is essentially unchanged if we know that no vertex in $G_{n,d+1}$ maps to $\{i,j\}$ with $|i-j|_{n-1} \geq 2d$. However, when $n-1=3d+1$ no two integers i,j can be as much as $2d$ apart.

4. The main lemma. We are now ready to prove our principal lemma, finishing the proof of the theorem.

LEMMA. *When $k \neq 3$, $\chi(G^\star(kd+3,d)) \geq k+1$, and $\chi(G^\star(3d+4,d)) \geq 4$.*

Proof. The cases $k=2$ and $k=3$ are quite easy, and we do them first. The rest of the proof is done by induction on k, beginning with $k=4$ as the basis of the induction.

When $k=2$, $G^\star(2d+3,d)$ contains a cycle of length $2d+3$, namely $\{0,1\}$, $\{d+1,d+2\}, \{2d+2,0\}, \cdots, \{d+2,d+3\}, \{0,1\}$, so its chromatic number is at least 3, as desired.

Now suppose that $k=3$. The graph $G^\star(3d+4,d)$ contains a subgraph isomorphic to $G_{3d+4,d+1}$, namely, the graph induced by the vertices $\{i, i+1 \bmod 3d+4\}$. This subgraph has star chromatic number $(3d+4)/(d+1)$ so its chromatic number is 4,

and therefore $G^*(3d+4, d)$ has chromatic number at least 4, as desired.

For the rest of the proof, it is convenient to adopt the point of view used in [FGK]. We think of the pairs of integers from $[0, n-1]$ as the edges of the graph K_n, and we are interested in coloring the edges of K_n so that edges which are distance d or more apart (as measured by d^*) are different colors. We picture K_n with all of its vertices on a circle and numbered consecutively around the circle. The edges $\{i, i+1\}$ we call *rim edges*, others we call diagonals. (We should, of course, write "$i+1 \bmod n$," but we will continue to leave out the mod—all arithmetic on vertices is mod n.)

At most $d+1$ rim edges may be colored any one color, and if this many are a single color, they must be consecutive on the circle. If the edges of K_n are colored with k colors, there must be at least 3 color classes containing $d+1$ rim edges.

Our proof is by induction. Suppose that $n = 4d+3$ and that the edges of K_n are properly colored with 4 colors. The color classes restricted to the rim edges must consist of three classes of $d+1$ consecutive edges and one class of d consecutive edges. Suppose that the vertices that lie between these color classes on the rim are i, $i+d+1$, $i+2d+2$, and $i+3d+3$. The diagonal $\{i, i+2d+2\}$ is at distance d from $\{i+d, i+d+1\}$, $\{i+d+1, i+d+2\}$, and $\{i+3d+2, i+3d+3\}$, and must therefore be the same color as the edges in the color class containing the d rim edges between $i+3d+3$ and i. Similarly, the diagonal $\{i+d+1, i+3d+3\}$ must be this same color. These two diagonals, however, are at distance d from each other so they must be different colors, a contradiction. This establishes the lemma for $k=4$.

Now suppose that $n = kd+3$, $k > 4$, and that the lemma is true at $k-1$. For a contradiction, suppose that the edges of K_n have been properly colored with k colors. We will refer to a color class restricted to rim edges as a *rim class.*

CLAIM 1. *For any rim class consisting of $d+1$ consecutive edges of color A and forming a path from vertex i to vertex $i+d+1$, we can adjust the coloring of K_n so that all edges of color A meet one of the vertices $\{i+1, i+2, \cdots, i+d\}$, except the edge $\{i, i+d+1\}$, and so that all edges that do meet one of these interior vertices are colored A.*

First, color all such edges color A. Then there may be edges like e in Fig. 1, which are also colored A but which are far from some of these newly colored edges. It is easy to see, however, that e can be colored the same color as edge f (or edge g if e meets vertex i) since edge e is distance at most $d-1$ from any edge which meets i.

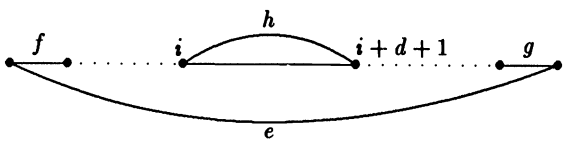


FIG. 1

If edge h is not color A then the d vertices $\{i+1, \cdots, i+d\}$ may be removed, leaving a complete graph on $(k-1)d+3$ vertices whose edges are properly colored with $k-1$ colors, a contradiction. This proves the claim. □

CLAIM 2. *Any rim class that does not consist of either d or $d+1$ consecutive edges cannot be adjacent to a rim class of $d+1$ edges that have color A.*

For a contradiction, assume such a rim class does exist. Without loss of generality suppose that the rim class colored A consists of the path $\{i, \cdots, i+d+1\}$ and the other rim class contains $\{i-1, i\}$. Adjust the coloring so that the A color class consists

of the rim edges, the edge h, and the diagonals meeting the interior of the rim class, as above. Adjacent to the rim class A will be some number (at most $d-1$) of consecutive edges of color B, then an edge of color C; see Fig. 2. (Vertex $i-d$ may be the left endpoint of the edge colored C.)

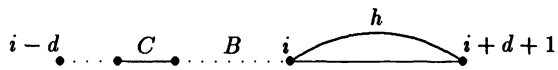


FIG. 2

Now if we color edge h and all edges incident at i which are color C with color B, and color all edges incident at $i-d$ which are color B with color C, then we still have a proper coloring of the edges. As above, removing the vertices $\{i+1,\cdots,i+d\}$ leaves a complete graph on $(k-1)d+3$ vertices whose edges are properly colored with $k-1$ colors, a contradiction. This proves the claim. □

CLAIM 3. *All rim classes consist of d or $d+1$ consecutive edges.*

Suppose not. Then we have somewhere a rim class of size $d+1$ and color A adjacent to one or more consecutive rim classes consisting of d consecutive edges, the last of which is adjacent to a class not of either type.

We will alter the coloring. Start by altering the coloring so that the edges of color A are precisely the edges which meet the interior of the rim class plus the edge joining the endpoints of the rim class. Suppose that the adjacent rim class consists of d edges of color A'. Every edge of color A' must meet one of the $d+1$ vertices that form this path, and it is therefore safe to color all edges that meet the $d-1$ interior vertices of the path with color A'. The same argument may be repeated for each of the rim classes of d consecutive edges in turn.

We now have a situation almost identical to that of Fig. 2, except that the rim class to the right of vertex i consists of only d consecutive edges; see Fig. 3. Using the same argument, we may color all edges incident at i that are color A or C with color B, and color all edges incident at $i-d$ that are color B with color C and still have a proper coloring of the edges. Once more, removing the vertices $\{i+1,\cdots,i+d\}$ leaves a complete graph on $(k-1)d+3$ vertices whose edges are properly colored with $k-1$ colors, a contradiction. This proves the claim. □

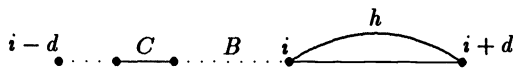


FIG. 3

Now we may assume that all diagonals that meet the interior of one or two rim classes are colored with the color of one of those rim classes, and that a diagonal joining the endpoints of a rim class of size $d+1$ is colored to match the rim class. Also, we may assume that each endpoint of each rim class of size d meets at least one diagonal whose other endpoint is outside of the rim class but which is colored the

color of the rim class, since, if not, we may remove d of the $d+1$ vertices in the rim class for a contradiction as above.

We refer to diagonals that join the endpoints of rim classes as *main diagonals.* Except for the diagonal joining the endpoints of the rim class, exactly one main diagonal at each end of a rim class of size d is colored to match that rim class—at least one as noted in the previous paragraph, and at most one, since they must *meet* at their other endpoints to be within distance $d-1$ of each other.

There may be fewer than k colors actually used on the rim, in which case there are some *extra colors* that appear only on main diagonals. Either the diagonals colored with extra color A meet at a single vertex, or there are three of them that form a triangle. If the former is the case, then we may remove the common vertex and any $d-1$ other vertices to form a complete graph on $(k-1)d+3$ vertices whose edges are properly colored with $k-1$ colors, a contradiction. We conclude that three edges are colored with each extra color. If $d \geq 3$, we may remove the three endpoints of these edges and enough other vertices to make a total of d vertices removed, producing a complete graph on $(k-1)d+3$ vertices whose edges are properly colored with $k-1$ colors, a contradiction.

Thus, either $d=2$ or there are no extra colors. Let R, $R \leq k$, be the number of colors that are used on the rim and $D=(d+1)R-kd-3$ the number of rim classes of size d. The number of main diagonals which can be colored is at most $3(k-R)+2D+R$ (that is, $3(k-R)$ edges colored with extra colors, $2D$ edges colored with the color of a rim class of size d and R edges joining the endpoints of a single rim class to each other), while the number of main diagonals that must be colored is $\binom{R}{2}$. We would like to know, therefore, that the former is less than the latter, which means that $0 < R^2 - R(1+4d) + 12 + 4kd - 6k$. If there are no extra colors, $R=k$ and this becomes $0 < k^2 - 7k + 12$ which is true when $k > 4$. Otherwise, $d=2$ and the question is whether $0 < R^2 - 9R + 2k + 12$. This quadratic in R has discriminant $33 - 8k$, which is negative when $k > 4$, so the inequality is established, the main diagonals cannot be colored, and this contradiction finishes the proof of the main lemma.

Acknowledgment The second author would like to thank Whitman College, where the bulk of this research was done, for its hospitality.

REFERENCES

[FGK] Z. Füredi, J. R. Griggs, and D. J. Kleitman, *Pair labellings with given distance*, SIAM J. Discrete Math., 2 (1989), pp. 491–499.

[V] A. Vince, *Star chromatic number*, J. Graph Theory, 12 (1988), pp. 551–559.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 151–162, May 1992
© 1992 Society for Industrial and Applied Mathematics

ROUTING WITH POLYNOMIAL COMMUNICATION-SPACE TRADE-OFF*

BARUCH AWERBUCH† AND DAVID PELEG‡

Abstract. This paper presents a family of memory-balanced routing schemes that use relatively short paths while storing relatively little routing information. The quality of the routes provided by a scheme is measured in terms of their *stretch*, namely, the maximum ratio between the length of a route connecting some pair of processors and their distance. The *hierarchical schemes* $\mathcal{H}_k$ (for every integer $k \geq 1$) presented in this paper guarantee a stretch factor of $O(k^2)$ on the length of the routes and require storing at most $O(k \cdot n^{1/k} \cdot \log n \log D)$ bits of routing information per vertex in an n-processor network with diameter D. The schemes are name independent and applicable to general networks with arbitrary edge weights. This improves on previous designs whose stretch bound was exponential in k.

The proposed schemes are based on a new efficient solution to a certain graph-theoretic problem concerning sparse graph covers. The new cover technique has already found several other applications in the area of distributed computing.

Key words. communication networks, routing tables, communication-space trade-offs, graph covers

AMS(MOS) subject classifications. 68Q22, 68R10, 05C85, 05C70

1. Introduction.

Delivering messages between pairs of processors is a basic activity of any distributed communication network. This task is performed using a routing scheme, which is a mechanism for routing messages in the network. The routing mechanism can be invoked at any origin vertex and be required to deliver a message to some destination vertex.

Using edge lengths to reflect transmission costs and delays, we naturally desire to route messages along paths that are as short as possible. A straightforward approach to this goal is to store a complete routing table in each vertex v in the network (specifying for each destination u the first edge along some shortest path from v to u). This approach clearly guarantees optimal routes, but is too expensive for large systems since it requires a total of $O(n^2 \log n)$ memory bits in an n-processor network. Thus a major problem in large-scale communication networks is the design of routing schemes that produce efficient routes and have relatively low memory requirements. The efficiency of a routing scheme is measured in terms of its *stretch*, namely, the maximum ratio between the length of a route produced by the scheme for some pair of processors and their distance.

The problem of managing the trade-off between efficiency and memory in routing schemes was first raised in [KK1]. The solution given there and in several subsequent papers [BJ], [KK2], [Pr], [S], [FJ1], [FJ2], [FJ3], [SK], [vLT1], [vLT2] applies only under some special assumptions or for special classes of network topologies. In [PU] the problem is dealt with for general networks. The paper presents a family of hierarchical routing schemes (for every integer $k \geq 1$) that guarantee stretch $O(k)$ and require storing

* Received by the editors April 16, 1990; accepted for publication (in revised form) April 2, 1991.

† Department of Mathematics and Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139. This author's research was supported by Air Force contract TNDGAFOSR-86-0078, by Army Research Office contract DAAL03-86-K-0171, by National Science Foundation contract CCR8611442, by Defense Advanced Research Projects Agency contract N00014-89-J-1988, and by a special grant from IBM.

‡ Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehovot 76100, Israel. This author's research was partially supported by an Allon Fellowship, by a Bantrell Fellowship, and by a Walter and Elise Haas Career Development Award. Part of the work was done during a visit at IBM T. J. Watson Research Center.

a total of $O(k^3 n^{1+1/k} \log n)$ bits of routing information in the network. This behavior is almost optimal, as implied by a lower bound given in [PU] on the space requirement of any scheme with a given stretch.

While the method of [PU] is appropriate for designing a static routing scheme, it is not suitable for situations in which the routing scheme needs to be updated occasionally. Such updates may be needed when the topology of the network changes (due to failures, recoveries or other reasons). Each such update involves recomputing the edge costs in the network and deciding on the new routes accordingly. (In this paper, we ignore the issue of determining the link costs and concentrate on the step of setting up a routing table.)

In such situations, it is necessary that the routing scheme can handle arbitrary edge costs (as well as arbitrary network topologies). Another crucial requirement is *name independence*. By this, we mean that the addresses used to describe the destination of messages should be permanently fixed and independent of the routing scheme. This precludes routing strategies in which the route design also involves selecting an addressing label for each vertex (typically, encrypting partial information necessary for determining the routes leading to it). This approach clearly violates the principle that routing-related system activities must be transparent to the user. Finally, in the nonstatic case, a bound on the total (or average) memory requirements is insufficient. This is because, in such setting, vertices may play different roles and thus require varying amounts of space. For instance, a vertex may be designated as a "communication center" or may just by chance be the crossing point of many routes in a particular scheme. This forces every vertex to have sufficient memory for performing the most demanding role it may ever need to perform. It is thus necessary to guarantee a bound on the worst-case memory requirements of each vertex. (See [ABLP] for a more detailed discussion of these issues.)

Unfortunately, the routing strategy of [PU] lacks all of these three properties. It deals only with unit-cost edges, it is name dependent, and there is no bound on the worst-case memory requirements of each vertex. Consequently, these problems are tackled in [ABLP], [P2]. The schemes proposed in these two papers succeed in achieving these desirable properties, but at the cost of an inferior efficiency-space trade-off. In an n-processor network of diameter D, the schemes HS_k of [ABLP], for $k \geq 1$, use $O(k \cdot \log n \cdot n^{1/k})$ bits of memory per vertex and guarantee a stretch of $O(k^2 \cdot 9^k)$, while the schemes R_k of [P2], for $k \geq 1$, use $O(\log D \cdot \log n \cdot n^{1/k})$ bits of memory per vertex and guarantee a stretch of $O(4^k)$. Thus the stretch becomes exponential in k, in contrast with the linear dependency achieved in [PU].[1]

The schemes presented in this paper are basically the same as those of [P2], except that they employ a new, more efficient construction for the underlying graph-theoretic structure (to be described later) and thus improve the trade-off. In particular, for every graph G and every integer $k \geq 1$, we construct a hierarchical routing scheme $\mathscr{H}_k$ with stretch $O(k^2)$ using $O(k \cdot n^{1/k} \cdot \log n \log D)$ memory bits per vertex. Thus the new scheme regains the polynomial dependency of the stretch factor on k. (Note that in all the results discussed here, the "interesting" range is $1 \leq k \leq \log n$; for $k > \log n$, the stretch degrades with no compensation in memory requirements.)

Let us comment that the schemes of [ABLP] still have two advantages over the new schemes. First, their space complexity is purely combinatorial; i.e., it does not depend on the edge weights. This may be significant if we allow superpolynomial edge weights. Second, that scheme has an efficient distributed preprocessing algorithm for setting the routes, while the preprocessing stage of the current schemes, as described here, is sequential.

[1] These bounds are presented in a slightly different form in [ABLP], [P2]; for comparison, we unify presentation here by fixing the exponent of the n factor in the memory bounds to be $1/k$.

More efficient distributed preprocessing algorithms and faster constructions are further developed in [AP4], [LS].

The schemes of [P2] and this paper are based on solving a certain graph problem (handled previously in [P1]) involving the construction of sparse covers. This problem (described in detail in § 3) addresses some basic issues concerning network decomposition strategies, and the new cover algorithm described here has already found several other useful applications in the area of distributed network algorithms. For some of these applications, see [AR], [AKP], [AP1], [AP2], [AP3], [AP5], [P2].

The paper is organized as follows. Section 2 formally defines the problem. Section 3 gives the new solution to the sparse cover problem. Finally, § 4 presents the hierarchical routing scheme.

2. Definition of the problem. We consider the standard model of a point-to-point communication network, described by an undirected graph $G = (V, E)$, $V = \{1, \cdots, n\}$. The vertices V represent the processors of the network, and the edges E represent bidirectional communication channels between the vertices. A vertex may communicate directly only with its neighbors, and messages between nonadjacent vertices are sent along some path connecting them in the network.

We assume the existence of a *weight* function $w : E \to \mathcal{R}^+$, assigning an arbitrary positive weight $w(e)$ to each edge $e \in E$. Also, there exists a *name* function name : $V \to U$, which assigns to each vertex $v \in V$ an arbitrary name name (v) from some ordered universe U of names. We sometimes abuse notation, referring to name (v) simply as v.

For two vertices u, w in a graph G, let $\text{dist}_G(u, w)$ denote the (weighted) length of a shortest path in G between those vertices, i.e., the cost of the cheapest path connecting them, where the cost of a path $(e_1, \cdots, e_s)$ is $\sum_{1 \leq i \leq s} w(e_i)$. (We sometimes omit the subscript G where no confusion arises.)

A *routing scheme RS* for the network G is a mechanism for delivering messages in the network. It can be invoked at any origin vertex u and be required to deliver a message M to some destination vertex v (which is specified by its *name*) via a sequence of message transmissions.

We now give precise definitions for our complexity measures for memory and stretch. The *memory requirement* of a protocol is the maximum amount of memory bits used by the protocol in any single processor in the network. The *communication cost* of transmitting a message over edge e is the weight $w(e)$ of that edge. The communication cost of a *protocol* is the sum of the communication costs of all message transmissions performed during the execution of the protocol. Let $C(RS, u, v)$ denote the communication cost of the routing scheme when invoked at an origin u, with respect to a destination v and an elementary ($O(\log n)$ bit) message, i.e., the total communication cost of all message transmissions associated with the delivery of the message. Given a routing scheme RS for an n-processor network $G = (V, E)$, we say that RS *stretches* the path from u to v by $C(RS, u, v)/\text{dist}(u, v)$. We define the *stretch factor* of the scheme RS as

$$\text{Stretch}(RS) = \max_{u,v \in V} \left\{ \frac{C(RS, u, v)}{\text{dist}(u, v)} \right\}.$$

Next, let us define some basic graph notation. The *j-neighborhood* of a vertex $v \in V$ is defined as

$$N_j(v) = \{ w \mid \text{dist}(w, v) \leqq j \}.$$

Given a subset of vertices $R \subseteq V$, denote the *m-neighborhood cover* of R by

$$\mathcal{N}_m(R) = \{ N_m(v) \mid v \in R \}.$$

Denote the *diameter* of the network by

$$D = \text{Diam}(G) = \max_{u,v \in V} (\text{dist}(u,v)).$$

For a vertex $v \in V$, let

$$\text{Rad}(v,G) = \max_{w \in V} (\text{dist}_G(v,w)).$$

Let Rad (G) denote the *radius* of the network, i.e.,

$$\text{Rad}(G) = \min_{v \in V} (\text{Rad}(v,G)).$$

A *center* of G is any vertex v realizing the radius of G (i.e., such that Rad (v, G) = Rad (G)). To simplify some of the following definitions, we avoid problems arising from 0-diameter or 0-radius graphs by defining Rad (G) = Diam $(G) = 1$ for the single-vertex graph $G = (\{v\}, \varnothing)$. Observe that for every graph G, Rad $(G) \leqq$ Diam $(G) \leqq 2$ Rad (G). (Again, in all of the above notations, we may sometimes omit the reference to G where no confusion arises.)

Finally, let us introduce some definitions concerning covers. Given a set of vertices $S \subseteq V$, let $G(S)$ denote the subgraph induced by S in G. A *cluster* is a subset of vertices $S \subseteq V$ such that $G(S)$ is connected. We use Rad (v, S) (respectively, Rad (S), Diam (S)) as a shorthand for Rad $(v, G(S))$ (respectively, Rad $(G(S))$, Diam $(G(S))$). A *cover* is a collection of clusters $\mathcal{S} = \{S_1, \cdots, S_m\}$ such that $\cup_i S_i = V$. The number of clusters in the cover $\mathcal{S}$ is denoted by $|\mathcal{S}|$. Given a collection of clusters $\mathcal{S}$, let Diam $(\mathcal{S})$ = $\max_i$ Diam (S_i) and Rad $(\mathcal{S})$ = $\max_i$ Rad (S_i). For every vertex $v \in V$, let $\deg_{\mathcal{S}}(v)$ denote the degree of v in the hypergraph $(V, \mathcal{S})$, i.e., the number of occurrences of v in clusters $S \in \mathcal{S}$. The *maximum degree* of a cover $\mathcal{S}$ is defined as

$$\Delta(\mathcal{S}) = \max_{v \in V} \deg_{\mathcal{S}}(v).$$

Given two covers $\mathcal{S} = \{S_1, \cdots, S_m\}$ and $\mathcal{T} = \{T_1, \cdots, T_k\}$, we say that $\mathcal{T}$ *subsumes* $\mathcal{S}$ if, for every $S_i \in \mathcal{S}$, there exists a $T_j \in \mathcal{T}$ such that $S_i \subseteq T_j$.

3. Constructing a sparse cover. The main novel graph-theoretic tool on which we base our constructions is the following theorem, to be proved in this section.

THEOREM 3.1. *Given a graph $G = (V, E)$, $|V| = n$, a cover $\mathcal{S}$, and an integer $k \geqq 1$, it is possible to construct a cover $\mathcal{T}$ that satisfies the following properties*:

(1) *$\mathcal{T}$ subsumes $\mathcal{S}$,*

(2) Rad $(\mathcal{T}) \leqq (2k-1)$ Rad $(\mathcal{S})$, *and*

(3) $\Delta(\mathcal{T}) \leqq 2k|\mathcal{S}|^{1/k}$.

Let us remark that it is possible to replace the degree bound of property (3) with $O(k \cdot n^{1/k})$. This requires a more complex algorithm and analysis, and therefore we prefer to state the theorem as above. In most of the applications of which we are aware, there is no real difference, as $|\mathcal{S}| = n$.

The two previously known results of this type, described in [P1], [P2], are both weaker than the current one. In the first, the bound on the radius of the output cover $\mathcal{T}$ is exponential in k (specifically, Rad $(\mathcal{T}) \leqq 4^k$ Rad $(\mathcal{S})$). The other result features a polynomial radius bound, but it bounds only the *average* degrees of vertices in the output cover $\mathcal{T}$.

3.1. The construction of the cover. The problem of constructing a subsuming cover is handled by reducing it to the subproblem of constructing a *partial cover*. The input of this problem is a graph $G = (V, E)$, $|V| = n$, a collection of (possibly overlapping)

clusters $\mathscr{R}$, and an integer $k \geq 1$. The output consists of a collection of *disjoint* clusters $\mathscr{DT}$ that subsume a subset $\mathscr{DR} \subseteq \mathscr{R}$ of the original clusters. The goal is to subsume "many" clusters of $\mathscr{R}$, while maintaining the radii of the output clusters in $\mathscr{DT}$ relatively small. We now describe a procedure `Cover` ($\mathscr{R}$), achieving this goal. The formal code for the procedure is given in Fig. 1.

The general structure of procedure `Cover` ($\mathscr{R}$) is similar to the algorithms presented in [P1], [P2]. It starts by setting $\mathscr{U}$, the collection of *unprocessed* clusters, to equal $\mathscr{R}$. The procedure then operates in stages, corresponding to the iterations of the main "repeat" loop in the code of Fig. 1, which are henceforth referred to as the *main stages*. Each main stage constructs one output cluster $Y \in \mathscr{DT}$, by merging some clusters of $\mathscr{U}$. The stage begins by arbitrarily picking an input cluster $\mathscr{S}$ in $\mathscr{U}$ and designating it as the *kernel* of the output cluster to be constructed next. The cluster is then repeatedly merged with intersecting input clusters from $\mathscr{U}$. This is done in a layered fashion, adding one layer at a time. This growth process is performed by iterations of the internal "repeat" loop of the procedure, henceforth referred to as the *internal iterations*. At each such internal iteration, the original cluster is viewed as the *kernel* Y, and the resulting cluster Z consists of all input clusters in $\mathscr{U}$ that intersect Y. For the next iteration, Y is set to the current Z. The merging process performed by the internal loop is carried repeatedly until reaching the appropriate *sparsity condition*, specified as follows. Throughout the process, the procedure keeps the clusters Y and Z also in an "unmerged" form, i.e., as collections $\mathscr{Y}$, $\mathscr{Z}$ containing the original input clusters from $\mathscr{U}$ that were merged into Y and Z. The cluster Z is viewed as satisfying the sparsity condition when $|\mathscr{Z}| \leq |\mathscr{R}|^{1/k}|\mathscr{Y}|$ (meaning that the next internal iteration increases the number of clusters merged into Z by a factor of no more than $|\mathscr{R}|^{1/k}$). Upon reaching this condition, the current output cluster is finalized to be the kernel Y of the resulting cluster Z, and the procedure adds Y to the output collection $\mathscr{DT}$. Also, every input cluster in the collection $\mathscr{Y}$ is added to $\mathscr{DR}$, and every input cluster in the collection $\mathscr{Z}$ is removed from $\mathscr{U}$. Then a new main stage is started, constructing the next output cluster. These main stages proceed until $\mathscr{U}$ is exhausted. The procedure then outputs the sets $\mathscr{DR}$ and $\mathscr{DT}$.

```
U ← R                                          /* set of unprocessed input clusters */
DR ← ∅                                         /* set of subsumed input clusters */
DT ← ∅                                         /* set of subsuming output clusters */
repeat                                         /* main stage */
   Select an arbitrary cluster S ∈ U.          /* initial kernel */
   Z ← {S}
   repeat                                      /* merge layers around kernel */
      Y ← Z
      Y ← ⋃_{S∈Y} S
      Z ← {S | S ∈ U, S ∩ Y ≠ ∅}.
   until |Z| ≤ |R|^{1/k}|Y|                    /* sparsity condition */
   U ← U − Z
   DT ← DT ∪ {Y}
   DR ← DR ∪ Y
until U = ∅
Output (DR, DT).
```

FIG. 1. *Procedure* `Cover` ($\mathscr{R}$).

```
R ← S                                   /* R is the collection of remaining (unsubsumed) clusters */
T ← ∅                                   /* T is the output cover */
repeat
   (DR, DT) ← Cover(R)                  /* invoke procedure Cover */
   T ← T ∪ DT
   R ← R \ DR
until R = ∅
```

FIG. 2. *Algorithm* `Main`.

It is important to note that the output cluster formed by each main stage consists of only the kernel Y, and not the entire cluster Z, which contains an additional "external layer" of $\mathcal{U}$ clusters. The role of this external layer is to act as a "protective barrier" shielding the generated cluster Y, and providing the desired disjointness between the different clusters Y added to $\mathcal{DT}$.

Note that each of the original clusters in $\mathcal{DR}$ is covered by some cluster $Y \in \mathcal{DT}$ constructed during the execution of the procedure. However, some original $\mathcal{R}$ clusters are excluded from consideration without being subsumed by any cluster in $\mathcal{DT}$; these are precisely the clusters merged into some external layer $\mathcal{Z} - \mathcal{Y}$. Therefore there may be clusters left in $\mathcal{R}$ after the main algorithm removes the elements of $\mathcal{DR}$. This is why a single application of procedure `Cover` by the main algorithm is not enough, and many phases are necessary. (In contrast, the algorithms of [P1], [P2] can essentially be thought of as based on a single application of a variant of this process, in which entire Z clusters are taken into the output cover, yielding possibly wide overlaps.)

We now present the algorithm `Main`, whose task is to construct a cover as in Theorem 3.1. The input to the algorithm is a graph $G = (V, E)$, $|V| = n$, a cover $\mathcal{S}$, and an integer $k \geq 1$. The output collection of cover clusters $\mathcal{T}$ is initially empty. The algorithm maintains the set of "remaining" clusters $\mathcal{R}$. These are the clusters not yet subsumed by the constructed cover. Initially, $\mathcal{R} = \mathcal{S}$, and the algorithm terminates once $\mathcal{R} = \varnothing$. The algorithm operates in (at most $k|\mathcal{S}|^{1/k}$) phases. Each phase consists of the activation of the procedure `Cover` ($\mathcal{R}$), which adds a subcollection of output clusters $\mathcal{DT}$ to $\mathcal{T}$ and removes the set of subsumed original clusters $\mathcal{DR}$ from $\mathcal{R}$. That the partial cover $\mathcal{DT}$ constructed by the procedure consists of *disjoint* clusters ensures that each phase contributes at most one to the degree of each vertex in the output cover $\mathcal{T}$. Algorithm `Main` is formally described in Fig. 2.

3.2. Correctness and analysis of the regional cover. The properties of procedure `Cover` are summarized by the following lemma.

LEMMA 3.2. *Given a graph $G = (V, E)$, $|V| = n$, a collection of clusters $\mathcal{R}$, and an integer k, the collections $\mathcal{DT}$ and $\mathcal{DR}$ constructed by procedure* `Cover` ($\mathcal{R}$) *satisfy the following properties*:

(1) $\mathcal{DT}$ *subsumes* $\mathcal{DR}$,

(2) $Y \cap Y' = \varnothing$ *for every* $Y, Y' \in \mathcal{DT}$,

(3) $|\mathcal{DR}| \geq |\mathcal{R}|^{1-1/k}$, *and*

(4) $\mathrm{Rad}(\mathcal{DT}) \leq (2k-1)\,\mathrm{Rad}(\mathcal{R})$.

Proof. First, let us note that since the elements of $\mathcal{U}$ at the beginning of the procedure are clusters (i.e., their induced graphs are connected), the construction process guarantees that every set Y added to $\mathcal{DT}$ is a cluster. Property (1) now holds directly from the construction.

Let us now prove property (2). Suppose, seeking to establish a contradiction, that there is a vertex v such that $v \in Y \cap Y'$. Without loss of generality, suppose that Y was created earlier (i.e., in an earlier main stage) than Y'. Since $v \in Y'$, there must be a cluster S' such that $v \in S'$ and S' was still in $\mathcal{U}$ when the algorithm started the main stage constructing Y'. However, every such cluster S' satisfies $S' \cap Y \neq \emptyset$. Therefore, in the main stage that created Y, the construction step creating the collection $\mathcal{Z}$ from Y at the last internal iteration should have added S' into $\mathcal{Z}$ and eliminated it from $\mathcal{U}$, a contradiction.

Property (3) is derived as follows. It is immediate from the sparsity condition (governing the termination of the internal loop) that the resulting pair $\mathcal{Y}$, $\mathcal{Z}$ satisfies $|\mathcal{Z}| \leq |\mathcal{R}|^{1/k}|\mathcal{Y}|$. Therefore

$$|\mathcal{R}| = \sum_{\mathcal{Z}} |\mathcal{Z}| \leq \sum_{\mathcal{Y}} |\mathcal{R}|^{1/k}|\mathcal{Y}| = |\mathcal{R}|^{1/k}|\mathcal{D}\mathcal{R}|,$$

which proves property (3).

Finally, we analyze the increase in the radius of clusters in the cover. Consider some main stage of procedure `Cover` in Fig. 1, starting with the selection of some cluster $S \in \mathcal{R}$. Let J denote the number of internal iterations executed during this main stage. Denote the initial set $\mathcal{Z}$ by $\mathcal{Z}_0$. Denote the set $\mathcal{Z}$ (respectively, Y, $\mathcal{Y}$) constructed on the ith internal iteration ($1 \leq i \leq J$) by $\mathcal{Z}_i$ (respectively, Y_i, $\mathcal{Y}_i$). Note that for $1 \leq i \leq J$, $\mathcal{Z}_i$ is constructed on the basis of $\mathcal{Y}_i$, $\mathcal{Y}_i = \mathcal{Z}_{i-1}$ and $Y_i = \cup_{S \in \mathcal{Y}_i} S$. We proceed along the following chain of claims.

CLAIM 3.3. $|\mathcal{Z}_i| \geq |\mathcal{R}|^{i/k}$ *for every* $0 \leq i \leq J-1$, *and strict inequality holds for* $i \geq 1$.

Proof. The proof follows by induction on i. The claim is immediate for $i = 0$. Assuming the claim for $i - 1 \geq 0$, it remains to prove that

$$|\mathcal{Z}_i| > |\mathcal{R}|^{1/k}|\mathcal{Z}_{i-1}|,$$

which follows directly from the fact that the sparsity condition was not met, since the internal loop was not terminated. □

COROLLARY 3.4. *It holds that* $J \leq k$.

CLAIM 3.5. *For every* $1 \leq i \leq J$, $\mathrm{Rad}(Y_i) \leq (2i-1)\,\mathrm{Rad}(\mathcal{R})$.

Proof. The proof follows by straightforward induction on i. The base case is immediate, since $Y_1 = S \in \mathcal{R}$. The inductive step follows from the fact that, for $2 \leq i \leq J$,

$$\mathrm{Rad}(Y_i) \leq \mathrm{Rad}(Y_{i-1}) + 2\,\mathrm{Rad}(\mathcal{R}),$$

since $\mathcal{Z}_{i-1}$ is created from Y_{i-1} by adding into it all $\mathcal{U}$ clusters intersecting it, and Y_i is simply a merge of all the clusters in $\mathcal{Z}_{i-1}$. □

It follows from Corollary 3.4 and Claim 3.5 that

$$\mathrm{Rad}(Y_J) \leq (2k-1)\,\mathrm{Rad}(\mathcal{R}),$$

which completes the proof of the last property of the lemma. □

We are now ready to prove Theorem 3.1.

Proof of Theorem 3.1. We must prove that, given a graph $G = (V, E)$, $|V| = n$, a cover $\mathcal{S}$, and an integer $k \geq 1$, the cover $\mathcal{T}$ constructed by algorithm `Main` satisfies the three properties of the theorem.

Let $\mathcal{R}_i$ denote the set $\mathcal{R}$ at the beginning of phase i and let $r_i = |\mathcal{R}_i|$. Let $\mathcal{D}\mathcal{T}_i$ denote the collection $\mathcal{D}\mathcal{T}$ added to $\mathcal{T}$ at the end of phase i and let $\mathcal{D}\mathcal{R}_i$ be the set $\mathcal{D}\mathcal{R}$ removed from $\mathcal{R}$ at the end of phase i.

Property (1) follows from the fact that $\mathscr{T} = \cup_i \mathscr{DT}_i$, $\mathscr{S} = \cup_i \mathscr{DR}_i$, and, by property (1) of Lemma 3.2, $\mathscr{DT}_i$ subsumes $\mathscr{DR}_i$ for every i. Property (2) follows directly from property (4) of Lemma 3.2. It remains to prove property (3). This property relies on the fact that, by property (2) of Lemma 3.2, each vertex v participates in at most one cluster in each collection $\mathscr{DT}_i$. Therefore it remains to bound the number of phases performed by the algorithm. This bound relies on the following observations. By property (3) of Lemma 3.2, in every phase i, at least $|\mathscr{DR}_i| \geqq |\mathscr{R}_i|^{1-1/k}$ clusters of $\mathscr{R}_i$ are removed from the set $\mathscr{R}_i$, i.e., $r_{i+1} \leqq r_i - r_i^{1-1/k}$.

CLAIM 3.6. *Consider the recurrence relation $x_{i+1} = x_i - x_i^\alpha$, for $0 < \alpha < 1$. Let $f(n)$ denote the least index i such that $x_i \leqq 1$ given $x_0 = n$. Then*

$$f(n) < \frac{n^{1-\alpha}}{(1-\alpha)\ln 2}.$$

Proof. It follows from the definition of $f(n)$ that

$$f(n) \leqq \frac{n/2}{(n/2)^\alpha} + f(n/2),$$

since x_i decreases by at least $(n/2)^\alpha$ in each of the first k steps $1 \leqq i \leqq k$ until $x_k \leqq n/2$. From this, we get (recalling that $\int a^x dx = a^x/\ln a$ and $\alpha - 1 < 0$) that

$$f(n) \leqq n^{1-\alpha} \sum_{j=1}^{\log n} (2^{\alpha-1})^j < n^{1-\alpha} \int_{x=0}^{\infty} (2^{\alpha-1})^x dx = \frac{n^{1-\alpha}}{(1-\alpha)\ln 2}.$$

Consequently, since $r_0 = |\mathscr{S}|$, $\mathscr{S}$ is exhausted after no more than $(k/\ln 2)|\mathscr{S}|^{1/k}$ phases of algorithm `Main`, and hence $\Delta(\mathscr{T}) \leqq 2k|\mathscr{S}|^{1/k}$. This completes the proof of the theorem. □

4. The routing scheme. In this section, we describe our routing strategy and the structures that it uses in the network. Our approach is based on constructing a *hierarchy* of covers in the network and using this hierarchy for routing. In each level, the graph is covered by clusters (namely, connected subgraphs), each managed by a center vertex. Each cluster has its own internal routing mechanism (described in the following section), enabling routing to and from the center. Messages are always transferred to their destinations using the internal routing mechanism of some cluster, along a route going through the cluster center. It is clear that this approach reduces the memory requirements of the routing schemes, since we must define routing paths only for cluster centers. However, it increases the communication cost, since messages need not be moving along shortest paths. Through an appropriate choice of the cluster cover, we guarantee that both overheads are low.

4.1. Tree routing. In this section, we discuss the basic *tree routing* component used inside clusters. This component, used earlier in [P2], is similar to (yet simpler than) the one used in [ABLP]. It is based on a shortest path tree T rooted at a center r of the cluster S and spanning the cluster. Routing messages to the root is a straightforward task. We need a mechanism enabling us to route a message from the root, where the destination is not necessarily in the tree (in which case, the message is to be returned to the root with an appropriate notification).

This subproblem was treated in previous papers using a simple scheme called the *interval routing scheme*, or ITR [SK], [PU]. This scheme is based on assigning the vertices $u \in T$ a depth-first numbering DFS(u) and storing at a vertex u its own depth-first search (DFS) number and the numbers DFS(u') of each of its children u' in the tree. Then routing a message from the root r to a vertex v (assuming that r knows the

value DFS (v)) involves propagating the message from each intermediate vertex u with children $u_1, \cdots, u_q$ (ordered by increasing DFS numbering) to the child u_i such that DFS (u_i) $\leqq$ DFS (v) $<$ DFS (u_{i+1}) (setting DFS (u_{q+1}) to ∞). In total, the communication cost of sending a message inside the cluster S is, at most, 2 Rad (S).

Using the ITR scheme for our purposes poses some new technical problems. First, the scheme requires using the DFS numbers as routing labels, which interferes with the name-independence requirement. This problem is solved in [ABLP] by introducing a distributed data structure called the "tree dictionary." This data structure stores the data items (name (v), DFS (v)) for every vertex $v \in S$. (Recall that name (v) is the original name of the vertex v, taken from the ordered universe U.) These data items are ordered by original names as keys. The structure therefore enables the root to retrieve the DFS label of a vertex v, DFS (v), using its original name name (v) as a key, and thus eliminates the need to know the DFS labels in advance by the origins.

The data structure is organized by simply storing the data items one at each node of the cluster, in increasing *name* order, along the DFS tour defined on the tree T. That is, the root r (with DFS (r) = 1) stores the first (lowest-keyed) item (corresponding to the node w with the lowest name (w)), the second vertex u (with DFS (u) = 2) stores the next item, and so forth. Let $K(u)$ denote the key (or *name*) of the item stored at u. In addition to its own data item, a vertex u stores the key $K(u')$ for each of its children u' in the tree.

Let us now describe the process by which a processor v searches for a label DFS (w), given an original name N_w = name (w). It first sends a query to the root asking for the item. There are three cases to be distinguished. If $N_w < K(r)$, then the search immediately ends in "failure." If $N_w = K(r)$, then the searched item is stored locally at the root, and the search is completed successfully. Finally, suppose that $N_w > K(r)$. Let the children of r (ordered by increasing DFS numbering) be $u_1, \cdots, u_q$. Advance the query to the child u_i such that $K(u_i) \leqq N_w < K(u_{i+1})$ (setting $K(u_{q+1}) = \infty$). The search now proceeds by propagating the query downward in the tree, applying the same rule at each intermediate vertex visited along the way (except that the first of the three cases cannot occur at any other vertex but the root). In the case where the search arrives at a "dead end" (i.e., a leaf u that does not store the searched item), the returned answer is again "failure." The reply is finally returned to the root, which then forwards it to v.

Clearly, the cost of the search is, at most, 2 Rad (S). Also, maintaining the tree dictionary involves the following memory requirements. Each vertex of S must store its own item, as well as the first key of each of its children in the tree. This leads us to the second problem introduced by using either the ITR scheme or the tree dictionary, namely, that the memory stored at a vertex depends linearly on its degree and thus may be as high as $\Omega(n)$ in the worst case. This interferes with the balanced-memory requirement. This second problem is handled in [ABLP] by proving that we can embed into any tree a tree of "small" degrees, without paying too high a price in memory and without increasing the depth of the tree "too much," where the degrees and depth are controlled by some parameter k.

LEMMA 4.1 (see [ABLP]). *For any rooted tree T, there exists an* embedded tree *T' on the same set of nodes and with the same root, but with a different set of edges, so that*

(1) *the maximal degree of T' is $2n^{1/k}$;*

(2) *an edge of T' is a path of length at most two in T;*

(3) $\text{depth}_{T'}(v) \leqq (2k-1)\,\text{depth}_T(v)$ *for every node v.*

We construct for the given tree T an embedded tree T' as in Lemma 4.1 and organize the dictionary on the embedded tree T', which is of maximal degree $O(n^{1/k})$. By property (1) of Lemma 4.1, each vertex of S must store $O(n^{1/k} \log n)$ bits. The length of the

resulting search path is stretched by a factor of $2k-1$ by property (3) of Lemma 4.1. Hence we now pay up to $4k\cdot \mathrm{Rad}(S)$ messages for the search of DFS(u) in the dictionary, for any $u\in V$, regardless of whether $u\in S$, plus up to $4k\cdot \mathrm{Rad}(S)$ messages for the routing itself.

Summarizing, we get the following lemma.

LEMMA 4.2. *For every graph G, cluster S in G, and integer $k\geq 1$, the tree-routing component described above guarantees the delivery of messages between vertices in the cluster S (or "failure" messages in the case where the destination is not in the cluster) with communication $8k\cdot \mathrm{Rad}(S)$ and requires using $O(n^{1/k}\log n)$ bits per vertex.*

Let us remark that, in fact, the scheme actually used in [ABLP] is aimed at solving a somewhat harder task than ours. It therefore uses a special type of *stratified* tree structure, which increases the resulting complexity. This feature is not needed here, and, consequently, we use only the basic "tree dictionary." In the scheme as presented in [ABLP], both the memory requirements and the communication complexity are higher by a factor of k due to this stratified structure.

4.2. Regional routing schemes. Each level in our hierarchy constitutes a *regional (C, m)-routing scheme*, which is a scheme with the following properties. For every two processors u, v, if $\mathrm{dist}(u, v)\leq m$, then the scheme succeeds in delivering messages from u to v. Otherwise, the routing might end in failure, in which case the message is returned to u. In either case, the communication cost of the entire process is at most C.

In this section, we describe how to construct a regional $(O(k^2m), m)$-routing scheme using $O(k\cdot n^{1/k}\cdot \log n)$ memory bits per vertex, for any integers k, $m\geq 1$. The main stage of the construction involves an application of Theorem 3.1. We start by setting $\mathscr{S}=\mathscr{N}_m(V)$ and constructing a cover $\mathscr{T}$ as in the theorem, with parameter k. Next, we provide internal routing services in each cluster T by selecting a center $\ell(T)$ and constructing a tree-routing component for T rooted at this center. By property (1) of Theorem 3.1, the cover $\mathscr{T}$ subsumes $\mathscr{N}_m(V)$; that is, for every vertex $v\in V$, there is a cluster $T\in\mathscr{T}$ such that $N_m(v)\subseteq T$. Consequently, we associate with every vertex $v\in V$ a *home cluster* $\mathrm{home}(v)\in\mathscr{T}$, which is the cluster containing $N_m(v)$. (In the case where there are several appropriate clusters, select one arbitrarily.) A processor v routes a message by sending it to its home cluster center $\ell(\mathrm{home}(v))$. The center uses the tree routing mechanism to forward the message to its destination. If that destination is not found in the cluster, the message is returned to the root and, from there, to the originator.

The correctness and complexity of the constructed regional scheme are dealt with in the following lemma, relying on the properties of the tree-routing mechanism and the constructed cover.

LEMMA 4.3. *For every graph G and integers $m, k\geq 1$, the mechanism described above is a regional $(O(k^2m), m)$-routing scheme, and it can be implemented using $O(k\cdot n^{1/k}\cdot \log n)$ memory bits per vertex.*

Proof. Let us first determine the stretch guarantees of the scheme. Suppose that $\mathrm{dist}(u, v)\leq m$ for some processors u, v. By definition, $v\in N_m(u)$. Let T be the home cluster of u. Then $N_m(u)\subseteq T$; so $v\in T$. Hence, by Lemma 4.2, the tree routing on T will succeed in passing the message from u to v. Furthermore, regardless of whether the message is delivered, the routing proceeds along a path of length at most $8k\cdot \mathrm{Rad}(T)$. By property (2) of Theorem 3.1,

$$8k\cdot \mathrm{Rad}(\ell(T), T)\leq 8k(2k-1)m<16k^2m.$$

Implementing the routing scheme involves the following space requirements. By Lemma 4.2, each vertex v must store up to $O(n^{1/k}\log n)$ bits for every $\mathscr{T}$ cluster to

whom it belongs (i.e., for $\deg_{\mathcal{T}}(v)$ clusters). By property (3) of Theorem 3.1, this degree is $O(k \cdot n^{1/k})$, since $|\mathcal{N}_m(V)| = n$. This gives a total of $O(k \cdot n^{2/k} \cdot \log n)$ bits per vertex. Finally, note that substituting $2k$ for k in the construction modifies the memory bound into $O(k \cdot n^{1/k} \cdot \log n)$, while multiplying the stretch bound by 4. □

4.3. The hierarchical routing scheme. Finally, we present our family of *hierarchical routing schemes*. For every fixed integer $k \geq 1$, construct the hierarchical scheme $\mathcal{H}_k$ as follows. Let $\delta = \lceil \log \text{Diam}(G) \rceil$. For $1 \leq i \leq \delta$, construct a regional $(O(k^2 m_i), m_i)$-routing scheme R_i, where $m_i = 2^i$, as in the last section, using $O(k \cdot n^{1/k} \cdot \log n)$ memory bits per vertex. Each processor v participates in all δ regional routing schemes R_i. In particular, v has a home cluster $\text{home}_i(v)$ in each R_i and it stores all the information it is required to store for each of these schemes.

The routing procedure operates as follows. Suppose a vertex u wishes to send a message to a vertex v. Then u first tries using the lowest-level regional scheme R_1; i.e., it forwards the message to its first home cluster center $\ell(\text{home}_1(v))$. If this trial fails, u retries sending its message, this time using regional scheme R_2, and so on, until it finally succeeds.

LEMMA 4.4. *The hierarchical routing scheme $\mathcal{H}_k$ has* $\text{Stretch}(\mathcal{H}_k) = O(k^2)$.

Proof. Suppose that a processor u needs to send a message to some other processor v. Let $d = \text{dist}(u, v)$ and $j = \lceil \log d \rceil$ (i.e., $2^{j-1} < d \leq 2^j$). The sender u successively tries forwarding the message using the regional schemes R_1, R_2, and so on, until the message finally succeeds in arriving v. By Lemma 4.3, the regional scheme R_j will necessarily succeed, if no previous level did. (Note that the highest-level scheme R_δ has $m_\delta = 2^\delta \geq \text{Diam}(G) \geq d$ and therefore will always succeed.)

Denote the total length of the combined path traversed by the message by ρ. By Lemma 4.3,

$$\rho \leq \sum_{i=1}^{j} O(k^2 2^i) \leq O(k^2 2^{j+1}) \leq O(k^2) \cdot \text{dist}(u, v). \qquad \square$$

THEOREM 4.5. *For every graph G and every fixed integer $k \geq 1$, it is possible to construct (in polynomial time) a hierarchical routing scheme $\mathcal{H}_k$ with* $\text{Stretch}(\mathcal{H}_k) = O(k^2)$, *using $O(k \cdot n^{1/k} \cdot \log n \log \text{Diam}(G))$ memory bits per vertex.*

Proof. Construct the $\delta = \lceil \log \text{Diam}(G) \rceil$ regional schemes R_i, as in the previous section. The memory requirements of the hierarchical scheme are thus composed of δ terms, each bounded by $O(k \cdot n^{1/k} \cdot \log n)$ by Lemma 4.3. The total memory requirements are thus bounded by $O(k \cdot n^{1/k} \cdot \log n \log \text{Diam}(G))$ bits per vertex. □

5. Discussion. Several problems remain open for further research. First, the trade-off obtained here is still not optimal, and it is conceivably possible to reduce the stretch factor of the routing schemes from $O(k^2)$ to $O(k)$. It is also desirable to eliminate the dependency of the memory requirements of the scheme on the edge weights.

The issue of efficient preprocessing, revolving around the problem of faster and communication-efficient distributed algorithms for constructing a coarsening cover, was studied further in [AP4], [LS]. In particular, the distributed asynchronous clustering algorithm of [AP4] has a global communication cost of $O(E \cdot \log^4 n)$. The randomized synchronous clustering algorithm of [LS] applies to the 1-neighborhood cover $\mathcal{N}_1(V)$ and takes time $O(\log^2 n)$.

Acknowledgments. The authors thank Steve Ponzio for pointing out an error in a previous version of the paper. Thanks are also due to two anonymous referees, whose comments helped to improve the presentation of this paper.

REFERENCES

[AR] Y. AFEK AND M. RICKLIN, *Sparser: a paradigm for running distributed algorithms*, 1990, unpublished manuscript.

[ABLP] B. AWERBUCH, A. BAR-NOY, N. LINIAL, AND D. PELEG, *Compact Distributed Data Structures for Adaptive Routing*, in Proc. 21st ACM Sympos. on Theory of Computing, Seattle, WA, May 1989, pp. 479–489.

[AKP] B. AWERBUCH, S. KUTTEN, AND D. PELEG, *On Buffer-Economical Store-and-Forward Deadlock Prevention*, IEEE Trans. on Communications, 1991, to appear.

[AP1] B. AWERBUCH AND D. PELEG, *Online Tracking of Mobile Users*, Tech. Report MIT/LCS/TM-410, Massachusetts Institute of Technology, Cambridge, MA, August 1989.

[AP2] ———, *Network Synchronization with Polylogarithmic Overhead*, in Proc. 31st Sympos. on Foundations of Computer Science, St. Louis, MO, October 1990, pp. 514–522.

[AP3] ———, *Sparse Partitions*, in Proc. 31st IEEE Sympos. on Foundations of Computer Science, St. Louis, MO, October 1990, pp. 503–513.

[AP4] ———, *Efficient Distributed Construction of Sparse Covers*, Tech. Report CS90-17, The Weizmann Institute, Rehovot, Israel, July 1990.

[AP5] ———, *Locality-Sensitive Resource Allocation*, Tech. Report CS90-27, The Weizmann Institute, Rehovot, Israel, November 1990.

[BJ] A. E. BARATZ AND J. M. JAFFE, *Establishing Virtual Circuits in Large Computer Networks*, Comput. Networks, 12 (1986), pp. 27–37.

[FJ1] G. N. FREDERICKSON AND R. JANARDAN, *Designing Networks with Compact Routing Tables*, Algorithmica, 3 (1988), pp. 171–190.

[FJ2] ———, *Efficient Message Routing in Planar Networks*, SIAM J. Comput., 18 (1989), pp. 843–857.

[FJ3] ———, *Space-Efficient Message Routing in c-decomposable Networks*, SIAM J. Comput., 19 (1990), pp. 164–181.

[KK1] L. KLEINROCK AND F. KAMOUN, *Hierarchical Routing for Large Networks; Performance Evaluation and Optimization*, Comput. Networks, 1 (1977), pp. 155–174.

[KK2] ———, *Optimal Clustering Structures for Hierarchical Topological Design of Large Computer Networks*, Networks, 10 (1980), pp. 221–248.

[LS] N. LINIAL AND M. SAKS, *Decomposing Graphs Into Regions of Small Diameter*, in Proc. 2nd ACM Sympos. on Discrete Algorithms, San Francisco, 1991, pp. 320–330.

[P1] D. PELEG, *Sparse Graph Partitions*, Report CS89-01, Dept. of Applied Math., The Weizmann Institute, Rehovot, Israel, February 1989.

[P2] ———, *Distance-Dependent Distributed Directories*, Information and Computation, to appear. See also Report CS89-10, Dept. of Applied Math., The Weizmann Institute, Rehovot, Israel, May 1989.

[PU] D. PELEG AND E. UPFAL, *A Tradeoff between Size and Efficiency for Routing Tables*, J. Assoc. Comput. Mach., 36 (1989), pp. 510–530.

[Pr] R. PERLMAN, *Hierarchical Networks and the Subnetwork Partition Problem*, in Proc. 5th Conf. on System Sciences, Hawaii, 1982.

[SK] N. SANTORO AND R. KHATIB, *Labelling and Implicit Routing in Networks*, Comput. J., 28 (1985), pp. 5–8.

[S] C. A. SUNSHINE, *Addressing Problems in Multi-Network Systems*, in Proc. IEEE INFOCOM, Las Vegas, NV, 1982.

[vLT1] J. VAN LEEUWEN AND R. B. TAN, *Routing with Compact Routing Tables*, in The Book of L, G. Rozenberg and A. Salomaa, eds., Springer-Verlag, New York, 1986, pp. 259–273.

[vLT2] ———, *Interval Routing*, Comput. J., 30 (1987), pp. 298–307.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 163–177, May 1992
© 1992 Society for Industrial and Applied Mathematics
002

CHVÁTAL CUTS AND ODD CYCLE INEQUALITIES IN QUADRATIC 0 - 1 OPTIMIZATION*

E. BOROS†¶[1], Y. CRAMA‡¶, AND P. L. HAMMER§¶

Abstract. In this paper a new lower bound for unconstrained quadratic 0 - 1 minimization is investigated. It is shown that this bound can be computed by solving a linear programming problem of polynomial size in the number of variables; and it is shown that the polyhedron $\mathbf{S}^{[3]}$, defined by the constraints of this LP formulation is precisely the first Chvátal closure of the polyhedron associated with standard linearization procedures. By rewriting the quadratic minimization problem as a balancing problem in a weighted signed graph, it can be seen that the polyhedron defined by the odd cycle inequalities is equivalent, in a certain sense, with $\mathbf{S}^{[3]}$. As a corollary, a compact linear programming formulation is presented for the maximum cut problem for the case of weakly bipartite graphs.

Key words. unconstrained quadratic 0 - 1 programming, pseudo-Boolean functions, weighted 2-satisfiability, maximum cut problem, Chvátal cut, Chvátal closure

AMS(MOS) subject classifications. 90C09, 90C20, 90C27

1. Introduction. The *quadratic* 0 - 1 *minimization problem* consists in determining the minimum over $\{0, 1\}^n$ of quadratic polynomials of the form

$$f(x) = q_0 + \sum_{i=1}^{n} q_i x_i + \sum_{1 \leqq i < j \leqq n} q_{ij} x_i x_j. \tag{1.1}$$

Since this problem is NP-hard, many approaches have been proposed to obtain good lower bounds on its optimal value. In [20] several such approaches were proved to yield exactly the same lower bound, called the roof-dual value of $f(x)$, and denoted here by C_2.

The simplest way of defining the roof-dual value is first to rewrite the problem $\min \{f(x) \mid x \in \{0, 1\}^n\}$ by introducing new variables y_{ij} and constraining y_{ij} to take the value $x_i x_j$ for every $x \in \{0, 1\}^n (1 \leqq i < j \leqq n)$. We thus obtain the following equivalent linear 0 - 1 program (see § 2):

$$\min L_f(x, y) = q_0 + \sum_{i=1}^{n} q_i x_i + \sum_{1 \leqq i < j \leqq n} q_{ij} y_{ij}$$

subject to

$$\left.\begin{aligned} y_{ij} &\geqq 0 \\ x_i - y_{ij} &\geqq 0 \\ x_j - y_{ij} &\geqq 0 \\ 1 - x_i - x_j + y_{ij} &\geqq 0 \end{aligned}\right\} \quad 1 \leqq i < j \leqq n, \tag{1.2}$$

$$x_i \in \{0, 1\}, \qquad 1 \leqq i \leqq n. \tag{1.3}$$

* Received by the editors March 14, 1990; accepted for publication (in revised form) April 2, 1991.

† DIMACS and RUTCOR, Rutgers University, New Brunswick, New Jersey 08903.

‡ Department of Quantitative Economics, University of Limburg, Maastricht, the Netherlands.

§ RUTCOR, Rutgers University, New Brunswick, New Jersey 08903.

¶ This work was partially supported by National Science Foundation grant DMS 89-06870 and Air Force Office of Scientific Research grants 89-0512 and 90-008.

[1] The work of this author was supported by a Rutgers University Research Council Award grant 2-02229 and by National Science Foundation grant STC 88-09648.

Now let $\mathbf{S}^{[2]}$ be the polyhedron described by the constraints (1.2). Then $C_2 = \min\{L_f(x, y) | (x, y) \in \mathbf{S}^{[2]}\}$. Clearly, C_2 is a lower bound on $\min\{f(x) | x \in \{0, 1\}^n\}$.

Some of the results of [20] were generalized in [9], where a hierarchy $C_2, C_3, \cdots, C_n$ of lower bounds was introduced, with the property that $C_2 \leqq C_3 \leqq \cdots \leqq C_n = \min\{f(x) | x \in \{0, 1\}^n\}$. The present paper is devoted to a further study of the bound C_3.

We recall in § 2 one of the possible definitions of the bound C_3 as being the optimal value of a linear programming problem $C_3 = \min\{L_f(x, y) | (x, y) \in \mathbf{S}^{[3]}\}$. We give an explicit description of the polyhedron $\mathbf{S}^{[3]}$ by a system of linear inequalities involving all inequalities (1.2), plus $O(n^3)$ additional inequalities (Theorem 2.3).

In § 3, we prove that $\mathbf{S}^{[3]}$ is nothing but the first Chvátal closure of $\mathbf{S}^{[2]}$ (Theorem 3.3).

In § 4, we first recall a transformation of the quadratic 0 – 1 minimization problem into a weighted signed graph balancing problem. We consider a set-covering formulation (CC) (see formula (4.2) of this paper) of the latter problem, where the objective is to cover all negative cycles of the signed graph by a minimum weight set of edges. Note that the number of constraints of (CC) may be exponentially large in the size of the graph. We establish in Theorem 4.2 that C_3 is exactly the optimal value of the linear relaxation of (CC).

We state in § 5 an interesting corollary of Theorem 4.2: the relaxation of the max-cut problem defined by all trivial and odd cycle inequalities [8], [17] can be expressed as a polynomial-size linear programming problem (Theorem 5.1). In particular, the max-cut problem for weakly bipartite graphs [5], [17] admits such a compact linear programming formulation (see also [6]).

The proofs of our main results rely heavily on algebraic manipulations of polynomial expressions like (1.1). The remainder of this section contains some definitions and basic observations about such expressions, which will be used throughout the paper.

A *pseudo-Boolean function* is a real-valued function defined on $\{0, 1\}^n$ (for some integer n). Every pseudo-Boolean function on $\{0, 1\}^n$ has a unique polynomial expression of the form

$$f(x_1, \cdots, x_n) = \sum_{T \in \Lambda} q_T \prod_{x \in T} x, \tag{1.4}$$

where $V = \{x_1, x_2, \cdots, x_n\}$ is a set of 0 – 1 variables, Λ is a collection of subsets of V, and $q_T \neq 0$ for all $T \in \Lambda$, and where $\prod_{x \in \varnothing} x = 1$.

Let $\bar{V} = \{\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_n\}$ denote the set of complemented variables, where the *complement* of a variable x_i is defined by $\bar{x}_i = 1 - x_i$. The elements of $L = V \cup \bar{V}$ are called *literals*.

A *posiform* is an expression of the form

$$\phi(x_1, \cdots, x_n, \bar{x}_1, \cdots, \bar{x}_n) = \sum_{T \in \Omega} a_T \prod_{u \in T} u, \tag{1.5}$$

where Ω is a collection of subsets of L, and $a_T > 0$ for all $T \in \Omega$. (For the sake of simplicity, we may assume that the elements of Ω do not contain complemented pairs of variables.) The *degree* of the posiform (1.5) is the maximum size of a set $T \in \Omega$.

Every posiform ϕ defines a unique pseudo-Boolean function f through the natural correspondence

$$f(x_1, \cdots, x_n) = \phi(x_1, \cdots, x_n, \bar{x}_1, \cdots, \bar{x}_n) \quad \text{for all } (x_1, \cdots, x_n) \in \{0, 1\}^n.$$

In this case, ϕ is said to be a posiform of f. Observe that a pseudo-Boolean function may have many distinct posiforms.

The set of quadratic pseudo-Boolean functions forms a vector space, of dimension $1 + n + \binom{n}{2}$, over the reals. Let $\mathscr{P}_2$ be the cone generated by the set of nonnegative, quadratic pseudo-Boolean functions having a posiform of degree two (*quadratic posiforms*), let $\mathscr{P}_3$ be the cone generated by the nonnegative, quadratic pseudo-Boolean functions having a posiform of degree three (*cubic posiforms*), and let $\mathscr{P}_n$ denote the set of all nonnegative, quadratic pseudo-Boolean functions in n variables (a slightly different definition of $\mathscr{P}_2$, $\mathscr{P}_3$, $\mathscr{P}_n$ was used in [9]; these definitions, however, can be proved to be equivalent). Clearly, $\mathscr{P}_2 \subseteq \mathscr{P}_3 \subseteq \mathscr{P}_n$, and the inclusions are strict.

Example 1.1. Let $f(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2x_3 - x_1 - x_2 - x_3 + 1$. Then $f \in \mathscr{P}_3$, since $f = x_1x_2x_3 + \bar{x}_1\bar{x}_2\bar{x}_3$. f is not in $\mathscr{P}_2$, however.

It was observed in [9] that the cones $\mathscr{P}_2$, $\mathscr{P}_3$, $\mathscr{P}_n$ are finitely generated. We denote by $\mathscr{B}(\mathscr{P}_2)$, $\mathscr{B}(\mathscr{P}_3)$, and $\mathscr{B}(\mathscr{P}_n)$ the sets of extremal elements of these three cones.

2. A linearization technique. We present in this section a precise definition and a characterization of the bound C_3. Our approach is based on a classical linearization technique for nonlinear 0 - 1 optimization problems (see, e.g., [1], [3], [4], [13], [15], [16], [20], [22], [24], [25], [27] or the survey [21]). We recall here the formulation given in [9].

First, observe that there is a natural one-to-one correspondence between the quadratic pseudo-Boolean functions over $\{0, 1\}^n$

$$f(x) \overset{\text{def}}{=} q_0 + \sum_{i=1}^{n} q_i x_i + \sum_{1 \leqq i < j \leqq n} q_{ij} x_i x_j \tag{2.1}$$

and the linear functions over $\mathbf{R}^{n+\binom{n}{2}}$, given by

$$f(x) \leftrightarrow \mathbf{L}_f(x, y) \overset{\text{def}}{=} q_0 + \sum_{i=1}^{n} q_i x_i + \sum_{1 \leqq i < j \leqq n} q_{ij} y_{ij}, \tag{2.2}$$

where (x, y) denotes the vector $(x_1, \cdots, x_n, y_{12}, \cdots, y_{n-1,n}) \in \mathbf{R}^{n+\binom{n}{2}}$.

Let $\mathbf{Q}$ denote the Boolean quadric polytope, i.e., the convex hull of the points $(x, y) \in \mathbf{R}^{n+\binom{n}{2}}$ satisfying the conditions $y_{ij} = x_i x_j$, for $1 \leqq i < j \leqq n$ (see [25]) as follows:

$$\mathbf{Q} \overset{\text{def}}{=} \operatorname{conv}\left\{(x,y) \,\middle|\, \begin{array}{l} y_{ij} = x_i x_j \; 1 \leqq i < j \leqq n; \\ x_i \in \{0,1\}, i = 1, \cdots, n \end{array}\right\}. \tag{2.3}$$

The minimization of $f(x)$ over $\{0, 1\}^n$ can be reformulated now as a linear programming problem

$$\min \mathbf{L}_f(x, y) \quad \text{s.t.}\,(x, y) \in \mathbf{Q}. \tag{2.4}$$

We say that a linear function

$$l(x, y) = l_0 + \sum_{i=1}^{n} l_i x_i + \sum_{1 \leqq i < j \leqq n} l_{ij} y_{ij}$$

induces a valid inequality for $\mathbf{Q}$ if

$$\forall (x, y) \in \mathbf{Q}: l(x, y) \geqq 0.$$

The following observation is immediate, using the bijection $f \leftrightarrow \mathbf{L}_f$ (see [9] and recall the definitions of $\mathscr{P}_n$ and $\mathscr{B}(\mathscr{P}_n)$ in § 1).

Remark 2.1. A quadratic function f is nonnegative, i.e., $f \in \mathscr{P}_n$, if and only if $\mathbf{L}_f$ induces a valid inequality for $\mathbf{Q}$. Furthermore, $f \in \mathscr{B}(\mathscr{P}_n)$ if and only if $\mathbf{L}_f$ induces a facet for $\mathbf{Q}$.

Example 2.1. The functions x_1x_2, $x_1\bar{x}_2 = x_1 - x_1x_2$, $\bar{x}_1x_2 = x_2 - x_1x_2$, and $\bar{x}_1\bar{x}_2 = 1 - x_1 - x_2 + x_1x_2$ are quadratic and nonnegative. It follows that the inequalities $y_{12} \geqq 0$, $x_1 - y_{12} \geqq 0$, $x_2 - y_{12} \geqq 0$, and $1 - x_1 - x_2 + y_{12} \geqq 0$ are valid for $\mathbf{Q}$.

Now there is a natural way to introduce a relaxation of (2.4) by defining the polyhedron $\mathbf{S}^{[k]}$ (for $k = 2, 3, n$) as the set of vectors (x, y) satisfying the inequalities $\mathbf{L}_g(x, y) \geqq 0$ for all $g \in \mathscr{B}(\mathscr{P}_k)$ and considering the minimization of $\mathbf{L}_f(x, y)$ over $\mathbf{S}^{[k]}$. For $k = 2, 3, n$, we define the lower bound C_k by

$$C_k \overset{\text{def}}{=} \min \{\mathbf{L}_f(x,y) \mid (x,y) \in \mathbf{S}^{[k]}\}.$$

Note that $\mathbf{Q} = \mathbf{S}^{[n]} \subseteq \mathbf{S}^{[3]} \subseteq \mathbf{S}^{[2]}$, and hence $C_2 \leqq C_3 \leqq C_n = \min \{f(x) \mid x \in \{0, 1\}^n\}$. We leave it as an easy exercise to verify that $\mathscr{B}(\mathscr{P}_2) = \{uv \mid u, v \in L\}$ (where L is the set of literals), and hence that $\mathbf{S}^{[2]}$ is described by inequalities (1.2). As discussed in the Introduction, C_2 is the roof-dual value of f (see [20]).

Our next goal in this section is to characterize more explicitly the polyhedron $\mathbf{S}^{[3]}$, and hence also the bound C_3. For this, we first need to describe the basis $\mathscr{B}(\mathscr{P}_3)$.

Consider the following sets of posiforms:

$$\mathscr{B}_2 = \{uv \mid u, v \in L\} \quad \text{and} \quad \mathscr{B}_3 = \{uvw + \bar{u}\bar{v}\bar{w} \mid u, v, w \in L\}. \tag{2.5}$$

As discussed above, $\mathscr{B}(\mathscr{P}_2) = \mathscr{B}_2$. We show now that $\mathscr{B}(\mathscr{P}_3) = \mathscr{B}_2 \cup \mathscr{B}_3$.

THEOREM 2.2. *$\mathscr{B}_2 \cup \mathscr{B}_3$ is a basis for $\mathscr{P}_3$.*

Proof. Since $uvw + \bar{u}\bar{v}\bar{w} = uv + uw + vw - u - v - w + 1$, $\mathscr{B}_2 \cup \mathscr{B}_3$ is contained in $\mathscr{P}_3$. Consider now a posiform ϕ, and assume that $\phi \in \mathscr{P}_3$. We want to show that ϕ can be written as a nonnegative combination of posiforms from $\mathscr{B}_2 \cup \mathscr{B}_3$. To check this, express ϕ in the form

$$\phi = \sum_{b \in \mathscr{B}_2 \cup \mathscr{B}_3} \lambda_b b + \sum_{T \in \Lambda} \alpha_T \prod_{u \in T} u, \tag{2.6}$$

where $\lambda_b \geqq 0$ ($b \in \mathscr{B}_2 \cup \mathscr{B}_3$), $\alpha_T > 0$, $|T| = 3$ ($T \in \Lambda$), and $|\Lambda|$ is as small as possible (trivially, ϕ can always be expressed in that form). Say $|\Lambda| > 0$, and let $T_1 = \{u, v, w\} \in \Lambda$. Since ϕ is a cubic posiform of a quadratic pseudo-Boolean function, the cubic part of $\alpha_{T_1} uvw$ must be cancelled by some other cubic terms, which can only be of the form $\{\bar{u}, v, w\}$ (or $\{u, \bar{v}, w\}$ or $\{u, v, \bar{w}\}$) or $\{\bar{u}, \bar{v}, \bar{w}\}$. By symmetry, we may assume that Λ contains a set T_2 that is either $\{\bar{u}, v, w\}$ or $\{\bar{u}, \bar{v}, \bar{w}\}$. We may assume without loss of generality that $\alpha_{T_1} \geqq \alpha_{T_2}$. Then, using the identities

$$\alpha_{T_1} uvw + \alpha_{T_2} \bar{u}vw = (\alpha_{T_1} - \alpha_{T_2}) uvw + \alpha_{T_2} vw,$$

and

$$\alpha_{T_1} uvw + \alpha_{T_2} \bar{u}\bar{v}\bar{w} = (\alpha_{T_1} - \alpha_{T_2}) uvw + \alpha_{T_2}(uvw + \bar{u}\bar{v}\bar{w}),$$

the size of Λ can be reduced in (2.6). Since this contradicts our choice of (2.6), the theorem is proved. □

As a corollary, we get the following theorem.

THEOREM 2.3. *For every quadratic pseudo-Boolean function in n variables, the bound C_3 can be computed by solving a linear programming problem involving $n + \binom{n}{2}$ variables and $4\binom{n}{2} + 4\binom{n}{3}$ constraints with $-1, 0, +1$ coefficients.*

Proof. Applying Theorem 2.2, we get that $\mathbf{S}^{[3]}$ is the polyhedron described by the inequalities

$$\left.\begin{aligned} y_{ij} &\geqq 0 \\ -x_i + y_{ij} &\leqq 0 \\ -x_j + y_{ij} &\leqq 0 \\ x_i + x_j - y_{ij} &\leqq 1 \end{aligned}\right\} 1 \leqq i < j \leqq n, \tag{2.7}$$

$$\left.\begin{aligned} x_i + x_j + x_k - y_{ij} - y_{ik} - y_{jk} &\leqq 1 \\ -x_i + y_{ij} + y_{ik} - y_{jk} &\leqq 0 \\ -x_j + y_{ij} - y_{ik} + y_{jk} &\leqq 0 \\ -x_k - y_{ij} + y_{ik} + y_{jk} &\leqq 0 \end{aligned}\right\} 1 \leqq i < j < k \leqq n.$$

3. Linearization and Chvátal closure. Let $\mathbf{P}$ be a polyhedron and let $\mathbf{P}_\mathbf{I} \subseteq \mathbf{P}$ be the convex hull of its integral points. If b is an integer and a is an integral vector such that $a^T(x, y) > b$ is valid over $\mathbf{P}$, then $a^T(x, y) \geqq b + 1$ is a valid inequality for $\mathbf{P}_\mathbf{I}$. This second inequality is called a *Chvátal cut* for $\mathbf{P}$. Let $\mathbf{P}'$ denote the *Chvátal closure of* $\mathbf{P}$, i.e., the polyhedron obtained by introducing all possible Chvátal cuts for $\mathbf{P}$:

$$\mathbf{P}' \overset{\text{def}}{=} \left\{ (u,v) \,\middle|\, \begin{array}{l} a^T(u,v) \geqq b+1 \text{ for all integral } a, b \\ \text{with } a^T(x,y) > b \text{ for all } (x,y) \in \mathbf{P} \end{array} \right\}.$$

Defining $\mathbf{P}^{(0)} = \mathbf{P}$ and $\mathbf{P}^{(k+1)} = \mathbf{P}^{(k)\prime}$ for $k = 0, 1, \cdots$, we get a decreasing sequence of polyhedra $\mathbf{P}^{(0)} \supseteq \mathbf{P}^{(1)} \supseteq \cdots \supseteq \mathbf{P}_\mathbf{I}$. It was shown in [12] that if $\mathbf{P}$ is bounded, there is always a finite index t such that $\mathbf{P}_\mathbf{I} \equiv \mathbf{P}^{(t)}$ (see also [28] for rational polytopes).

The proof that $\mathbf{S}^{[2]\prime} = \mathbf{S}^{[3]}$ will rely on a couple of technical lemmas. Let us call a posiform (1.5) *integral* if all coefficients a_T ($T \in \Omega$) are integral.

LEMMA 3.1. *Let* $S = \{u_1, \cdots, u_l\}$ *be a subset of the literals, and let*

$$g_S = \sum_{i=1}^{l-1} u_i \bar{u}_{i+1} + u_l u_1 + \sum_{i=1}^{l-1} \bar{u}_i u_{i+1} + \bar{u}_l \bar{u}_1, \tag{3.1}$$

$$h_S = \sum_{i=1}^{l-1} u_i \bar{u}_{i+1} + u_l \bar{u}_1 + \sum_{i=1}^{l-1} \bar{u}_i u_{i+1} + \bar{u}_l u_1. \tag{3.2}$$

Then $g_S = 1 + 2q_g$ *and* $h_S = 2q_h$ *for some integral posiforms* $q_g, q_h \in \mathscr{P}_3$.

Proof. The lemma is easily proved by induction on l, using the following observation: for arbitrary literals $\{u, v, w\} \subseteq L$,

$$2(uvw + \bar{u}\bar{v}\bar{w}) + 1 = (uv + \bar{u}\bar{v}) + (uw + \bar{u}\bar{w}) + (vw + \bar{v}\bar{w}). \tag{3.3}$$ □

The posiforms g_S and h_S defined by (3.1)–(3.2) will be called *circulant posiforms*.

LEMMA 3.2. *Let* $g \in \mathscr{P}_2$ *be a quadratic integral posiform such that* $g(x)$ *is odd for every* $x \in \{0, 1\}^n$. *Then* $g = 1 + 2h$ *for some integral posiform* $h \in \mathscr{P}_3$.

Proof. We can assume that all the coefficients of g are 0 or 1 and that g has as few terms as possible. In particular, we can also assume that g is simple; i.e., no u and $\bar{u}$ or no uv and $u\bar{v}$ have positive coefficients at the same time in g (where u, v are arbitrary literals).

• If the term uv is present in g, then $\overline{uv}$ is also present. To see this, write g in the form $g = uv + a\bar{u}\bar{v} + bu + cv + d$, where a is a 0 – 1 coefficient and b, c, d are functions

not depending on u or v. Because $g|_{u=1,v=1} + g|_{u=1,v=0} + g|_{u=0,v=1} + g|_{u=0,v=0}$ must be even, it follows immediately that $a = 1$.

• We can assume that there is no linear part in g. To see this, we note that if u is present in g, then uv cannot be present for any v; indeed $u + uv + \bar{u}\bar{v} = uv + u\bar{v} + uv + \bar{u}\bar{v} = 2uv + \bar{v}$, and hence $g - 2uv$ would also be an "odd quadratic posiform," with fewer terms than g from which the conclusion would easily follow. Then u, and thus $\bar{u}$, are not involved in any other terms of g; hence $g|_{u=0}$ and $g|_{u=1}$ can be of the same parity only if both u and $\bar{u}$ are present, contradicting the simplicity of g.

Thus, in the nontrivial cases, we can assume that g is of the form

$$g = \sum_{(i,j)\in J_0} (x_i x_j + \bar{x}_i \bar{x}_j) + \sum_{(i,j)\in J_1} (x_i \bar{x}_j + \bar{x}_i x_j) \tag{3.4}$$

for some disjoint sets J_0 and J_1 of pairs of indices.

Let $d_0(x_i) = |\{(i,j) \in J_0\}|$ and $d_1(x_i) = |\{(i,j) \in J_1\}|$.

• $d_0(x_i)$ and $d_1(x_i)$ are of the same parity; i.e., $d_0(x_i) + d_1(x_i)$ is even for every variable x_i. To see this let X denote the 0-vector, and let X' denote the Boolean vector different from zero only at the ith coordinate. Then $g(X) - g(X')$ is even, but $g(X) - g(X')$ is equal to $d_0(x_i) - d_1(x_i)$.

• Consider now the graph G with vertex-set $\{1, 2, \cdots, n\}$ and edge-set $J_0 \cup J_1$. Every vertex has even degree in G. By a well-known theorem of Euler, it follows that $J_0 \cup J_1$ can be partitioned into edge disjoint cycles. This partitioning is easily seen to induce a similar decomposition of g into a sum of circulant posiforms.

The statement now follows immediately from Lemma 3.1 and the fact that g is odd. □

We now prove the announced statement.

THEOREM 3.3. $\mathbf{S}^{[3]}$ *is the Chvátal closure of* $\mathbf{S}^{[2]}$.

Proof. To show that $\mathbf{S}^{[2]\prime} \subseteq \mathbf{S}^{[3]}$, it is enough to prove that the constraints

$$\mathbf{L}_b(x,y) \geqq 0 \text{ for } b \in \mathscr{B}(\mathscr{P}_3) \setminus \mathscr{B}(\mathscr{P}_2)$$

are valid inequalities for $\mathbf{S}^{[2]\prime}$. The posiforms $b \in \mathscr{B}(\mathscr{P}_3) \setminus \mathscr{B}(\mathscr{P}_2)$ are exactly the posiforms $b = uvw + \bar{u}\bar{v}\bar{w}$ for some $\{u, v, w\} \subseteq L$, by Theorem 2.2. Therefore, by (3.3) $\mathbf{L}_{uvw+\bar{u}\bar{v}\bar{w}} + \frac{1}{2} \geqq 0$ is a linear consequence of $\mathbf{S}^{[2]}$; hence $\mathbf{L}_{uvw+\bar{u}\bar{v}\bar{w}} \geqq 0$ is a proper Chvátal cut for $\mathbf{S}^{[2]}$.

For the converse relation, $\mathbf{S}^{[3]} \subseteq \mathbf{S}^{[2]\prime}$, we show that any Chvátal cut for $\mathbf{S}^{[2]}$ is a valid inequality for $\mathbf{S}^{[3]}$. Let us consider a nonnegative linear combination of the constraints describing $\mathbf{S}^{[2]}$

$$a^T(x,y) - z' = \sum_{b\in\mathscr{B}(\mathscr{P}_2)} \alpha_b \mathbf{L}_b(x,y) \geqq 0, \tag{3.5}$$

$\alpha_b \geqq 0$. From this, we can obtain a proper Chvátal cut only if a is integral, $\lceil z' \rceil > z = \min_{(x,y)\in\mathbf{S}^{[2]}} a^T(x,y) \geqq z'$, and z is not integer. Here z is the optimum value of the linear programming problem $z = \min a^T(x,y)$ subject to $\mathbf{L}_b(x,y) \geqq 0$ for all $b \in \mathscr{B}(\mathscr{P}_2)$. Let B denote the matrix of a dual optimal basis of this problem and let $\beta^T \geqq 0$ be the corresponding dual solution, i.e., $a^T = \beta^T B$. Lemma 4 in [25] states that any basic submatrix of $\mathbf{S}^{[2]}$ can be transformed by elementary row and column operations to a form, containing 0s everywhere, except 2×2 blocks in the main diagonal, all of which have determinants equal to ± 2. This implies that there are unimodular matrices U and

V such that the components of $(UBV)^{-1}$ are all integer multiples of $\frac{1}{2}$. Since $2\beta^T = a^T V(2(UBV)^{-1})U$, it follows that the components of 2β are all integers. Now we have

$$a^T(x,y) - z = \sum_{b \in \mathscr{B}(\mathscr{P}_2)} \beta_b \mathbf{L}_b(x,y),$$

(where the summation runs only over the basic elements of $\mathscr{B}(\mathscr{P}_2)$); hence $2z$ is an integer and the resulting Chvátal cut has the form

$$a^T(x,y) \geqq z + \tfrac{1}{2}. \tag{3.6}$$

Furthermore, for this cut to be proper, $2z$ must be odd. Hence

$$g \overset{\text{def}}{=} 2\mathbf{L}_a^{-1} - 2z = \sum_{b \in \mathscr{B}(\mathscr{P}_2)} 2\beta_b b \tag{3.7}$$

is a quadratic posiform with integral coefficients, which takes only odd values. Then Lemma 3.2 applies, and $g - 1 \in \mathscr{P}_3$ follows. Thus $\mathbf{L}_{g-1} \geqq 0$; i.e., (3.6) is a linear consequence of $\mathbf{S}^{[3]}$. □

Finally, we show that the bound C_3 "strictly improves" on the roof duality bound C_2. This is stated more precisely in the following theorem.

THEOREM 3.4. *If* $\min_{x \in \{0,1\}^n} f(x) > C_2$, *then* $C_3 > C_2$.

Proof. By definition of C_2, there is a quadratic posiform $\Phi \in \mathscr{P}_2$ such that $f = C_2 + \Phi$. Let $\Phi(x, \bar{x}) = \sum_{u,v \in L} \alpha_{uv} uv$, where $\alpha_{uv} \geqq 0$ for $u, v \in L$. The inequality $\min f(x) > C_2$ is equivalent to the fact that the quadratic Boolean equation

$$\bigvee_{\alpha_{uv} > 0} uv = 0 \tag{3.8}$$

is inconsistent. This implies by [2] that there is a variable x and there are literals u_i, $i = 1, \cdots, k$ and v_j, $j = 1, \cdots, l$ such that α_{xu_1}, $\alpha_{\bar{u}_i u_{i+1}}$ for $i = 1, \cdots, k-1$, $\alpha_{\bar{u}_k x}$, $\alpha_{\bar{x}v_1}$, $\alpha_{\bar{v}_j v_{j+1}}$ for $j = 1, \cdots, l-1$, and $\alpha_{\bar{v}_l \bar{x}}$ are all positive. Let us choose $\varepsilon > 0$ such that 2ε is still less than any of these coefficients, and let Φ' be such that

$$f = C_2 + \Phi' + \varepsilon\left(xu_1 + \bar{u}_k x + \sum_{i=1}^{k-1} \bar{u}_i u_{i+1} + \bar{x}v_1 + \bar{v}_l \bar{x} + \sum_{j=1}^{l-1} \bar{v}_j v_{j+1}\right). \tag{3.9}$$

Here, by the selection of ε, Φ' is again a quadratic posiform. Thus to prove the theorem, it is enough to show that the quadratic expression in the brackets can also be expressed as the sum of a positive constant plus a cubic posiform. For this, let us consider the identity $2ab = (ab + \bar{a}\bar{b}) - 1 + a + b$, where a and b are arbitrary literals. If we apply this identity to every term in the brackets of (3.9), we get

$$\begin{aligned} &2\left(xu_1 + \bar{u}_k x + \sum_{i=1}^{k-1} \bar{u}_i u_{i+1} + \bar{x}v_1 + \bar{v}_l \bar{x} + \sum_{j=1}^{l-1} \bar{v}_j v_{j+1}\right) \\ &= \left[(xu_1 + \bar{x}\bar{u}_1) + \sum_{i=1}^{k-1} (\bar{u}_i u_{i+1} + u_i \bar{u}_{i+1}) + (\bar{u}_k x + u_k \bar{x})\right] \\ &\quad + \left[(\bar{x}v_1 + x\bar{v}_1) + \sum_{j=1}^{l-1} (\bar{v}_j v_{j+1} + v_j \bar{v}_{j+1}) + (\bar{v}_l \bar{x} + v_l x)\right]. \end{aligned} \tag{3.10}$$

Here both of the bracketed expressions are circular posiforms of the type (3.1); thus the statement follows from Lemma 3.1. □

A weaker statement stating that all fractional vertices of $\mathbf{S}^{[2]}$ are cut off by facets of $\mathbf{S}^{[3]}$ (namely, by the so-called "triangle inequalities," i.e., inequalities in (2.7) involving three indices) has been shown in [25]. It is a direct consequence of Theorem 3.4, also.

Note that Theorem 3.4 provides an indirect polynomial time test for the tightness of C_2 (see [20] for a more direct approach). By contrast, it was shown in [9], [11] that it is NP-hard to decide whether $\min_{x \in \{0,1\}^n} f(x) = C_3$.

4. Odd cycles in signed graphs. The close relationship between quadratic 0 – 1 minimization, on the one hand, and combinatorial problems like signed graph balancing or max-cut, on the other hand, has long been noted and exploited (see, e.g., [8], [10], [18], [19], [21], [26], etc.). For the sake of completeness, we now recall the necessary concepts (see [10] for a more thorough discussion).

Let G be a *weighted signed graph*, i.e., an undirected graph (V, E) together with a set of positive weights $\{\alpha_e | e \in E\}$ and a bipartition $\{E^+, E^-\}$ of the edge-set E. The edges in E^+ are called *positive*; those in E^- are called *negative*. A cycle of G is called *odd* if it contains an odd number of negative edges (odd cycles are often called negative, or frustrated; we use this terminology to stress the analogy between the signed graph balancing problem and the max-cut problem; see § 5). The graph G is *balanced* if it has no odd cycle [23].

The *balancing problem* for a weighted signed graph G is to find a subset $F \subseteq E$ of the edges, the deletion of which makes the graph balanced, and such that the total weight of F, i.e., $\sum_{e \in F} \alpha_e$, is minimal. This problem can easily be converted to a quadratic 0 – 1 minimization problem. Let x_i, $i \in V$ be 0 – 1 variables associated to the vertices of G. Let

$$\phi_G \overset{\text{def}}{=} \sum_{(i,j) = e \in E^-} \alpha_e(x_i x_j + \bar{x}_i \bar{x}_j) + \sum_{(i,j) = e \in E^+} \alpha_e(x_i \bar{x}_j + x_i \bar{x}_j). \tag{4.1}$$

Then the minimization of ϕ_G over $\{0, 1\}^{|V|}$ is equivalent with the graph balancing problem for G, and an optimal set F is formed by those edges for which the corresponding terms in ϕ_G do not vanish at the optimum (see [19]).

Conversely, to a quadratic pseudo-Boolean function f, we can associate a weighted signed graph G_f, such that the minimization of f corresponds in a natural way to the balancing problem for G_f. To see this, first consider the following definitions, inspired by formula (4.1).

For variables x and y, an expression $x\bar{y} + \bar{x}y$ is called a *positive bi-term*, while $xy + \bar{x}\bar{y}$ is called a *negative bi-term*.

If E denotes a collection of bi-terms such that no pair of variables is involved in more than one element of E, and $\alpha_e > 0$ for $e \in E$, then the quadratic pseudo-Boolean expression $\phi = \sum_{e \in E} \alpha_e e$ is called a *bi-form*. Note that $\phi(x) = \phi(\bar{x})$ for every bi-form ϕ and for every 0 – 1 vector x.

Bi-forms offer a natural representation of quadratic pseudo-Boolean functions, as shown below.

Remark 4.1. (see [10]). For any quadratic pseudo-Boolean function f in the variables $x_1, \cdots, x_n$ there is a unique constant c_f and a unique bi-form ϕ_f in the variables $x_0, x_1, \cdots, x_n$ such that

$$f(x_1, \cdots, x_n) = c_f + \phi_f(1, x_1, \cdots, x_n).$$

We call ϕ_f the *bi-form* of f. Note that, if f is given by its unique quadratic polynomial, then c_f and ϕ_f can easily be found in $O(n^2)$ time. Hence, the minimization of f is polynomially equivalent to the minimization of its bi-form.

To a quadratic pseudo-Boolean function f and its unique bi-form $\phi_f = \sum_{e \in E} \alpha_e e$, we can now associate a unique weighted signed graph G_f, as follows.

The vertices of G_f correspond to the indices $\{0, 1, \cdots, n\}$ of the variables of ϕ_f, and its edges correspond to those pairs (i, j) for which there is a bi-term in ϕ_f involving the variables x_i and x_j.

An edge of G_f is called *positive* (*negative*) if the associated bi-term is positive (negative); the weight of an edge is the positive coefficient α_e of the associated bi-term in ϕ_f.

With these definitions, it is easy to see that $\phi_{G_f} \equiv \phi_f$, and thus the minimization of f is equivalent to the balancing problem on G_f.

Example 4.1. Consider the quadratic pseudo-Boolean function given by

$$f = -3x_1 + 12x_2 - x_3 + 3x_4 + 14x_5 \\ -10x_1x_2 + 12x_1x_3 - 6x_1x_5 - 14x_2x_3 + 4x_3x_4 - 10x_4x_5.$$

The unique bi-form of f is then

$$\phi_f = 5(x_0\bar{x}_1 + \bar{x}_0x_1) + 6(x_0x_5 + \bar{x}_0\bar{x}_5) + 5(x_1\bar{x}_2 + \bar{x}_1x_2) \\ + 6(x_1x_3 + \bar{x}_1\bar{x}_3) + 3(x_1\bar{x}_5 + \bar{x}_1x_5) + 7(x_2\bar{x}_3 + \bar{x}_2x_3) \\ + 2(x_3x_4 + \bar{x}_3\bar{x}_4) + 5(x_4\bar{x}_5 + \bar{x}_4x_5),$$

satisfying the equation $f(x_1, \cdots, x_5) = \phi_f(1, x_1, \cdots, x_5) - 13$; i.e., $c_f = -13$. The corresponding weighted signed graph G_f is given on Fig. 1.

Consider now the following "cycle covering" formulation of the balancing problem for G_f:

$$\text{(4.2)} \qquad \text{(CC)} \qquad \begin{aligned} \text{minimize} \quad & \sum_{e \in E} \alpha_e y_e \\ \text{s.t.} \quad & \sum_{e \in C} y_e \geqq 1 \quad \forall C \in \mathscr{C}, \\ & y_e \in \{0, 1\} \quad \forall e \in E, \end{aligned}$$

where $\mathscr{C}$ denotes the collection of all odd cycles for G_f.

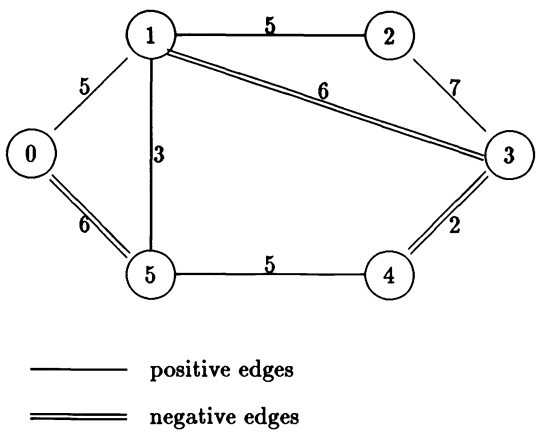


FIG. 1. *The graph G_f of the pseudo-Boolean function given in Example* 4.1.

The linear programming dual of the continuous relaxation of problem (CC) is the "cycle packing" problem, below:

$$\text{(4.3)} \qquad \text{(CP)} \qquad \begin{aligned} &\text{maximize} \sum_{C\in\mathscr{C}} \xi_C \\ &\text{s.t.} \quad \sum_{C\ni e} \xi_C \leqq \alpha_e \quad \forall e\in E(G_f), \\ &\qquad \xi_C \geqq 0 \qquad\quad \forall C\in\mathscr{C}. \end{aligned}$$

Denoting by $v(\text{CC})$ (respectively, $v(\text{CP})$) the optimal value of (CC) (respectively, (CP)), we can summarize the observations made so far in this section by the following relationships:

$$\text{(4.4)} \qquad \min_{x\in\{0,1\}^n} f(x) - c_f = \min_{x\in\{0,1\}^{n+1}} \phi_f(x) = v(\text{CC}) \geqq v(\text{CP}).$$

So $c_f + v(\text{CP})$ is a lower bound for the quadratic 0 – 1 minimization problem. The main result of this section is stated in the following result.

THEOREM 4.2. *For every quadratic pseudo-Boolean function f, $C_3 = c_f + v(\text{CP})$.*

This statement complements the result proved in [10, Thm. 3.1], that $C_2 - c_f$ is always equal to the optimal value of a further relaxation of the cycle packing problem.

The proof of Theorem 4.2 relies on a characterization of C_3, which was established in [9] and is restated below.

LEMMA 4.3. *For every quadratic pseudo-Boolean function f, C_3 is the largest constant c such that $f - c$ is a cubic posiform:*

$$C_3 = \max\{c \mid f = c + \phi, \phi\in\mathscr{P}_3, c \text{ constant}\}.$$

Proof. See [9]. □

For the sake of simplicity, we do not distinguish in what follows between edges of a signed graph and bi-terms of the associated bi-form. Thus, the notation $e \in E^+$ can indifferently refer to a positive edge $e = (i, j)$ of G or to a positive bi-term $(x_i\bar{x}_j + \bar{x}_i x_j)$ of ϕ_G. The same applies for the notation $e \in E^-$.

Proof of Theorem 4.2. Let ξ denote an optimal solution to problem (CP); i.e., $v(\text{CP}) = \sum_{C\in\mathscr{C}} \xi_C$. We construct a cubic posiform $\psi \in \mathscr{P}_3$, such that $f = c_f + v(\text{CP}) + \psi$, as follows. Observe that if C is an odd cycle in G_f, then $\sum_{e\in C} e$ is a posiform of type (3.1); hence Lemma 3.1 applies, and we have

$$\begin{aligned} f = c_f + \phi_f &= c_f + \sum_{e\in E(G_f)} \alpha_e e \\ &= c_f + \sum_{e\in E(G_f)} \left(\alpha_e - \sum_{C\ni e} \xi_C\right) e + \sum_{C\in\mathscr{C}} \xi_C \left(\sum_{e\in C} e\right) \\ &= c_f + \sum_{e\in E(G_f)} \left(\alpha_e - \sum_{C\ni e} \xi_C\right) e + \sum_{C\in\mathscr{C}} \xi_C (1 + 2q_C) \\ &= c_f + \sum_{C\in\mathscr{C}} \xi_C + \left[\sum_{e\in E(G_f)} \left(\alpha_e - \sum_{C\ni e} \xi_C\right) e + \sum_{C\in\mathscr{C}} 2\xi_C q_C\right] \\ &= c_f + v(\text{CP}) + \psi. \end{aligned}$$

Since $\psi \in \mathscr{P}_3$, the inequality $C_3 \geqq c_f + v(\text{CP})$ follows from Lemma 4.3.

For the converse inequality, let us use Lemma 4.3 to write $f = C_3 + \psi$, where $\psi \in \mathscr{P}_3$. Let also $W = C_3 - c_f$. We will show how to construct a feasible solution ξ of problem

(CP), which has an objective function value at least equal to W. This will imply $W \leqq v(\mathrm{CP})$, and hence $C_3 \leqq c_f + v(\mathrm{CP})$, as required.

Now comes the construction of ξ. By Theorem 2.2, ψ has the form

$$\psi = \sum \beta_u u + \sum \beta_{uv} uv + \sum \beta_{uvw}(uvw + \bar{u}\bar{v}\bar{w}),$$

where $u, v, w \in L$ are literals, and β_u, β_{uv}, and β_{uvw} are nonnegative reals. Introducing a new variable, x_0, we now define

$$\eta = \sum \beta_u(ux_0 + \bar{u}\bar{x}_0) + \sum \beta_{uv}(uvx_0 + \bar{u}\bar{v}\bar{x}_0) + \sum \beta_{uvw}(uvw + \bar{u}\bar{v}\bar{w})$$

$$= \sum \gamma_{uv}(uv + \bar{u}\bar{v}) + \sum \gamma_{uvw}(uvw + \bar{u}\bar{v}\bar{w}),$$

where u, v, w are literals from $L \cup \{x_0, \bar{x}_0\}$, and γ_{uv}, γ_{uvw} are nonnegative reals obtained from the β's. For this posiform η, we have $\psi(x_1, \cdots, x_n) = \eta(1, x_1, \cdots, x_n) = \eta(0, \bar{x}_1, \cdots, \bar{x}_n)$, and thus

$$f(x_1, \cdots, x_n) = c_f + W + \eta(1, x_1, \cdots, x_n). \tag{4.5}$$

If there is no cubic term in η, then η is a bi-form, and thus, by (4.5) and by the uniqueness of ϕ_f, we have $W = 0$. Then the feasible solution $\xi = 0$ is as required.

If there are cubic terms in η, then, using (3.3),

$$\gamma_{uvw}(uvw + \bar{u}\bar{v}\bar{w}) = \frac{\gamma_{uvw}}{2}[(uv + \bar{u}\bar{v}) + (vw + \bar{v}\bar{w}) + (wu + \bar{w}\bar{u})] - \frac{\gamma_{uvw}}{2}, \tag{4.6}$$

and we can represent η as a nonnegative combination of bi-terms, which we consider as edges of an associated graph G_η on the vertex set $\{0, 1, \cdots, n\}$. Note that, if we do not combine the coefficients of identical bi-terms after using identity (4.6), then we can look at G_η as a graph with parallel edges. In particular, to each cubic term $\gamma_{uvw}(uvw + \bar{u}\bar{v}\bar{w})$ of η, there corresponds an odd triangle $C = \{(uv + \bar{u}\bar{v}), (vw + \bar{v}\bar{w}), (wu + \bar{w}\bar{u})\}$ with weight $\delta_C = \gamma_{uvw}/2$. Let $T = \{(C, \delta_C)\}$ denote the collection of pairs of triangles and weights, corresponding to the cubic terms of η, and let $E = \{(e, \delta_e)\}$ denote the collection of pairs of edges $e = (uv + \bar{u}\bar{v})$ and weights $\delta_e = \gamma_{uv}$ corresponding to the quadratic terms of η. With these notations and with $W' = W$, we have

$$W + \eta \equiv W' + \sum_{(e,\delta_e) \in E} \delta_e e + \sum_{(C,\delta_C) \in T} \delta_C \left(-1 + \sum_{e \in C} e \right). \tag{4.7}$$

Assume now for a moment that the expression

$$\phi = \sum_{(e,\delta_e) \in E} \delta_e e + \sum_{(C,\delta_C) \in T} \delta_C \sum_{e \in C} e, \tag{4.8}$$

which appears in the right-hand side of (4.7), is a bi-form. Then, from identities (4.5) and (4.7), and using Remark 4.1, we can conclude that $W' = W = \sum_{(C,\delta_C) \in T} \delta_C$, $\phi = \phi_f$, and $G_\eta = G_f$. Since all triangles C such that $(C, \delta_C) \in T$ are odd cycles, the assignment

$$\xi_C = \begin{cases} \delta_C \text{ if } (C, \delta_C) \in T, \\ 0 \text{ otherwise} \end{cases}$$

clearly defines a feasible solution for problem (CP), with objective function value equal to W. Thus we are done in this case.

Unfortunately, the expression ϕ given by (4.8) is usually not a bi-form. Indeed, the same pair of variables may appear both in a positive and in a negative bi-term of ϕ, thus

contradicting our definition of a bi-form. With this in mind, we say that two edges e_1 and e_2 of G_η are in conflict if they have the same endpoints, but their signs are different; i.e., $e_1 = (uv + \bar{u}\bar{v})$ and $e_2 = (\bar{u}v + u\bar{v})$ for some literals u, v. Note that $e_1 + e_2 \equiv 1$. This identity will be used in what follows to eliminate all conflicting pairs of edges from G_η.

More precisely, we explain how T, E, and W' can be iteratively modified in such a way that the number of conflicting pairs of edges decreases at each iteration, while the following properties are preserved (*): (4.7) remains valid, $W' \geqq W$, the coefficients in T and E are nonnegative, and the cycles in T with positive weight are odd.

Observe that if we can maintain (*) while eliminating all conflicting pairs of edges, then the expression ϕ in (4.8) will eventually be transformed into a bi-form, and the theorem will follow as above.

We now explain the transformation of G_η. All properties (*) hold originally.

If there is a pair $(C_i, \delta_{C_i}) \in T$ for $i = 1, 2$, $\delta_{C_1} \geqq \delta_{C_2}$ in which exactly one pair of edges $e_i \in C_i$, $i = 1, 2$ are in conflict, let $D = (C_1 \cup C_2) \setminus \{e_1, e_2\}$. Then there is an odd cycle $C_3 \subseteq D$. Let $H = D \setminus C_3$ and apply the identity

$$\begin{aligned}
&\delta_{C_1}\left(-1 + \sum_{e \in C_1} e\right) + \delta_{C_2}\left(-1 + \sum_{e \in C_2} e\right) \\
&= (\delta_{C_1} - \delta_{C_2})\left(-1 + \sum_{e \in C_1} e\right) + \delta_{C_2}\left(-2 + \sum_{e \in C_1} e + \sum_{e \in C_2} e\right) \\
&= (\delta_{C_1} - \delta_{C_2})\left(-1 + \sum_{e \in C_1} e\right) + \delta_{C_2}\left(-1 + \sum_{e \in D} e\right) \\
&= (\delta_{C_1} - \delta_{C_2})\left(-1 + \sum_{e \in C_1} e\right) + \delta_{C_2}\left(-1 + \sum_{e \in C_3} e\right) + \delta_{C_2} \sum_{e \in H} e.
\end{aligned}$$

Thus, by replacing (C_1, δ_{C_1}) by $(C_1, \delta_{C_1} - \delta_{C_2})$ and (C_2, δ_{C_2}) by (C_3, δ_{C_2}) in T, and adding (e, δ_{C_2}) to E for $e \in H$, all the properties (*) hold.

If there is a pair $(C_i, \delta_{C_i}), \in T$ for $i = 1, 2$, $\delta_{C_1} \geqq \delta_{C_2}$, with $k > 1$ conflicting pairs of edges, say, $e_{1_j} \in C_1$ and $e_{2_j} \in C_2$ are in conflict for $j = 1, \cdots, k$, then let $H = (C_1 \cup C_2) \setminus \bigcup_{j=1}^{k} \{e_{1_j}, e_{2_j}\}$ and apply the equation

$$\delta_{C_1}\left(-1 + \sum_{e \in C_1} e\right) + \delta_{C_2}\left(-1 + \sum_{e \in C_2} e\right) = (\delta_{C_1} - \delta_{C_2})\left(-1 + \sum_{e \in C_1} e\right) + \delta_{C_2}\left(k - 2 + \sum_{e \in H} e\right).$$

Thus, by replacing (C_1, δ_{C_1}) by $(C_1, \delta_{C_1} - \delta_{C_2})$ and deleting (C_2, δ_{C_2}) from T, adding the pairs (e, δ_{C_2}) to E for $e \in H$, and modifying W' by $W' = W' + (k - 2)\delta_{C_2}$, we can simplify G_η, still maintaining properties (*).

If there is a pair $(a, \delta_a) \in E$ and $(C, \delta_C) \in T$, $\delta_a \geqq \delta_C$ such that a is in conflict with $b \in C$, then we have

$$\delta_a a + \delta_C\left(-1 + \sum_{e \in C} e\right) = (\delta_a - \delta_C)a + \delta_C \sum_{e \in C \setminus \{b\}} e.$$

In this case, we delete (C, δ_C) from T, replace (a, δ_a) by $(a, \delta_a - \delta_C)$, and add (e, δ_C) to E for $e \in C \setminus \{a\}$. If $\delta_C \geqq \delta_a$ in a similar case, then

$$\delta_a a + \delta_C\left(-1 + \sum_{e \in C} e\right) = \delta_a \sum_{e \in C \setminus \{b\}} e + (\delta_C - \delta_a)\left(-1 + \sum_{e \in C} e\right);$$

thus we delete the pair (a, δ_a) from E, replace (C, δ_C) by $(C, \delta_C - \delta_a)$ in T, and add $(e, \delta_C - \delta_a)$ to E for $e \in C \setminus \{a\}$. It is easy to check that properties $(*)$ still hold.

Finally, if $(a, \delta_a) \in E$ and $(b, \delta_b) \in E$, such that a and b are in conflict, $\delta_a \geqq \delta_b$, then

$$\delta_a a + \delta_b b = \delta_b + (\delta_a - \delta_b)a,$$

hence deleting (b, δ_b) from E, replacing (a, δ_a) by $(a, \delta_a - \delta_b)$ in E, and modifying W' by $W' = W' + \delta_b$, $(*)$ is still satisfied.

In each of the above steps, we delete also those pairs from E and T, which have weight 0.

It is straightforward to verify that repeating these steps we can arrive to a form in which there are no more conflicts, thus proving the theorem. □

Let us remark that the first half of the statement, the inequality $C_3 \geqq c_f + v(\mathrm{CP})$, also follows from the fact that the odd cycle inequalities are linear consequences of the triangle inequalities, shown in [6], [25]. This linear dependency is implied by identity (3.3), or by Lemma 3.1, which were used in our proof.

5. Consequences for the max-cut problem. An immediate and useful consequence of Theorems 2.2 and 4.2 is that a compact linear programming formulation can be given for certain combinatorial problems, which are originally described by a (possibly) exponentially large set of inequalities.

One such problem is to find a maximum weight cut in a weakly bipartite graph.

More generally, given a graph $G = (V, E)$, let $P_B(G)$ denote the convex hull of the incidence vectors of those subsets of edges that form bipartite subgraphs of G. Given nonnegative weights α_e $(e \in E)$ on the edges, the *max-cut problem* for G is $\max \sum_{e \in E} \alpha_e y_e$ subject to $y \in P_B(G)$. (See [7], [17].)

A cycle of G is *odd* if it contains an odd number of edges (this is consistent with the definition given in § 4, if we consider G as a signed graph with negative edges only). Let $\mathscr{C}$ be the collection of odd cycles of G. The following relaxation of the max-cut problem was introduced in [17]:

$$\max \sum_{e \in E} \alpha_e y_e, \tag{5.1}$$

$$\sum_{e \in C} y_e \leqq |C| - 1 \quad \forall C \in \mathscr{C}, \tag{5.2}$$

$$0 \leqq y_e \leqq 1 \qquad \forall e \in E.$$

It was shown in [17] that the separation problem for the system of inequalities (5.2) can be solved in polynomial time. Hence, the LP (5.1)–(5.2) is polynomially solvable by the ellipsoid method. Those graphs for which $P_B(G)$ is exactly described by the constraints (5.2) are called *weakly bipartite*. It was proved in [14] that the graphs not contractible to K_5 are weakly bipartite.

We obtain now, as a corollary of Theorem 4.2, the following theorem.

THEOREM 5.1. *The problem* (5.1)–(5.2) *is equivalent to a linear programming problem involving* $n + \binom{n}{2}$ *variables and* $4\binom{n}{2} + 4\binom{n}{3}$ *constraints with* $-1, 0, +1$ *coefficients, where* $n = |V| - 1$.

Proof. To see this, let us switch to the variables $z_e = 1 - y_e$ $(e \in E)$ in (5.1)–(5.2),

$$\min \sum_{e \in E} \alpha_e z_e,$$

$$\sum_{e \in C} z_e \geqq 1 \quad \forall C \in \mathscr{C}, \tag{5.3}$$

$$0 \leqq z_e \leqq 1 \quad \forall e \in E.$$

Let v_1 (respectively, v_2) denote the optimal value of (5.1)–(5.2) (respectively, (5.3)). Clearly, $v_1 = \sum_{e \in E} \alpha_e - v_2$. Let $V = \{0, 1, \cdots, n\}$ and make G a signed graph by assuming that all its edges are negative. Then problem (5.3) is just the linear programming dual of problem (CP) corresponding to G. Hence, by Theorem 4.2, $v_2 = C_3$ for the corresponding pseudo-Boolean function $f = f_G$ defined by

$$f_G(1, x_1, \cdots, x_n) = \sum_{e=(0,i)\in E, i\neq 0} \alpha_e x_i + \sum_{e=(i,j)\in E, i,j\neq 0} \alpha_e (x_i x_j + \bar{x}_i \bar{x}_j).$$

($c_{f_G} = 0$, with the above definition). Thus, Theorem 2.3 implies the desired result. □

We remark that the above proof uses only the first half of Theorem 4.2. A similar result to Theorem 5.1 has been proved in [6] (see also [25]).

REFERENCES

[1] W. P. ADAMS AND P. M. DEARING, *On the equivalence between roof duality and Lagrangian duality for unconstrained* 0 – 1 *quadratic programming problems*, Tech. Report No. URI-061, Clemson University, Clemson, SC, September 1988.

[2] B. APSVALL, M. F. PLASS, AND R. E. TARJAN, *A linear-time algorithm for testing the truth of certain quantified Boolean formulas*, Inform. Process. Lett., 8 (1979), pp. 121–123.

[3] E. BALAS AND J. B. MAZZOLA, *Nonlinear* 0 – 1 *programming*: I. *Linearization techniques*. II. *Dominance relations and algorithms*, Math. Programming, 30 (1984), pp. 1–45.

[4] M. BALINSKI, *Integer programming*: *Methods*, *uses*, *computation*, in Mathematics of the Decision Science, Part I, G. B. Dantzig and A. F. Veinott, Jr., eds., American Mathematical Society, Providence, RI, 1968, pp. 179–256.

[5] F. BARAHONA, *The max-cut problem in graphs not contractible to* K_5, Oper. Res. Lett., 2 (1983), pp. 107–111.

[6] ———, *On cuts and matchings in planar graphs*, Research Report No. 88503-OR, Institut fur Operations Research, Universität Bonn, Bonn, Germany, 1988.

[7] F. BARAHONA, M. GRÖTSCHEL, AND A. R. MAHJOUB, *Facets of the bipartite subgraph polytope*, Math. Oper. Res., 10 (1985), pp. 340–358.

[8] F. BARAHONA AND A. R. MAHJOUB, *On the cut polytope*, Math. Programming, 36 (1968), pp. 157–173.

[9] E. BOROS, Y. CRAMA, AND P. L. HAMMER, *Upper bounds for quadratic* 0 – 1 *maximization*, Oper. Res. Lett., 9 (1990), pp. 73–79.

[10] E. BOROS AND P. L. HAMMER, *A max-flow approach to improved roof duality in quadratic* 0 – 1 *minimization*, RUTCOR Research Report 15-89, Rutgers University, New Brunswick, NJ, 1989.

[11] J.-M. BOURJOLLY, P. L. HAMMER, W. R. PULLEYBLANK, AND B. SIMEONE, *Combinatorial methods for bounding a quadratic pseudo-Boolean function*, RUTCOR Research Report 27-89, Rutgers University, New Brunswick, NJ, 1989.

[12] V. CHVÁTAL, *Edmonds polytopes and a hierarchy of combinatorial problems*, Discrete Math., 4 (1973), pp. 305–337.

[13] Y. CRAMA, *Linearization techniques and concave extensions in nonlinear* 0 – 1 *optimization*, RUTCOR Research Report 32-88, Rutgers University, New Brunswick, NJ, 1988.

[14] J. FONLUPT, A. R. MAHJOUB, AND J.-P. UHRY, *Composition of graphs and the bipartite subgraph polytope*, Research Report #459, Lab. ARTEMIS (IMAG), University de Grenoble, Grenoble, France, 1984.

[15] R. FORTET, *Applications de l'algebre de Boole en recherche operationelle*, Revue Francaise de Recherche Operationelle, 4 (1960), pp. 17–26.

[16] F. GLOVER AND E. WOOLSEY, *Converting the* 0 – 1 *polynomial programming problem to a* 0 – 1 *linear problem*, Oper. Res., 22 (1974), pp. 180–182.

[17] M. GRÖTSCHEL AND W. R. PULLEYBLANK, *Weakly bipartite graphs and the max-cut problem*, Oper. Res. Lett., 1 (1981), pp. 23–27.

[18] P. L. HAMMER, *Some network flow problems solved with pseudo-Boolean programming*, Oper. Res., 13 (1965), pp. 388–399.

[19] ———, *Pseudo-Boolean remarks on balanced graphs*, Internat. Ser. Numer. Math., 36 (1977), pp. 60–78.

[20] P. L. HAMMER, P. HANSEN, AND B. SIMEONE, *Roof duality, complementation and persistency in quadratic* 0 – 1 *optimization*, Math. Programming, 28 (1984), pp. 121–155.

[21] P. HANSEN, *Methods of nonlinear* 0 - 1 *programming*, Ann. Discrete Math., 5 (1979), pp. 53–70.
[22] P. HANSEN, S. H. LU, AND B. SIMEONE, *On the equivalence of paved-duality and standard linearization in nonlinear optimization*, Discrete Appl. Math., 29 (1990), pp. 187–193.
[23] F. HARARY, *On the notion of balance of a signed graph*, Michigan Math. J., 2 (1953/1954), pp. 143–146.
[24] S. H. LU AND A. C. WILLIAMS, *Roof duality for* 0 - 1 *nonlinear optimization*, Math. Programming, 37 (1987), pp. 357–360.
[25] M. PADBERG, *The Boolean quadric polytope: Some characteristics, facets and relatives*, Math. Programming, 45 (1989), pp. 139–172.
[26] J. C. PICARD AND H. D. RATLIFF, *A graph theoretic equivalent for integer programs*, Oper. Res., 21 (1973), pp. 261–269.
[27] J. RHYS, *A selection problem of shared fixed costs and network flows*, Management Sci., 17 (1970), pp. 200–207.
[28] A. SCHRIJVER, *Theory of Linear and Integer Programming*, John Wiley, New York, 1986.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 178–198, May 1992
© 1992 Society for Industrial and Applied Mathematics
003

MINIMUM TIME BROADCAST NETWORKS TOLERATING A LOGARITHMIC NUMBER OF FAULTS*

LUISA GARGANO† AND UGO VACCARO†

Abstract. Consider a network in which n processors are connected by communication lines and are allowed to communicate with at most one other processor at a time. Broadcast is the task of transmitting a message originated at one node to all other nodes in the network. Presented in this paper is a broadcasting scheme that can tolerate up to $k \leqq \lfloor \log n \rfloor$ line failures; that is, it assures that each node in the network will receive the message from the originator when up to $k \leqq \lfloor \log n \rfloor$ lines are faulty. The time required by the broadcast protocol is minimum, except in some cases that might require one unit of time more than the minimum. Moreover, an algorithm for constructing networks supporting the broadcast scheme and having approximately the minimum possible number of lines is given.

Key words. communication networks, broadcasting, fault-tolerant communication

AMS(MOS) subject classifications. 05C38, 94C15

1. Introduction.

Consider a communication network consisting of processors that communicate by exchanging messages through bidirectional channels. Broadcasting is the process of delivering a message from a processor to all other processors in the network. We represent the network by a connected undirected graph $G = (V, E)$, where the set of nodes V represents the set of processors of the network and the set of edges E corresponds to the communication lines between processors. Nodes connected by a line are called *neighbors*. The processor that has a message to be delivered to all other processors is called the *originator*. The model we consider in this paper assumes that the communication lines are bidirectional and obeys the following constraints:

(1) each call requires one unit of time;

(2) any processor may participate in at most one call per time unit;

(3) any processor may only call a neighbor.

The broadcast problem and this model in particular seem to be of natural interest in the design of distributed networks and parallel computing systems and have been studied by several authors [2]–[9], [11]–[15].

Consider a network $G = (V, E)$ and let the originator be the node $u \in V$. The broadcast time $t(u, G)$ is the minimum number of time units necessary to broadcast from u on G. *The broadcast time of the network* G, denoted by $t(G)$, is the maximum broadcast time from any node u of G. Let $T(n)$ be the minimum of $t(G)$ over all networks with n nodes. Since the number of nodes that may have received the message from the originator can at most double after each time instant, it is clear that $T(n) \geqq \lceil \log n \rceil$.[1] The problem of constructing sparse communication networks having broadcast time *exactly* equal to $\lceil \log n \rceil$ was first investigated by Farley [5], and several algorithms have been subsequently given to improve on Farley's results [2], [3], [8]. A recent result by Grigni and Peleg [9] shows that the minimum number of communication lines of a network having broadcast time $\lceil \log n \rceil$ belongs to $\Theta(L(n-1)n)$, where $L(n)$ is the number of consecutive leading ones in the binary representation of n.

* Received by the editors April 18, 1988; accepted for publication (in revised form) October 26, 1990. This work was partially supported by the Italian Ministry of Universities and Scientific Research and by the National Council of Research (CNR), "Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo." Additional support to the first author was provided by a Latin American Scholarship Program of American Universities—Southern Italy Fellowship Program fellowship. The work of the first author was done while she was visiting the Computer Science Department of Columbia University, New York, New York 10027.

† Dipartimento di Informatica ed Applicazioni, Università di Salerno, 84081 Baronissi (SA), Italy.

[1] All logarithms in this paper are in base 2.

Liestman [12] first introduced the important issue of fault-tolerant broadcasting. Consider a network $G = (V, E)$ and let $E' \subset E$ be any set of edges, $|E'| \leq k$. The set of edges E' represents faulty communication lines and the subgraph $G' = (V, E - E')$ represents the functioning part of the network. A k fault-tolerant broadcasting scheme is a broadcast protocol that assures that any node in the network will receive the message from the originator in presence of up to k line failures, i.e., when *any* set E' of size $|E'| \leq k$ is faulty. It is important to point out that in this model the sequence of message transmissions (*calls*) is fixed and cannot be changed when faults are detected.

In analogy with the fault-free case, we denote by $t_k(u, G)$ the minimum time needed for a k fault-tolerant broadcast from the vertex u to all the other nodes in G. Moreover, we define the k fault-tolerant broadcast time of the network G, $t_k(G)$, as the maximum of $t_k(u, G)$ over all nodes u in G and denote by $T_k(n)$ the minimum of $t_k(G)$ over all networks with n nodes. The study of the function $T_k(n)$ was initiated by Liestman [12] who provided the following lower bound:

$$T_k(n) \geq \lceil \log n \rceil + k \quad \text{for } n - 2 \geq k \geq 1.$$

Moreover, Liestman proved that

$$T_1(n) = \lceil \log n \rceil + 1$$

and

$$T_2(n) = \begin{cases} \lceil \log n \rceil + 2 & \text{if } n \geq 5, \quad n \neq 2^i - 1 \\ \lceil \log n \rceil + 3 & \text{if } n \geq 7, \quad n = 2^i - 1. \end{cases}$$

The problem of establishing general upper bounds on $T_k(n)$ was considered by Peleg and Schäffer [15] and by Maddaluno [13], who showed that $T_k(n) \leq \lceil \log n \rceil + 2k + 4$. Moreover, Peleg and Schäffer provided an alternative k fault-tolerant broadcast scheme that has a better performance when $k \gg \log n$.

The above papers left open the problem of determining the exact value of $T_k(n)$ for $k \geq 3$, that is, the problem of designing $k \geq 3$ fault-tolerant broadcasting schemes which require minimum time. This is the problem we consider in the first part of the work. In particular, in § 2 we present a broadcasting scheme on an even number of nodes that can tolerate up to $k \leq \lfloor \log n \rfloor$ line failures and that can be completed in $\lceil \log n \rceil + k$ (i.e., minimum) time units. In § 3 we modify the broadcast scheme of § 2 so that it can be applied to an odd number of nodes. We show that the number of time units required to complete the process is at most one more than the lower bound. These results allow us to determine the values of $T_k(n)$, $n \geq 8$, as shown below:

$$T_k(n) = \begin{cases} \lceil \log n \rceil + k & \text{for } n \text{ even and } k \leq \lfloor \log n \rfloor \text{ or} \\ & n \text{ odd and } k \leq \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor; \\ \lceil \log (n-1) \rceil + k + 1 & \text{for } n \text{ odd and } k \text{ satisfying the two inequalities} \\ & k \geq 2^{\lceil \log n \rceil} - n - 1 + \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor, \\ & k \leq \lfloor \log (n-1) \rfloor; \\ \lceil \log n \rceil + k \text{ or } \lceil \log (n-1) \rceil + k + 1 & \\ & \text{for } n \text{ odd and the remaining } k \leq \lfloor \log (n-1) \rfloor. \end{cases}$$

In the second part of the paper we consider the problem of constructing sparse broadcast networks supporting minimum time k fault-tolerant broadcasting schemes.

This problem has been studied in [4] and [12] in the case where $k = 1, 2$. In § 4 we give an algorithm to construct broadcast networks having $O(n(k + \lceil \log n \rceil)/2)$ edges that support the k fault-tolerant broadcast protocols presented in §§ 2 and 3. This implies that the number of edges of such networks differs from the minimum possible number of edges of any broadcast network tolerating a logarithmic number of faults in at most a constant factor. In particular, for each $k \leqq \lfloor \log n \rfloor$, they have the minimum possible number of edges when n is a power of 2.

2. Fault-tolerant broadcast—an even number of nodes. In this section we provide an algorithm to perform k fault-tolerant broadcast on n processors in minimum time, that is, in $\lceil \log n \rceil + k$ time units, for any n even and $k \leqq \lfloor \log n \rfloor$. In this and the next section we will assume that the network is represented by a complete graph.

For the sake of simplicity, let the n processors (hereafter simply referred to as *nodes*) be labeled by the integers $\{0, 1, \cdots, n-1\}$, $n \geqq 8$. Let us first describe the general strategy of the broadcast. We recall that the goal of the k fault-tolerant broadcasting process is to create $k+1$ edge-disjoint calling paths from the originator to each node in the network. Moreover, we want to ensure that the message sent by the originator reaches each node within $\lceil \log n \rceil + k$ time units.

We proceed in two steps. In the first step each node receives the message exactly once. During the second step the remaining k paths from the originator to each node are created. The task of the first step is accomplished in the following way: At time unit 1, the node set is divided into two subsets of consecutive numbered nodes, $\{0, \cdots, n/2 - 1\}$ and $\{n/2, \cdots, n-1\}$. The originator calls a node in the subset it does not belong to. In this way there are two equally sized subsets of nodes and in each one there is an informed node. The node that has been called by the originator will broadcast on its subset, while the originator continues broadcasting on the other half of nodes. In general after time unit $t-1$ each informed node has to broadcast on a set of consecutive numbered nodes. This set is divided into two subsets of consecutive numbered nodes such that either they have equal size or the one the informed node belongs to has one element more than the other. The informed node calls a node in the other subset, which will broadcast on it, while it continues broadcasting on its subset. The process continues for $\lceil \log n \rceil$ time units at the end of which there is a calling path from the originator to each node in the network. During the second step the nodes in $\{0, \cdots, n/2 - 1\}$ exchange calls, for k consecutive time units, with nodes in $\{n/2, \cdots, n-1\}$. The second step creates the remaining k paths from the originator to each node in the network.

Let 0 be the originator and the number n of nodes be even. In order to formally describe the first step, consider the following partition of the node set $\{0, \cdots, n-1\}$ into sets $A_1, \cdots, A_{\lceil \log n \rceil + 1}$ defined by

$$A_t = \left\{ \left\lceil \frac{n}{2^t} \right\rceil, \left\lceil \frac{n}{2^t} \right\rceil + 1, \cdots, \left\lceil \frac{n}{2^{t-1}} \right\rceil - 1 \right\} \quad \text{for } t = 1, \cdots, \lceil \log n \rceil \tag{1}$$

$$A_{\lceil \log n \rceil + 1} = \{0\}.$$

Note that the sizes of the sets A_t satisfy the following relations:

$$|A_t| \in \{\lceil n/2^t \rceil, \lceil n/2^t \rceil - 1\} \tag{2}$$

and

$$\lceil n/2^t \rceil = n - \sum_{i=1}^{t} |A_i|. \tag{3}$$

Consider then the following broadcast scheme from 0 to $\{0, \cdots, n-1\}$.

Time unit 1. 0 calls the node $n-1 \in A_1$ that, during the following $\lceil \log n \rceil - 1$ time units, broadcasts to the $n/2 \leqq 2^{\lceil \log n \rceil - 1}$ nodes in A_1;

Time unit t $(2 \leqq t \leqq \lceil \log n \rceil)$. 0 calls the node $\lceil n/2^t \rceil \in A_t$ that, during the following $\lceil \log n \rceil - t$ time units, broadcasts to the $|A_t| \leqq 2^{\lceil \log n \rceil - t}$ nodes in A_t.

The broadcast from the informed node in A_t to the other nodes in A_t can be performed in $\lceil \log n \rceil - t$ time units according to the protocol previously outlined. We should point out, however, that any of the known algorithms (e.g., [3], [5], [8], [9]) can be applied, since $|A_t| \leqq 2^{\lceil \log n \rceil - t}$. Therefore the above broadcasting scheme, in absence of line failures, informs each node in the network in $\lceil \log n \rceil$ time units.

Example 1. Let the node set be $\{0, \cdots, 9\}$. One has $A_1 = \{5, 6, 7, 8, 9\}$, $A_2 = \{3, 4\}$, $A_3 = \{2\}$, $A_4 = \{1\}$, and $A_5 = \{0\}$. A calling tree on the node set $\{0, \cdots, 9\}$, constructed according to the above rules, is given in Fig. 1.

To obtain a k fault-tolerant broadcast protocol it is necessary to introduce some redundancy into the above scheme. To achieve this, we introduce a second series of calls referred to as the *second step*. It consists of calls exchanged between nodes in the set $\{0, \cdots, n/2 - 1\}$ and nodes in $\{n/2, \cdots, n-1\}$, for k consecutive time units. As we analyze the second step, time unit t will denote the tth time unit of the second step, unless otherwise specified.

During the second step each node (if informed) calls according to the following rule.

Time unit t $(1 \leqq t \leqq k \leqq \lfloor \log n \rfloor)$. Each node $x \in \{0, \cdots, n-1\}$ (if informed) calls node $f_t^{(n)}(x)$ defined as

$$f_t^{(n)}(x) = \begin{cases} \dfrac{n}{2} + \left(x \oplus \left\lceil \dfrac{n}{2^t} \right\rceil \right) & \text{if } x < n/2, \\ \left(x \ominus \left\lceil \dfrac{n}{2^t} \right\rceil \right) & \text{if } x \geqq n/2, \end{cases} \tag{4}$$

where $\ominus$ and $\oplus$ denote, respectively, the subtraction and the addition modulo $n/2$.

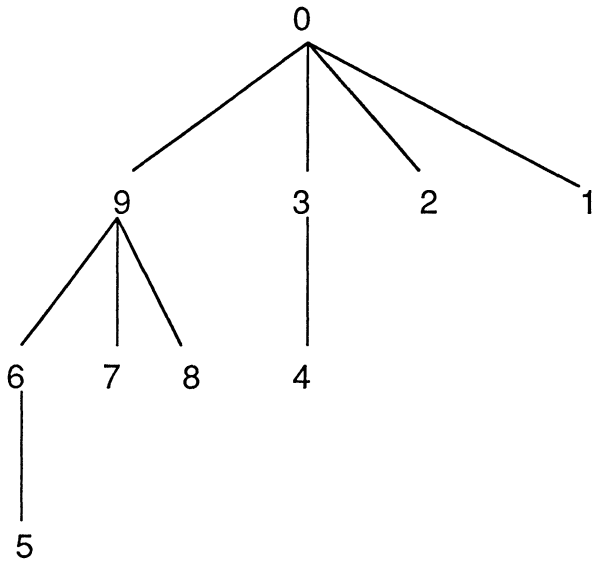


FIG. 1. *Broadcast tree for* $n = 10$ *nodes.*

Note that $f_t^{(n)}(f_t^{(n)}(x)) = x$, that is,

$$f_t^{(n)}(x) = y \text{ if and only if } f_t^{(n)}(y) = x, \tag{5}$$

and that

$$f_t^{(n)}(x) \neq f_s^{(n)}(x) \quad \text{for each } s \text{ and } t \text{ such that } 1 \leqq s < t \leqq \lfloor \log n \rfloor. \tag{6}$$

Since the even integer n can be arbitrary but it is fixed, for sake of simplicity we will write $f_t(x)$ for $f_t^{(n)}(x)$ throughout the rest of this section.

We will show that the calls of the first step followed by the calls of the first k time units of the second step give a k fault-tolerant broadcast protocol; that is, each node receives the message along $k+1$ edge disjoint paths, within the kth time instant of the second step.

An *informer* of a node x at time t is any node a from which x receives the message (in the absence of failures) during the first t time units of the second step. More precisely, a is an informer of x at time t if either $a = x$ or the calls of the first t time units of the second step form a path

$$(a, f_{i_1}(a) = a_1)(a_1, f_{i_2}(a_1) = a_2) \cdots (a_{r-1}, f_{i_r}(a_{r-1}) = x),$$

for some integers $i_1, i_2, \cdots, i_r$, $1 \leqq r \leqq t$, such that $1 \leqq i_1 < \cdots < i_r \leqq t$. We denote by $\inf(x, t)$ the set of informers of x at time t. Formally, $\inf(x, 0) = \{x\}$ and for each $t \leqq \lfloor \log n \rfloor$ we have $\inf(x, t) = \{x\} \cup \{a \mid$ there exist integers $1 \leqq i_1 < \cdots < i_r \leqq t$ such that $f_{i_r}(\cdots(f_{i_1}(a))) = x\}$. From the definition of $\inf(x, t)$ and from (4) and (5) we get

$$\begin{aligned}\inf(x,t) &= \inf(x,t-1) \cup \{a \mid \text{ exist } 1 \leqq i_1 < \cdots < i_s \leqq t-1 \text{ with } f_t(f_{i_s}(\cdots(f_{i_1}(a)))) = x\} \\ &= \inf(x,t-1) \cup \{a \mid \text{ exist } 1 \leqq i_1 < \cdots < i_s \leqq t-1 \\ &\qquad \text{with } f_{i_s}(\cdots(f_{i_1}(a))) = f_t(x)\} \cup \{f_t(x)\}\end{aligned}$$

and, therefore,

$$\inf(x,t) = \inf(x,t-1) \cup \inf(f_t(x), t-1). \tag{7}$$

Example 2. For $n = 10$ and $x = 3$ we have

$$\inf(3,0) = \{3\}, \inf(3,1) = \{3,8\}, \inf(3,2) = \{1,3,6,8\},$$
$$\inf(3,3) = \{0,1,2,3,5,6,7,8\}.$$

The paths to node 3 from its informers are shown in Fig. 2; their edges are represented by solid lines. Dashed lines represent the edges used during the first step.

The following properties of the sets $\inf(x, t)$ will be useful to prove the correctness of the proposed broadcast scheme. Let S_t be the set formed by 0 and the $2^{t-1} - 1$ possible sums of elements in $\{|A_2|, \cdots, |A_t|\}$, e.g., $S_1 = \{0\}$, $S_2 = \{0, |A_2|\}$, $S_3 = \{0, |A_2|, |A_3|, |A_2| + |A_3|\}$, and so on.

Property 1. For each node x and time instant t, $1 \leqq t \leqq \lfloor \log n \rfloor$,

$$\inf(x,t) = \begin{cases} \{x \ominus z \mid z \in S_t\} \cup \{n/2 + (x \ominus z) \mid z \in S_t\} & \text{if } x < n/2, \\ \{x \oplus z \mid z \in S_t\} \cup \{n/2 + (x \oplus z) \mid z \in S_t\} & \text{if } x \geqq n/2. \end{cases}$$

Property 2. For each node x and time instant t, $1 \leqq t \leqq \lfloor \log n \rfloor$, the set of informers of x at time t, $\inf(x, t)$, is equal to a set $\{a_1, \cdots, a_{2^t}\}$ with $a_1 < a_2 < \cdots < a_{2^t}$ satisfying

$$n + (a_1 - a_{2^t}), a_i - a_{i-1} \in \{\lceil n/2^t \rceil, \lceil n/2^t \rceil - 1\}, \qquad i = 2, \cdots, 2^t. \tag{8}$$

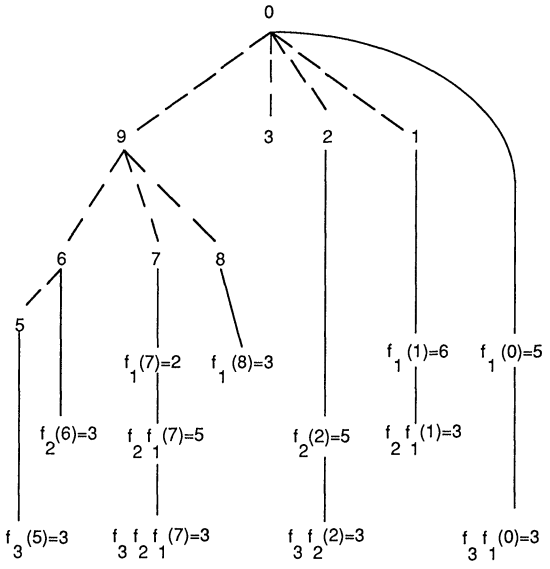


FIG. 2. *Broadcast for n = 10: Paths to node 3 from its informers are represented by solid edges; dashed edges represent those used during the first step.*

Property 3. For each node x and time instant t, $1 \leqq t \leqq \lfloor \log n \rfloor$,

$$\inf(x, t-1) \cap \inf(f_t(x), t-1) = \varnothing.$$

The proofs of Properties 1–3 are given in Appendix 1.

A *t-path to a node x* is any calling path from the originator to x formed by calls of the first step and of the first t time units of the second step.

Given a t-path P to a node x, there exists exactly one node $I(P) \in \inf(x, t)$ such that $P = RQ$ where

— R is a path, created by the calls of the first step, from the originator to $I(P)$;

— Q is a path from $I(P)$ to x which is created by the calls of the first t time units of the second step; that is, $Q = (I(P), f_{i_1}(I(P)) = a_1)(a_1, f_{i_2}(a_1) = a_2) \cdots (a_{i_r-1}, f_{i_r}(a_{i_r-1}) = x)$ for some integers $i_1, i_2, \cdots, i_r$ such that $1 \leqq i_1 < \cdots < i_r \leqq t$; Q is empty if $I(P) = x$.

As an example, consider the calling tree of Example 3 given in Fig. 2: the 3-path $P_1 = (0, 1)(1, 6)(6, 3)$ from 0 to 3 has $R_1 = (0, 1)$ (dashed line), $Q_1 = (1, 6)(6, 3)$ (solid lines), and $I(P_1) = 1$; the 3-path $P_2 = (0, 2)(2, 5)(5, 3)$ has $R_2 = (0, 2)$, $Q_2 = (2, 5)(5, 3)$, and $I(P_2) = 2$.

LEMMA 1. *Let $P = RQ$ be a $(t-1)$-path from the originator to node x and $P' = R'Q'$ be a $(t-1)$-path from the originator to $f_t(x)$. The paths P and $P'(f_t(x), x)$ are edge disjoint if and only if $I(P)$ and $I(P')$ do not belong to the same set A_i, for any $i \in \{1, \cdots, \lceil \log n \rceil\}$.*

Proof. Necessity. Since the first step creates a tree in which the subtrees of the originator have node sets $A_1, \cdots, A_{\lceil \log n \rceil}$, if $I(P)$ and $I(P')$ belong to the same A_i for some i then the paths R and R' share the first edge, that is, the edge from 0 to some node in A_i.

Sufficiency. Suppose that $I(P)$ and $I(P')$ do not belong to the same subtree (i.e., to some set A_i). Since the calls of the first step create a tree with subtrees having node sets $A_1, \cdots, A_{\lceil \log n \rceil}$, it follows that the paths formed by the first step R and R' are edge-

disjoint. Moreover, since the edges used during the first and the second step are different, there cannot be common edges between the paths R and $Q'(f_t(x), x)$, or the paths R' and Q. It remains to show that the paths Q and $Q'(f_t(x), x)$ are edge-disjoint. Suppose first that Q and Q' share an edge $(a, f_r(a))$, for some $1 \leqq r \leqq t-1$. We can write

$$Q=\beta(a,f_r(a))\gamma \quad \text{and} \quad Q'=\beta'(a,f_r(a))\gamma' \quad \text{or} \quad Q'=\beta'(f_r(a),a)\gamma'.$$

This implies that $a \in \inf(x, t-1)$ and $a \in \inf(f_t(x), t-1)$, which contradicts Property 3. Finally, since $Q = \alpha(f_s(x), x)$ for some $s \leqq t-1$, from (6) we get that the edge $(f_t(x), x)$ does not appear in Q and the paths Q and $Q'(f_t(x), x)$ are edge disjoint. □

We are now able to prove the desired result.

THEOREM 1. *The sequence of* $\lceil \log n \rceil$ *calls performed in the first step followed by the calls of the first* $k \leqq \lfloor \log n \rfloor$ *time units of the second step gives a* k *fault-tolerant broadcast protocol.*

Proof. We show that the calls of the first step followed by those of the first k time units of the second step create $k+1$ edge disjoint paths from the originator of the broadcast to each other node. The proof is by induction on k. For $k = 0$ the above assertion is true since each node receives the information during the first step. Suppose the theorem is true for $k-1$; we prove it for k, $k \leqq \lfloor \log n \rfloor$.

Given a node x, by the inductive hypothesis after the first $(k-1)$ time units of the second step there exist k edge disjoint paths, $P_0, \cdots, P_{k-1}$, from the originator to x. Write them as $P_i = R_i Q_i$ and let

$$I(P_i) \in A_{r_i} \quad \text{for } i=0, \cdots, k-1.$$

Moreover, let $P'_0 = R'_0 Q'_0$, $P'_1 = R'_1 Q'_1$, $\cdots$, be all the $(k-1)$-paths to $f_k(x)$.

We show that there exists a j such that $P_0, \cdots, P_{k-1}, P'_j(f_k(x), x)$ are edge disjoint. The proof is by contradiction. Therefore we suppose that for each j the paths $P_0, \cdots, P_{k-1}, P'_j(f_k(x), x)$ are not edge disjoint. By Lemma 1, this implies that for each j there exists an i, $0 \leqq i \leqq k-1$, such that $I(P'_j) \in A_{r_i}$. Since by the inductive hypothesis k of the paths P' from 0 to $f_k(x)$ are edge-disjoint, by Lemma 1 we obtain that for each $r = 1, \cdots, \lceil \log n \rceil$

(9) there exists an i such that $I(P_i) \in A_r \Leftrightarrow$ there exists a j such that $I(P'_j) \in A_r$.

We distinguish two cases.

Case 1. There is at least one index r_i such that $r_i \geqq k+1$, for some $0 \leqq i \leqq k-1$. Using (1) and (2), we get $|A_{r_i}| \leqq |A_{k+1}| \leqq \lceil n/2^{k+1} \rceil$. From (9), it follows that there exist two informers of x at time k, namely $I(P_i)$ and some $I(P'_j) \in \inf(f_k(x), k-1) \subset \inf(x, k)$, which both belong to A_{r_i}. Therefore denoting the absolute value with $\|\cdot\|$, we get

$$\|I(P_i)-I(P'_j)\| \leqq |A_{r_i}| - 1 \leqq \lceil n/2^{k+1} \rceil - 1 < \lceil n/2^k \rceil - 1,$$

contradicting Property 2 which states $\|I(P_i) - I(P'_j)\| \geqq \lceil n/2^k \rceil - 1$.

Case 2. All indices r_i, for $0 \leqq i \leqq k-1$, are less than or equal to k. Consider the following two subcases.

(a) $\inf(x, k-1) \subseteq \bigcup_{i=1}^{k} A_i$.

Relation (9) implies that also $\inf(f_k(x), k-1) \subseteq \bigcup_{i=1}^{k} A_i$. In fact, by definition of informer, for each $I \in \inf(f_k(x), k-1)$ we can find a $(k-1)$-path P' to $f_k(x)$ with $I = I(P')$. Therefore we get that $\inf(x, k) \cap (V - \bigcup_{i=1}^{k} A_i) = \varnothing$. It follows that there exist $n - \sum_{i=1}^{k} |A_i|$ consecutive numbered nodes, none of which is an informer of x at time k. From this we get that writing $\inf(x, k)$

as $\{a_1, \cdots, a_{2^k}\}$ with $a_1 < \cdots < a_{2^k}$ either $n + (a_1 - a_{2^k})$ or $a_l - a_{l+1}$, for some l, assumes a value not less than $n - \sum_{i=1}^{k} |A_i| + 1$. Using (3) we get that $n - \sum_{i=1}^{k} |A_i| + 1 \geqq \lceil n/2^k \rceil + 1$, contradicting Property 2.

(b) There exists a node I such that $I \in \inf(x, k-1) \cap A_r$ for some $r > k$.

Let $P = RQ$ be the $(k-1)$-path to x such that $I(P) = I$. First we show that there exist $k-1$ paths among $P_0 = R_0Q_0, \cdots, P_{k-1} = R_{k-1}Q_{k-1}$ which are edge disjoint with P. Since $I(P_i) \in A_{r_i} \neq A_r$ for $i = 0, \cdots, k-1$, the paths R and R_i, formed by calls of the first step, are edge disjoint. If the path Q shares an edge with at most one path Q_i, for some $i \leqq k-1$, the above assertion is trivially true. Suppose on the contrary that Q shares edges with two paths Q_i and Q_j, for some $i, j \leqq k-1$. There must exist r, s, with $1 \leqq r < s \leqq k-1$ such that

$$Q = A(a, f_r(a))B(b, f_s(b))C$$

and

$$Q_i = A_i(a, f_r(a))B_i \quad \text{or} \quad Q_i = A_i(f_r(a), a)B_i, \quad \text{and}$$

$$Q_j = B_j(b, f_s(b))C_j \quad \text{or} \quad Q_j = B_j(f_s(b), b)C_j$$

with $(b, f_s(b)), (f_s(b), b) \notin B_i$, since Q_i and Q_j are edge disjoint. As illustrated in Fig. 3, it follows that from the node a there are two different paths to x, which implies that for some time instant t, with $s \leqq t \leqq k-1$, a appears twice as informer of x at time t. Hence for some time unit $t < k$ and node y we have $a \in \inf(y, t) \cap \inf(f_{t+1}(y), t)$ contradicting Property 3. Therefore we have that for some index i, $0 \leqq i \leqq k-1$, the paths $P_0, \cdots, P_{i-1}, P_{i+1}, \cdots, P_{k-1}$, P are edge disjoint. On the other hand, $I(P) \in A_r$, with $r \geqq k+1$; i.e., the paths $P_0, \cdots, P_{i-1}, P_{i+1}, \cdots, P_{k-1}$, P satisfy Case 1, and we can again derive a contradiction (if we suppose that none of the paths $P'_j(f_k(x), x)$ is edge-disjoint from each of the paths $P_0, \cdots, P_{i-1}, P_{i+1}, \cdots, P_{k-1}, P$).

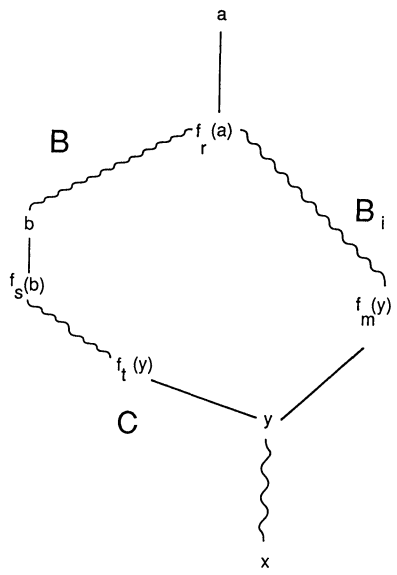


FIG. 3. *Illustration of Case* 2(b) *of Lemma* 1.

Therefore there exist $k+1$ edge-disjoint paths from the originator to x at time k, and the theorem is proved. □

3. Fault-tolerant broadcast—an odd number of nodes. Let the number n of nodes in the network be odd. In this section we prove that it is always possible to perform a k fault-tolerant broadcast in $\lceil \log n \rceil + k + 1$ time units, for each $k \leqq \lfloor \log (n-1) \rfloor$. We show that if k is greater than a certain value, depending on n, it is not possible to perform a broadcast in time less than $\lceil \log n \rceil + k + 1$. Finally, we show that for some $k \leqq \lfloor \log (n-1) \rfloor$ it is possible to perform a k fault-tolerant broadcast in optimal time $\lceil \log n \rceil + k$.

First we give an upper bound that holds for all $k \leqq \lfloor \log (n-1) \rfloor$.

THEOREM 2. *For each odd integer n, it is possible to perform k fault-tolerant broadcast in time $\lceil \log (n-1) \rceil + k + 1$ for each $k \leqq \lfloor \log (n-1) \rfloor$.*

Proof. Let the node set be $\{0, \cdots, n-1\}$ with n odd, and the originator be the node $n-1$. The idea is to apply the broadcast scheme of § 2 to the even-sized set of nodes $\{0, \cdots, n-2\}$ with the following modifications: In the first step of the broadcast the calls that should be made by 0 are now performed by $n-1$, which will also call 0 at time $\lceil \log (n-1) \rceil + 1$; the second step consists of calls exchanged for k consecutive time units among nodes in $\{0, \cdots, n-2\}$ according to rule (4). Note that (4) can be consistently applied since the size of $\{0, \cdots, n-2\}$ is even.

More formally, partition the set $\{0, \cdots, n-2\}$ into sets $A_1, \cdots, A_{\lceil \log (n-1) \rceil + 1}$ as described in (1); that is,

$$A_t = \left\{ \left\lceil \frac{(n-1)}{2^t} \right\rceil, \left\lceil \frac{(n-1)}{2^t} \right\rceil + 1, \cdots, \left\lceil \frac{(n-1)}{2^{t-1}} \right\rceil - 1 \right\},$$

$$t = 1, \cdots, \lceil \log (n-1) \rceil, \quad A_{\lceil \log (n-1) \rceil + 1} = \{0\}.$$

Consider the following scheme:

First step. Time unit 1. The node $n-1$ calls the node $n-2 \in A_1$ that, during the following $\lceil \log (n-1) \rceil - 1$ time units, broadcasts to A_1;

Time unit t, $(2 \leqq t \leqq \lceil \log (n-1) \rceil)$. The node $n-1$ calls the node $\lceil (n-1)/2^t \rceil \in A_t$ that, during the following $\lceil \log (n-1) \rceil - t$ time units broadcasts to A_t;

Time unit $\lceil \log (n-1) \rceil + 1$. The node $n-1$ calls the node 0.

Second step. The nodes numbered from 0 to $n-2$ exchange calls for k consecutive time units according to the previous rule (4); that is, at time unit t, $1 \leqq t \leqq k$, each node $x \in \{0, \cdots, n-2\}$ calls node

$$f_t^{(n-1)}(x) = \begin{cases} \dfrac{n-1}{2} + \left(x \oplus \left\lceil \dfrac{n-1}{2^t} \right\rceil \right) & \text{if } x < (n-1)/2 \\ \left(x \ominus \left\lceil \dfrac{n-1}{2^t} \right\rceil \right) & \text{if } x \geqq (n-1)/2, \end{cases}$$

where $\ominus$ and $\oplus$ denote, respectively, the subtraction and the addition modulo $(n-1)/2$; the originator does not make any call in the second step.

The calls of the $\lceil \log (n-1) \rceil + 1$ time units of the first step followed by the calls of the first k time units of the second step produce a k fault-tolerant broadcast. This follows from Theorem 1 applied to nodes $0, \cdots, n-2$. In fact, let us first note that the originator $n-1$ does not need to be called. Moreover, for each node $x \neq n-1$ the following two relations hold:

(i) whenever the k fault-tolerant broadcast from 0 to $\{0, \cdots, n-2\}$ (defined in § 2) creates a path to x of the type $P = (0, \lceil (n-1)/2^t \rceil)P'$, $1 \leqq t \leqq \lceil \log (n-1) \rceil$ the above broadcast from $n-1$ to $\{0, \cdots, n-1\}$ creates the path $(n-1, \lceil (n-1)/2^t \rceil)P'$;

(ii) if the k fault-tolerant broadcast from 0 to $\{0, \cdots, n-2\}$ (defined in § 2) creates the path to x of the type $Q = (0, f_s^{(n-1)}(0))Q'$, for some $1 \leqq s \leqq k$, the above broadcast from $n-1$ to $\{0, \cdots, n-1\}$ creates the path $(n-1, 0)(0, f_s^{(n-1)}(0))Q'$. Since Theorem 1 tells us that there exist $k+1$ edge disjoint paths from 0 to x—of which at most one can be of the type in (ii); otherwise, 0 appears twice as an originator of x contradicting Property 3—it is easy to see that also in this case there are $k+1$ edge disjoint paths from $n-1$ to x. Noting that in Theorem 1 we do not distinguish node 0, that is, 0 also receives the message along a new disjoint path at each time unit of the second step, the theorem is proved. □

Example 3. Consider $n = 11$ and $k = 2$. Fig. 4 shows the calling tree resulting from applying the above scheme. In the figure only the three edge disjoint paths to each node are represented.

Note that in case $n = 2^{\lceil \log n \rceil} - 1$, it holds that $\lceil \log (n-1) \rceil + k + 1 = \lceil \log n \rceil + k$.

The following result is a generalization of the lower bound for $k = 2$ presented in [12]. An equivalent result is also reported in [15].

THEOREM 3. *Let n be an odd integer. For each $k > 2^{\lceil \log n \rceil} - n - 1 + \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor$ it is not possible to perform k fault-tolerant broadcast to n nodes in less than $\lceil \log n \rceil + k + 1$ time units.*

Theorem 2 gives an upper bound on the k fault-tolerant broadcast time for all $k \leqq \lfloor \log (n-1) \rfloor$. We now show that for some small k it is possible to perform a k fault-tolerant broadcast protocol in time $\lceil \log n \rceil + k$.

THEOREM 4. *Let n be an odd integer. For each $k \leqq \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor$ it is possible to perform k fault-tolerant broadcast to n nodes in $\lceil \log n \rceil + k$ time units.*

Proof. Again we modify the method used in case of an even number of members. Suppose that the originator is the node $n-1$. Let k be an integer such that

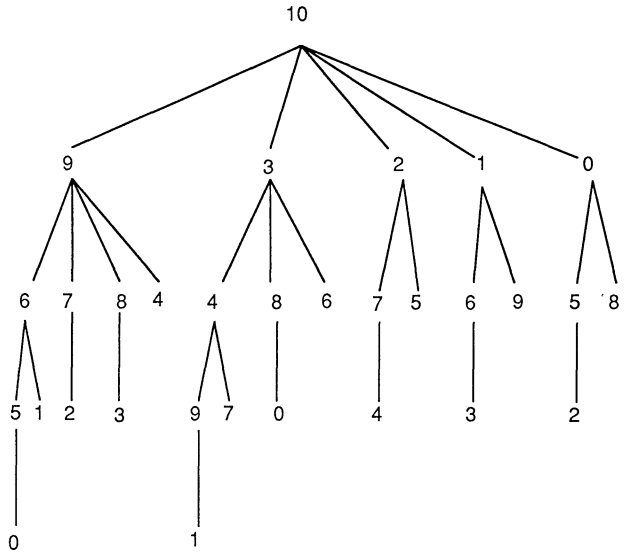


FIG. 4. 2 *fault-tolerant broadcast according to Theorem* 2 *for* $n = 11$ *nodes.*

$\lceil (n-1)/2^k \rceil + 1 \leqq 2^{\lceil \log n \rceil - k}$. Consider the partition of $\{0, \cdots, n-2\}$ into sets $A_1, \cdots, A_{\lceil \log (n-1) \rceil + 1}$ as defined in (1), that is,

$$A_t = \left\{ \left\lceil \frac{(n-1)}{2^t} \right\rceil, \left\lceil \frac{(n-1)}{2^t} \right\rceil + 1, \cdots, \left\lceil \frac{(n-1)}{2^{t-1}} \right\rceil - 1 \right\},$$

$$t = 1, \cdots, \lceil \log (n-1) \rceil, \quad A_{\lceil \log (n-1) \rceil + 1} = \{0\},$$

and partition the set $\{0, \cdots, n-1\}$ into $k+1$ subsets $C_1, \cdots, C_{k+1}$, with $C_i = A_i$, for $i = 1, \cdots, k$, and $C_{k+1} = \cup_{i=k+1}^{\lceil \log (n-1) \rceil + 1} A_i \cup \{n-1\} = \{0, \cdots, n-1\} - \cup_{i=1}^{k} A_i$. Note that by (3) and the definition of k, we have $|C_{k+1}| = \lceil (n-1)/2^k \rceil + 1 \leqq 2^{\lceil \log n \rceil - k}$.

Consider the following calling scheme:

First Step. Time unit 1. The originator $n-1$ calls the node $n-2 \in C_1 = A_1$ that, during the following $\lceil \log (n-1) \rceil$ time units, broadcasts to C_1;

Time unit $t \leqq k$. $n-1$ calls the node $\lceil (n-1)/2^t \rceil \in C_t = A_t$ that, during the following $\lceil \log (n-1) \rceil - t$ time units, broadcasts to C_t;

Time unit $k+1$. $n-1$ begins the broadcast to the $\lceil (n-1)/2^k + 1 \rceil \leqq 2^{\lceil \log n \rceil - k}$ nodes in the set C_{k+1}; the broadcast will be completed within time $\lceil \log n \rceil$.

Second step. The nodes numbered from 0 to $n-2$ exchange calls for k consecutive time units according to rule (4); that is, at time unit t, $1 \leqq t \leqq k$, each node $x \in \{0, \cdots, n-2\}$ calls node $f_t^{(n-1)}(x)$; the originator $n-1$ is inactive.

Since relations (2) and (3) hold on nodes $0, \cdots, n-2$ for each time unit less or equal to k, Properties 1–3 hold. The proofs of Lemma 1 and Theorem 1 can be repeated to show that the calls of the $\lceil \log n \rceil$ time units of the first step followed by the ones of the first k time units of the second step produce a k fault-tolerant broadcast (note that for each time unit $t \leqq k$ the node set $C_t = A_t$ of the subtree, rooted in the originator, formed by the calls of the first step from time instant t to time instant k is the same as that considered in Theorem 1).

It remains to find the maximum value of the integer k for which the above scheme works, that is, the maximum k satisfying $\lceil (n-1)/2^k \rceil + 1 \leqq 2^{\lceil \log n \rceil - k}$. Solving the above

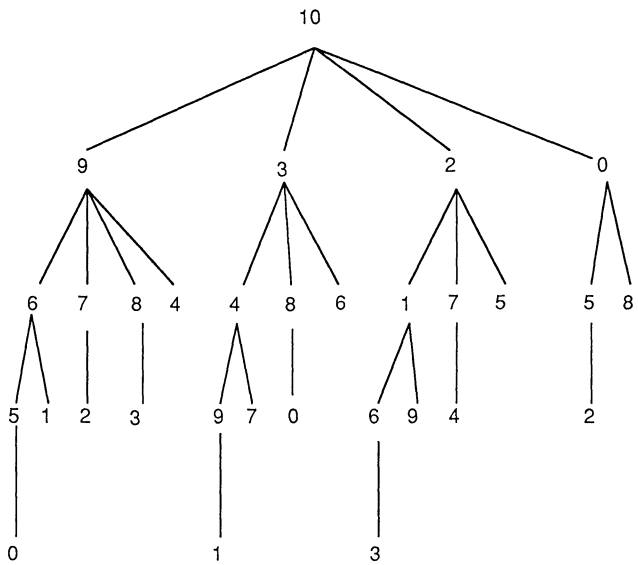


FIG. 5. 2 *fault-tolerant broadcast according to Theorem* 4 *for* $n = 11$ *nodes*.

inequality we obtain that the given scheme allows a k fault-tolerant broadcast in time $\lceil \log n \rceil + k$ if $k \leqq \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor$. □

Example 4. Consider $n = 11$ and $k = 2$. We have $C_1 = A_1 = \{6, 7, 8\}$, $C_2 = A_2 = \{3, 4\}$, $C_3 = \{0, 1, 2, 10\}$. The calls of the first step and those of the first two time units of the second step are represented in Fig. 5.

4. On the number of edges of fault-tolerant broadcast networks. We have assumed to this point that the network is a complete graph, with nodes labeled from 0 to $n - 1$ and that the originator of the broadcast is node 0. In this section we study the number of edges needed to construct sparse fault-tolerant broadcast networks supporting our broadcasting scheme, when *any* node can be the originator.

We first derive a lower bound on the degree of each node in a network with k fault-tolerant broadcast time $\lceil \log n \rceil + k$. Consider the originator of the broadcast. Since the first k calls from the originator can use faulty edges, the calls from time $k + 1$ to $k + \lceil \log n \rceil$ must give a broadcast. Let a_t be the node the originator calls at time $k + t$. Within time $\lceil \log n \rceil + k$ at most $2^{\lceil \log n \rceil - t}$ nodes can be reached from a_t. Therefore if the originator calls r times after time unit k it must hold that

$$\sum_{t=1}^{r} 2^{\lceil \log n \rceil - t} = 2^{\lceil \log n \rceil} - 2^{\lceil \log n \rceil - r} \geqq n - 1.$$

From this we get that the degree of the originator cannot be less than $k + \lceil \log n \rceil - \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor$. This gives a lower bound of

$$\left\lceil \frac{n}{2} (k + \lceil \log n \rceil - \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor) \right\rceil \tag{10}$$

on the number of edges in a network with k fault-tolerant broadcast time equal to $\lceil \log n \rceil + k$.

We now consider the number of edges needed to construct a network supporting the broadcast scheme introduced in § 2. Moreover, we require that any node could act as originator.

If the originator is a node $\alpha \neq 0$, we can consider for the first step a different partition of the node set depending on α. Let us first consider the case n even. Consider the sets $A_i(\alpha)$ obtained in the following way: $A_0(\alpha) = V = \{0, \cdots, n - 1\}$; for each $t \leqq \lceil \log n \rceil$, if $V - \bigcup_{i=1}^{t-1} A_i(\alpha) = \{a, a + 1, \cdots, a + b - 1\}$ we define

$$A_t(\alpha) = \begin{cases} \{a + \lceil b/2 \rceil, \cdots, a + b - 1\} & \text{if } \alpha < a + \lceil b/2 \rceil, \\ \{a, \cdots, a + \lfloor b/2 \rfloor - 1\} & \text{if } \alpha \geqq a + \lceil b/2 \rceil; \end{cases} \tag{11}$$

$$A_{\lceil \log n \rceil + 1}(\alpha) = \{\alpha\}.$$

Consider then the following broadcast scheme.

First step. Time unit 1. If $\alpha < n/2$ then α calls $(n/2) + (\alpha \ominus 1)$, while if $\alpha \geqq n/2$ then α calls $(\alpha \oplus 1)$. In the following time units the newly informed node broadcasts to $A_1(\alpha)$.

Time unit t, $1 < t \leqq \lceil \log n \rceil$. α calls a node in the set $A_t(\alpha)$ that will broadcast to $A_t(\alpha)$ while α continues broadcasting to $V - A_1(\alpha) - \cdots - A_t(\alpha)$.

Any other informed node y calls with a similar rule, that is, let $\{c, \cdots, c + d - 1\}$ the set to which y has to broadcast: If $y \geqq c + \lceil d/2 \rceil$ then y calls a node in the set $\{c, \cdots, c + \lfloor d/2 \rfloor - 1\}$ that will broadcast to this set and y continues broadcasting to the remaining nodes; if $y < c + \lceil d/2 \rceil$ then y calls a node in the set $\{c + \lceil d/2 \rceil, \cdots, c + d - 1\}$ that will broadcast to this set and y continues broadcasting to the other nodes.

Second step. At time unit t, $1 \leqq t \leqq k$, each node $x \in \{0, \cdots, n-1\}$ calls node $f_t^{(n)}(x)$ as defined in (4).

Example 5. For $n = 10$, originator $\alpha = 3$ and $k = 2$ the calling tree is shown in Fig. 6.

In the first step of the above scheme each informed node partitions the subset to which it has to broadcast into two subsets whose sizes differ at most by one, and informs a node in the subset it does not belong to. Therefore after each time unit t, each informed node has to broadcast to at most $\lceil n/2^t \rceil \leqq 2^{\lceil \log n \rceil - t}$ nodes. Hence (in absence of faults) any node is informed within the $\lceil \log n \rceil$ time units of the first step.

In order to prove that broadcast scheme is k fault-tolerant for any $k \leqq \lfloor \log n \rfloor$, we make the following remarks:

R1. For each α and t the sets $V - A_1(\alpha) - \cdots - A_{t-1}(\alpha)$ and $A_t(\alpha) \subseteq V - A_1(\alpha) - \cdots - A_{t-1}(\alpha)$ are formed by consecutive numbered nodes in V (as in the special case $\alpha = 0$).

R2. For each α and t

$$|A_t(\alpha)| = \left\lfloor \frac{n - \sum_{i=1}^{t-1} |A_i(\alpha)|}{2} \right\rfloor = |A_t(0)| = |A_t|, \qquad (A_t \text{ as defined in (1)}),$$

and therefore we obtain the analogous of relations (2) and (3); that is,

$$|A_t(\alpha)| \in \{\lceil n/2^t \rceil, \lceil n/2^t \rceil - 1\} \quad \text{and} \quad \lceil n/2^t \rceil = n - \sum_{i=1}^{t} |A_i(\alpha)|.$$

R3. Since, from R2, $|A_t(\alpha)| = |A_t|$ and the definition of the informers does not depend on the particular originator α, Properties 1–3 hold.

R4. Lemma 1 holds for any originator α, if we substitute A_t with $A_t(\alpha)$. Therefore we can use Properties 1–3, Lemma 1, and R1 and R2 to repeat the proof of Theorem 1 for any originator α.

We now construct a graph that supports the first step of the above described broadcast scheme where each node can be the originator. Given two integers a and b, define the

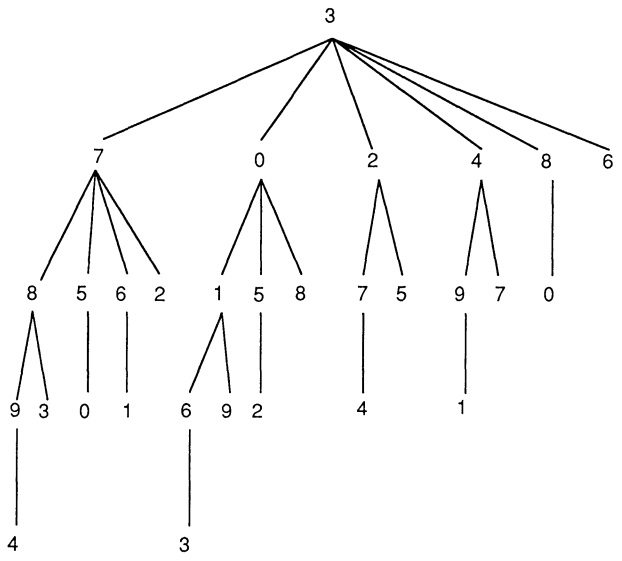


FIG. 6. 2 *fault-tolerant broadcast tree for* $n = 10$ *nodes when the originator is the node* 3.

graph $(\{a, a+1, \cdots, a+b-1\}, E(a, b))$ with node set $\{a, a+1, \cdots, a+b-1\}$ (where a is the smallest node and b represents the size of the node set), and edge set $E(a, b)$ obtained in the following way:

$$E(a,b)=\begin{cases}\varnothing & \text{if } b=1\\ \left\{\left(i,\frac{b}{2}+i\right)\middle| i=a,\cdots,a+\frac{b}{2}-1\right\}\cup\left\{\left(a+\frac{b}{2}-1,a+\frac{b}{2}\right)\right\} & \\ \quad \cup E\left(a,\frac{b}{2}\right)\cup E\left(a+\frac{b}{2},\frac{b}{2}\right) & \text{if } b>1 \text{ is even}\\ \left\{\left(i,\left\lceil\frac{b}{2}\right\rceil+i\right)\middle| i=a,\cdots,a+\left\lfloor\frac{b}{2}\right\rfloor-1\right\} & \\ \quad \cup\left\{\left(a+\left\lfloor\frac{b}{2}\right\rfloor\pm 1,a+\left\lfloor\frac{b}{2}\right\rfloor\right)\right\}\cup E\left(a,\left\lceil\frac{b}{2}\right\rceil\right) & \\ \quad \cup E\left(a+\left\lfloor\frac{b}{2}\right\rfloor,\left\lceil\frac{b}{2}\right\rceil\right) & \text{if } b>1 \text{ is odd.}\end{cases}$$

Let $E_n = E(0, n/2) \cup E(n/2, n/2) \cup \{(i, n/2 + i \ominus 1) \mid 0 \leqq i < n/2\}$. Finally, define the graph $G_n = (\{0, \cdots, n-1\}, E_n)$.

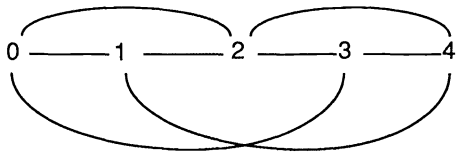
LEMMA 2. *The graph $G_n = (\{0, \cdots, n-1\}, E_n)$ supports the first step of the broadcast scheme from any originator, according to the above-described method.*

Proof. The proof is given in Appendix 2. □

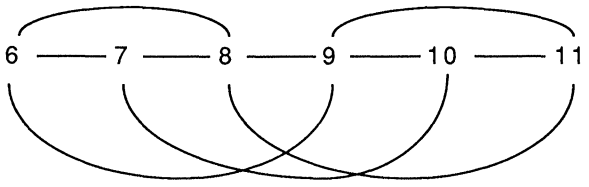
Example 6. The graphs with edge sets $E(0, 5)$ and $E(6, 6)$ are shown in Fig. 7.

We then have the following result.

THEOREM 5. *For each n even and $k \leqq \lfloor \log n \rfloor$, the graph $G_{n,k} = (V, E_{n,k})$, with $V = \{0, \cdots, n-1\}$ and $E_{n,k} = E_n \cup \{(x, f_t^{(n)}(x)) \mid 0 \leqq x \leqq n-1 \text{ and } 1 \leqq t \leqq k\}$,*



(a)



(b)

FIG. 7. (a) *Graph* $(\{0, \cdots, 5\}, E(0, 5))$; (b) *Graph* $(\{6, \cdots, 11\}, E(6, 6))$.

has k fault-tolerant broadcast time equal to $\lceil \log n \rceil + k$ *and number of edges upper bounded by* $\frac{n}{2}(\lceil \log n \rceil + k + 5) - \lceil \log n \rceil - 8$.

Proof. By Lemma 2, the edges in the set E_n support the calls of the first step of the broadcast scheme. Noting that the calls of the second step use the same set of $kn/2$ edges $\{(x, f_t^{(n)}(x)) | 0 \leqq x \leqq n-1 \text{ and } 1 \leqq t \leqq k\}$, regardless of the originator, we get the first part of the theorem. We now give an upper bound on the number of edges used for the first step, that is, on the size of the set E_n. Let $e(b)$ be the number of edges in the graph $(\{a, a+1, \cdots, a+b-1\}, E(a, b))$ (it is independent of the actual value of node a). By definition, $e(b) = \lceil b/2 \rceil + 1 + 2e(\lceil b/2 \rceil)$, with $e(2) = 1$. Solving the above recurrence relation we get

$$e(b) = \sum_{i=0}^{\lceil \log b \rceil - 2} 2^i \left(\left\lceil \frac{b}{2^{i+1}} \right\rceil + 1 \right) + 2^{\lceil \log b \rceil - 1}.$$

Since $2^i \lceil b/2^{i+1} \rceil \leqq b/2 + 2^i - \frac{1}{2}$, we obtain

$$e(b) \leqq \sum_{i=0}^{\lceil \log b \rceil - 2} \left(\frac{b}{2} + 2^{i+1} - \frac{1}{2} \right) + 2^{\lceil \log b \rceil - 1} = \frac{b-1}{2}(\lceil \log b \rceil - 1) + 3(2^{\lceil \log b \rceil - 1}) - 2.$$

Therefore from the definition of E_n, we get

$$|E_n| = 2e(n/2) + \left| \left\{ \left(i, \frac{n}{2} + i \ominus 1 \right) \middle| 0 \leqq i < \frac{n}{2} \right\} \right|$$

$$\leqq 2\left(\frac{n-2}{4}(\lceil \log n \rceil - 2) + 3(2^{\lceil \log n \rceil - 2}) - 2 \right) + \frac{n}{2}.$$

Noting that $2^{\lceil \log n \rceil - 2} \leqq n/2 - 1$ we get

$$|E_n| \leqq \frac{n}{2}(\lceil \log n \rceil + 5) - \lceil \log n \rceil - 8.$$

Since $|E_{n,k}| = |E_n| + kn/2$, we obtain the desired bound on $E_{n,k}$. □

Note that the upper bound given in Theorem 5 on the number of edges of the $\lfloor \log n \rfloor$ fault-tolerant broadcast network supporting our broadcast scheme is of the same order of magnitude as the lower bound given in (10). Also, note that if n is a power of 2, $n = 2^m$, the first step can be performed on a graph having the structure of two $(m-1)$-dimensional cubes connected by the one-to-one matching $\{(i, n/2 + i \ominus 1) \mid 0 \leqq i < n/2\}$. In this case the scheme of the first step is: At time 1 the originator calls its neighbor in the other $(m-1)$-cube; at time t each informed vertex calls its neighbor along the tth dimension, for some ordering of the $m-1$ dimensions in the cubes. From this and from (10), one gets the following result.

COROLLARY 1. *For* $n = 2^m$ *and for each* $k \leqq m$, *the graph that supports the above-described k fault-tolerant broadcast scheme has the minimum possible number of edges among all k fault-tolerant broadcast networks.*

Let us assume now that the number of network members is odd. Consider the scheme introduced in Theorem 2 and a graph on $\{0, \cdots, n-1\}$ whose edges are $(n-1, i)$, $i = 0, \cdots, n-2$, plus those in the graph $G_{n-1,k}$ obtained for the nodes in the set $\{0, \cdots, n-2\}$. If the originator is $n-1$ the scheme is described in Theorem 2. In case the originator is a node $\alpha < n-1$, it is easy to see that such a scheme can be modified as follows:

First step. The calls of the first step of the broadcast follow the same scheme as in case of the broadcast from α to $\{0, \cdots, n-2\}$, during the first $\lceil \log (n-1) \rceil$ time units. At time $\lceil \log (n-1) \rceil + 1$ the originator α calls $n-1$.

Second step. At time unit t, $1 \leqq t \leqq k$, each node $x \in \{0, \cdots, n-2\}$ calls node $f_t^{(n-1)}(x)$ as defined in (4) except that the originator α (which is inactive) is replaced by $n-1$, that is, at each time unit t the node $f_t^{(n-1)}(\alpha)$ calls $n-1$ and $n-1$ calls $f_t^{(n-1)}(\alpha)$.

The number of edges needed to complete the above scheme from any node are not more than

$$|E_{n-1,k}| + |\{(i,n-1) | i=0, \cdots, n-2\}|$$

$$\leqq \frac{n-1}{2}(\lceil \log(n-1)\rceil + k + 5) - \lceil \log(n-1)\rceil - 8 + n - 1$$

$$= \frac{n}{2}(\lceil \log(n-1)\rceil + k + 7) - \frac{1}{2}(3\lceil \log(n-1)\rceil + k + 23).$$

Finally, consider the scheme described in Theorem 4. If the originator is $n-1$, the broadcast scheme is the one in Theorem 4. Let $k = \lfloor \log(2^{\lceil \log n\rceil} - n + 1)\rfloor$. If the originator is a node $\alpha < n-1$, partition the set $\{0, \cdots, n-1\}$ into subsets $C_1(\alpha), \cdots, C_{\lceil \log n\rceil + 1}(\alpha)$, where $C_t(\alpha) = A_t(\alpha)$, for $t = 1, \cdots, k$, is obtained by applying (11) to $A_0(\alpha) = \{0, \cdots, n-2\}$, while $C_t(\alpha)$, for $t \geqq k+1$, are defined as in (11) but considering also node $n-1$; that is, if $\{0, \cdots, n-1\} - \cup_{j=1}^{t-1} C_j(\alpha) = \{a, a+1, \cdots, a+b-2\} \cup \{n-1\}$, define

$$C_t(\alpha) = \begin{cases} \left\{a + \left\lceil \frac{b}{2} \right\rceil, \cdots, a+b-2\right\} \cup \{n-1\} & \text{if } i < a + \left\lceil \frac{b}{2} \right\rceil, \\ \left\{a, \cdots, a + \left\lfloor \frac{b}{2} \right\rfloor - 1\right\} & \text{if } i \geqq a + \left\lceil \frac{b}{2} \right\rceil; \end{cases}$$

$$C_{\lceil \log n\rceil + 1}(\alpha) = \{\alpha\}.$$

The broadcast scheme is the following.

First step. At each time unit t, $1 \leqq t \leqq \lceil \log n\rceil$, α calls a node in $C_t(\alpha)$ that, during the following $\lceil \log n\rceil - t$ time units, broadcasts to $C_t(\alpha)$.

Second step. At time unit t, $1 \leqq t \leqq k$, each node $x \in \{0, \cdots, n-2\}$ calls node $f_t^{(n-1)}(x)$ as defined in (4) except that if the originator is $\alpha \neq n-1$, node α (which is inactive) is replaced by $n-1$, that is, at each time unit t the node $f_t^{(n-1)}(\alpha)$ calls $n-1$ and $n-1$ calls $f_t^{(n-1)}(\alpha)$.

Example 7. Consider $n = 11$, $\alpha = 6$ and $k = \lfloor \log 6\rfloor = 2$. One has

$$C_1(6) = A_1(6) = \{0,1,2,3,4\}, \quad C_2(6) = A_2(6) = \{8,9\},$$

$$C_3(6) = \{7,10\}, \quad C_4(6) = \{5\}.$$

The calls of the first step and of the two time units of the second step are represented in Fig. 8.

Consider the graph with node set $\{0, \cdots, n-1\}$ and edge set $E_{n-1} \cup \{(j, n-1) | j = 0, \cdots, n-2\}$. Following the lines of Lemma 2 and considering the fact that node $n-1$ is connected to any other node, it is possible to show that this graph allows the first step from any originator. Therefore the following result holds.

THEOREM 6. *For any odd n and $k \leqq \lfloor \log(n-1)\rfloor$ and k fault-tolerant broadcast network $G'_{n,k} = (\{0, \cdots, n-1\}, E'_{n,k})$ with node set $\{0, \cdots, n-1\}$ and edge set $E'_{n,k} = E_{n-1} \cup \{(i, n-1) | i = 0, \cdots, n-2\})$ satisfies*

$$|E'_{n,k}| \leqq \frac{n}{2}(\lceil \log(n-1)\rceil + k + 7) - \frac{1}{2}(3\lceil \log(n-1)\rceil + k + 23)$$

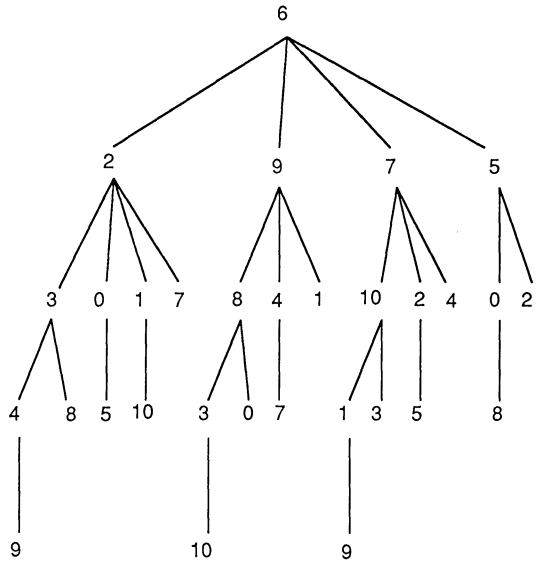


FIG. 8. 2 *fault-tolerant broadcast tree for* $n = 11$ *nodes when the originator is the node* 6.

and has k fault-tolerant broadcast time not greater than $\lceil \log n \rceil + k$ *if* $k \leqq \lfloor \log 2^{\lceil \log n \rceil} - n + 1 \rfloor$, $\lceil \log n \rceil + k + 1$ *otherwise*.

5. Conclusions and open problems. We have given a method to construct minimum time, or almost minimum time, k fault-tolerant broadcast networks, for each value of $k \leqq \lfloor \log n \rfloor$, n being the number of processors in the network. The broadcast primitive seems to be important in communication networks and parallel computing. Our construction shows that it is possible to obtain $\lfloor \log n \rfloor$ fault-tolerant broadcast networks with optimal broadcast time and optimal order of magnitude of edges.

Several open problems remain in the area. We list the most important of them here.

(1) We have shown that $T_k(n) \in \{\lceil \log n \rceil + k, \lceil \log n \rceil + k + 1\}$ for n odd and k such that $\lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor < k < \min \{2^{\lceil \log n \rceil} - n - 1 + \lfloor \log (2^{\lceil \log n \rceil} - n + 1) \rfloor, \lfloor \log (n - 1) \rfloor\}$. What is the true value of $T_k(n)$ in this range?

(2) Can the upper bounds on the number of edges given in Theorems 5 and 6 be improved? For $k = 1$ and 2, better upper bounds are obtained by using ad hoc designs [4].

(3) We have shown that if one asks that the broadcast time from each node takes $\lceil \log n \rceil + k$ time units the degree of the originator cannot be less than $k + \lceil \log n \rceil - \lfloor \log 2^{\lceil \log n \rceil} - n + 1 \rfloor$. Since it must hold $k + \lceil \log n \rceil - \lfloor \log 2^{\lceil \log n \rceil} - n + 1 \rfloor \leqq n - 1$, it follows $k \leqq n - \lceil \log n \rceil - 1 - \lfloor \log 2^{\lceil \log n \rceil} - n + 1 \rfloor$. If n is odd one has also $k \leqq 2^{\lceil \log n \rceil} - n - 1 - \lfloor \log 2^{\lceil \log n \rceil} - n + 1 \rfloor$. The best known result for $k > \lceil \log n \rceil$ shows that the broadcast can be done in $O(\log n + k)$ [13], [15]. Then a question naturally arises: Is it possible to perform a k fault-tolerant broadcast protocol on n nodes in time $\lceil \log n \rceil + k$ for each k satisfying the above inequalities?

Acknowledgments. The authors wish to express their sincere gratitude to Moti Young and the anonymous referees whose advice has greatly improved the presentation of the paper.

Appendix 1. Proof of Property 1. Let $x < n/2$. From the definition of $\inf(x, t)$, we have that $a \in \inf(x, t) - \{x\}$ if and only if there exists $1 \leqq i_1 < \cdots < i_r \leqq t$ such

that $x = f_{i_r}(f_{i_{r-1}}(\cdots f_{i_1}(a)))$, or equivalently from (5), $a = f_{i_1}(f_{i_2}(\cdots f_{i_r}(x)))$. Applying the definition of $f_i(x)$ given in (4), we get

$$\begin{aligned} a &= f_{i_1}(f_{i_2}(\cdots f_{i_{r-1}}(n/2 + x \oplus \lceil n/2^{i_r} \rceil))) \\ &= f_{i_1}(f_{i_2} \cdots f_{i_{r-2}}(x \oplus \lceil n/2^{i_r} \rceil \ominus \lceil n/2^{i_{r-1}} \rceil)) \\ &= f_{i_1}(f_{i_2} \cdots f_{i_{r-2}}(x \oplus (\lceil n/2^{i_r} \rceil - \lceil n/2^{i_{r-1}} \rceil))). \end{aligned}$$

By iterating the reasoning, we obtain

$$a = \begin{cases} x \oplus \left(\left\lceil \dfrac{n}{2^{i_r}} \right\rceil - \left\lceil \dfrac{n}{2^{i_{r-1}}} \right\rceil + \left\lceil \dfrac{n}{2^{i_{r-2}}} \right\rceil - \cdots + \left\lceil \dfrac{n}{2^{i_2}} \right\rceil - \left\lceil \dfrac{n}{2^{i_1}} \right\rceil \right) & \text{if } r \text{ is even,} \\ \dfrac{n}{2} + x \oplus \left(\left\lceil \dfrac{n}{2^{i_r}} \right\rceil - \left\lceil \dfrac{n}{2^{i_{r-1}}} \right\rceil + \left\lceil \dfrac{n}{2^{i_{r-2}}} \right\rceil - \cdots - \left\lceil \dfrac{n}{2^{i_2}} \right\rceil + \left\lceil \dfrac{n}{2^{i_1}} \right\rceil \right) & \text{if } r \text{ is odd.} \end{cases}$$

We note now that (3) tells us that $\lceil n/2^j \rceil = n - \sum_{s=1}^{j} |A_s|$; therefore, for any $i > j$ it holds $\lceil n/2^i \rceil - \lceil n/2^j \rceil = -\sum_{s=j+1}^{i} |A_s|$. By using these two relations we get

(A1.1)

$$a = \begin{cases} x \ominus \left(\displaystyle\sum_{s=i_{r-1}+1}^{i_r} |A_s| + \sum_{s=i_{r-3}+1}^{i_{r-2}} |A_s| + \cdots + \sum_{s=i_1+1}^{i_2} |A_s| \right) & \text{if } r \text{ is even,} \\ \dfrac{n}{2} + x \ominus \left(\displaystyle\sum_{s=i_{r-1}+1}^{i_r} |A_s| + \sum_{s=i_{r-3}+1}^{i_{r-2}} |A_s| + \cdots + \sum_{s=i_2+1}^{i_3} |A_s| + \sum_{s=1}^{i_1} |A_s| \right) & \text{if } r \text{ is odd.} \end{cases}$$

The set $\inf(x, t)$ is then formed by all the elements satisfying the relation (A1.1) for any choice of $i_1, \cdots, i_r$ with $r \geqq 1$ and $1 \leqq i_1 < \cdots < i_r \leqq t$. Varying $i_1, \cdots, i_r$ with r even we obtain all the $2^{t-1} - 1$ nonzero elements in S_t. In case r is odd, noting that the addition of $|A_1| = n/2$ modulo $n/2$ does not affect the sums in (A1.1), varying the i_j's we obtain each element in S_t. Hence $\inf(x, t) = \{x \ominus z | z \in S_t\} \cup \{n/2 + (x \ominus z) | z \in S_t\}$. In the same way, if $x \geqq n/2$ we can get $\inf(x, t) = \{x \oplus z | z \in S_t\} \cup \{n/2 + (x \oplus z) | z \in S_t\}$. □

Proof of Property 2. The proof is by induction on t. For $t = 1$ property is obvious, since $\inf(x, 1)$ is equal to $\{x, f_1(x) = x + n/2\}$ if $x < n/2$, and to $\{x, f_1(x) = x - n/2\}$ if $x \geqq n/2$. Let $\inf(x, t-1)$ satisfy (8). From Property 1 we can write $\inf(x, t) = \{a_1, \cdots, a_{2^{t-2}}\} \cup \{n/2 + a_1, \cdots, n/2 + a_{2^{t-2}}\}$, with $a_1 < \cdots < a_{2^{t-2}} \leqq n/2$. Let $x < n/2$, from Property 1 one has

$$\begin{aligned} \inf(x,t) = \inf(x,t-1) &\cup \{a_1 \ominus |A_t|, \cdots, a_{2^{t-2}} \ominus |A_t|\} \\ &\cup \{n/2 + (a_1 \ominus |A_t|), \cdots, n/2 + (a_{2^{t-2}} \ominus |A_t|)\}. \end{aligned}$$

Suppose first $a_1 \geqq |A_t|$. We prove that

$$\begin{aligned} a_1 - |A_t| < a_1 < \cdots < a_{i-1} < a_i - |A_t| < a_i < \cdots < a_{2^{t-2}} < n/2 + a_1 - |A_t| \\ < n/2 + a_1 < \cdots < n/2 + a_{i-1} < n/2 + a_i - |A_t| < n/2 + a_i \\ < \cdots < n/2 + a_{2^{t-2}} \end{aligned}$$

and that (8) holds. The differences

(A1.2) $\quad a_i - (a_i - |A_t|)$ and $n/2 + a_i - (n/2 + a_i - |A_t|)$ for $i = 1, \cdots, 2^{t-2}$

are all equal to $|A_t| > 0$. Moreover, by (2), $|A_t| \in \{\lceil n/2^t \rceil, \lceil n/2^t \rceil - 1\}$ and (8) holds in these cases. Consider now the remaining differences, that is, $(a_i - |A_t|) - a_{i-1}$, for $i = 2, \cdots, 2^{t-2}$, $(n/2 + a_1 - |A_t|) - a_{2^{t-2}}$, and $n + (a_1 - (n/2 + a_{2^{t-2}}))$. They can be written all as the difference between two consecutive informers in $\inf(x, t-1)$ (e.g., $a_i - a_{i-1}$, $i = 2, \cdots, 2^{t-2}$ and $n + a_1 - (n/2 + a_{2^{t-2}})$) minus $|A_t|$. Therefore, by the inductive hypothesis they can assume only the two possible values $\lceil n/2^{t-1} \rceil - |A_t|$ and $\lceil n/2^{t-1} \rceil - |A_t| - 1$. Since (3) tells us that $\lceil n/2^{t-1} \rceil = n - \sum_{j=1}^{t-1} |A_j|$, we have $\lceil n/2^{t-1} \rceil - |A_t| = n - \sum_{j=1}^{t} |A_j| = \lceil n/2^t \rceil > 0$. Hence all differences are >0 and (8) holds. Suppose now that $a_1 < |A_t|$. We prove that

$$a_1 < a_2 - |A_t| < \cdots < a_{i-1} < a_i - |A_t| < a_i < \cdots < a_{2^{t-2}} < a_1 \ominus |A_t|$$

$$< \frac{n}{2} + a_1 < \cdots < \frac{n}{2} + a_{i-1} < \frac{n}{2} + a_i - |A_t| < \frac{n}{2} + a_i < \cdots < \frac{n}{2} + a_{2^{t-2}}$$

$$< \frac{n}{2} + (a_1 \ominus |A_t|)$$

and that (8) holds.

Note first that for $i > 1$ and $t \leqq \lfloor \log n \rfloor$, by the inductive hypothesis we get $a_i > a_2 \geqq a_1 + \lceil n/2^{t-1} \rceil - 1 \geqq \lceil n/2^t \rceil \geqq |A_t|$, that is, $a_i - |A_t| \geqq 0$, for $i \geqq 2$. The differences

$$\text{(A1.3)} \qquad \begin{array}{l} a_i - (a_i - |A_t|), \qquad (n/2 + a_i) - (n/2 + a_i - |A_t|) \quad \text{for } i = 2, \cdots, 2^{t-2}, \\ n + [a_1 - (n/2 + (a_1 \ominus |A_t|))], \quad \text{and} \quad (n/2 + a_1) - (a_1 \ominus |A_t|) \end{array}$$

are all equal to $|A_t| \in \{\lceil n/2^t \rceil, \lceil n/2^t \rceil - 1\}$ and (8) holds for them. Moreover, as in the previous case, we get that all the remaining differences can be written as the difference (modulo n) between two consecutive informers in $\inf(x, t-1)$ minus $|A_t|$. Therefore (8) holds in any case for $x < n/2$. For $x \geqq n/2$ it is possible to make a similar proof, noting that from Property 1, $\inf(x, t) = \inf(x, t-1) \cup \{a_1 \oplus |A_t|, \cdots, a_{2^{t-2}} \oplus |A_t|\} \cup \{n/2 + (a_1 \oplus |A_t|), \cdots, n/2 + (a_{2^{t-2}} \oplus |A_t|)\}$. □

Proof of Property 3. It is immediate from Property 2. In fact it tells us that $\inf(x, t-1) = \{a_1, \cdots, a_{2^{t-1}}\}$ for some $a_1 < a_2 < \cdots < a_{2^{t-1}}$; $\inf(f_i(x), t-1) = \{b_1, \cdots, b_{2^{t-1}}\}$ for some $b_1 < b_2 < \cdots < b_{2^{t-1}}$. Moreover, $\inf(x, t) = \{a_1, \cdots, a_{2^{t-1}}\} \cup \{b_1, \cdots, b_{2^{t-1}}\} = \{c_1, \cdots, c_{2^t}\}$ with $c_1 < c_2 < \cdots < c_{2^t}$. Hence $a_i \neq b_j$ for $1 \leqq i, j \leqq 2^{t-1}$. □

Appendix 2. Proof of Lemma 3. First note that calls at time $t = 1$ use edges in the set $\{(i, n/2 + (i \ominus 1)) \mid i = 0, \cdots, n/2 - 1\}$. It remains to show that the edges in $E(0, n/2) \cup E(n/2, n/2)$ support the calls from time 2 to $\lceil \log n \rceil$, i.e., the internal broadcast to $\{0, \cdots, n/2 - 1\}$ and $\{n/2, \cdots, n-1\}$, according to the given scheme.

We will show the following more general fact.

FACT 1. The set of edges $E(a, b)$ allows the broadcast in time $\lceil \log b \rceil$ according to the given scheme on each of the three sets

$$\{a, \cdots, a+b-1\}, \quad \{a+1, \cdots, a+b-1\}, \quad \{a, \cdots, a+b-2\}.$$

This is obvious for $b = 1$ and $b = 2$. We show now that if

$$\begin{array}{lll} E(a, b/2) & \text{and} \quad E(a + b/2, b/2) & \text{if } b \text{ is even}, \\ E(a, \lceil b/2 \rceil) & \text{and} \quad E(a + \lfloor b/2 \rfloor, \lceil b/2 \rceil) & \text{if } b \text{ is odd} \end{array}$$

satisfies Fact 1 then also $E(a, b)$ satisfies Fact 1. Let x be the node that has to perform the broadcast.

Case 1: *Broadcast on* $\{a, \cdots, a+b-1\}$. Suppose that b is even. The node x that has to broadcast calls along the edge $(i, i+b/2) \in E(a, b)$ with either $x = i$ or $x = i + b/2$. At successive time units i and $i + b/2$ can broadcast to $\{a, \cdots, a + b/2 - 1\}$ and $\{a + b/2, \cdots, a + b - 1\}$ using the edges in $E(a, b/2)$ and $E(a + b/2, b/2)$, respectively. The time needed is $1 + \lceil \log b/2 \rceil \leqq \lceil \log b \rceil$.

Suppose that b is odd. A node $x \leqq a + \lfloor b/2 \rfloor - 1$ calls the node $x + \lceil b/2 \rceil \in \{a + \lceil b/2 \rceil, \cdots, a + b - 1\}$. The node $x = a + \lfloor b/2 \rfloor$ calls $x + 1 = a + \lceil b/2 \rceil$. By the inductive hypothesis, at successive time units x can broadcast to the set $\{a, \cdots, a + \lfloor b/2 \rfloor\}$ with edges in $E(a, \lceil b/2 \rceil)$ and the other informed node can broadcast to $\{a + \lfloor b/2 \rfloor + 1, \cdots, a + b - 1\}$ with edges in $E(a + \lfloor b/2 \rfloor, \lceil b/2 \rceil)$.

A node $x \geqq a + \lceil b/2 \rceil$ calls the node $x - \lceil b/2 \rceil \in \{a, \cdots, a + \lfloor b/2 \rfloor\}$. By the inductive hypothesis, at successive time units x can broadcast to $\{a + \lfloor b/2 \rfloor, \cdots, a + b - 1\}$ with edges in $E(a + \lfloor b/2 \rfloor, \lceil b/2 \rceil)$ and the other informed node to $\{a, \cdots, a + \lceil b/2 \rceil - 2\}$ with edges in $E(a, \lceil b/2 \rceil)$.

The time needed is $1 + \lceil \log \lceil b/2 \rceil \rceil \leqq \lceil \log b \rceil$.

Case 2: *Broadcast on* $\{a + 1, \cdots, a + b - 1\}$. In this case $a + 1 \leqq x \leqq a + b - 1$. Suppose that b is even. The node $x < a + b/2$ calls $x + b/2 \in \{a + b/2 + 1, \cdots, a + b - 1\}$. By the inductive hypothesis x can continue broadcasting to $\{a + 1, \cdots, a + b/2 - 1\}$ with edges in $E(a, b/2)$, and $x + b/2$ to $\{a + b/2, \cdots, a + b - 1\}$ with edges in $E(a + b/2, b/2)$.

A node $x \geqq a + b/2 + 1$ calls $x - b/2 \in \{a + 1, \cdots, a + b/2 - 1\}$. The node $x = a + b/2$ calls $x - 1 = a + b/2 - 1$. By the inductive hypothesis x can continue broadcasting to $\{a + b/2, \cdots, a + b - 1\}$ with edges in $E(a + b/2, b/2)$ and the other informed node to $\{a + 1, \cdots, a + b/2 - 1\}$ with edges in $E(a, b/2)$. Again the time needed is $1 + \lceil \log b/2 \rceil \leqq \lceil \log b \rceil$.

Suppose that b is odd. A node $x \leqq a + \lfloor b/2 \rfloor - 1$ (respectively, $x = a + \lfloor b/2 \rfloor$) calls the node $x + \lceil b/2 \rceil$ (respectively, $a + \lfloor b/2 \rfloor + 1$) in $\{a + \lceil b/2 \rceil, \cdots, a + b - 1\}$. By the inductive hypothesis, at successive time units x can broadcast to $\{a + 1, \cdots, a + \lfloor b/2 \rfloor\}$ with edges in $E(a, \lceil b/2 \rceil)$ and the other informed node to $\{a + \lceil b/2 \rceil, \cdots, a + b - 1\}$ with edges in $E(a + \lfloor b/2 \rfloor, \lceil b/2 \rceil)$.

Consider now $x > a + \lfloor b/2 \rfloor$. The node $x = a + \lceil b/2 \rceil$ calls $a + \lfloor b/2 \rfloor$. A node $x \geqq a + \lceil b/2 \rceil + 1$ calls the node $x - \lceil b/2 \rceil \in \{a + 1, \cdots, a + \lfloor b/2 \rfloor - 1\}$. By the inductive hypothesis, at successive time units x can broadcast to $\{a + \lceil b/2 \rceil, \cdots, a + b - 1\}$ with edges in $E(a + \lfloor b/2 \rfloor, \lceil b/2 \rceil)$ and the other informed node to $\{a + 1, \cdots, a + \lfloor b/2 \rfloor\}$ with edges in $E(a, \lceil b/2 \rceil)$.

The time needed is $1 + \lceil \log \lceil b/2 \rceil \rceil \leqq \lceil \log b \rceil$.

Case 3: *Broadcast on* $\{a, \cdots, a + b - 2\}$. In this case $a + 1 \leqq x \leqq a + b - 1$ can be easily proved using the same arguments as Case 2. □

REFERENCES

[1] K. BERMAN AND M. HAWRYLYCZ, *Telephone problems with failures*, SIAM J. Algebraic Discrete Methods, 7 (1986), pp. 13–17.

[2] J. C. BERMOND, P. HELL, A. L. LIESTMAN, AND J. G. PETERS, *New minimum broadcast graphs and sparse broadcast graphs*, Discrete Applied Math., to appear.

[3] S. C. CHAU AND A. L. LIESTMAN, *Constructing minimal broadcast networks*, J. Combin. Inform. Systems Science, 10 (1985), pp. 110–122.

[4] ———, *Constructing fault-tolerant minimal broadcast networks*, J. Combin. Inform. Systems Science, 11 (1986), pp. 1–18.

[5] A. M. FARLEY, *Minimal broadcast networks*, Networks, 9 (1979), pp. 313–332.

[6] A. M. FARLEY, S. T. HEDETNIEMI, S. MITCHELL, AND A. PROSKUROWSKI, *Minimum broadcast graphs*, Discrete Math., 25 (1979), pp. 189–193.

[7] A. M. FARLEY AND A. PROSKUROWSKI, *Broadcasting in trees with multiple originators*, SIAM J. Algebraic Discrete Methods, 2 (1979), pp. 189–193.

[8] L. GARGANO AND U. VACCARO, *On the construction of minimal broadcast networks*, Networks, 19 (1989), pp. 673–689.

[9] M. GRIGNI AND D. PELEG, *Tight bounds on minimum broadcast graphs*, SIAM J. Discrete Math., 4 (1991), pp. 207–222.

[10] R. W. HADDAD, S. ROY, AND A. A. SCHÄFFER, *On gossiping with faulty telephone lines*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 439–445.

[11] S. M. HEDETNIEMI, S. T. HEDETNIEMI, AND A. LIESTMAN, *A survey of broadcasting and gossiping in communication networks*, Networks, 18 (1988), pp. 319–351.

[12] A. L. LIESTMAN, *Fault-tolerant broadcast graphs*, Networks, 15 (1985), pp. 159–171.

[13] G. MADDALUNO, *Algorithms for the construction of fault-tolerant networks*, thesis, Università di Salerno, 1987. (In Italian.)

[14] S. MITCHELL AND S. T. HEDETNIEMI, *A census of minimum broadcast graphs*, J. Combin. Inform. Systems Science, 5 (1980), pp. 119–129.

[15] D. PELEG AND A. A. SCHÄFFER, *Time bounds on fault tolerant broadcasting*, Networks, 19 (1989), pp. 803–822.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 199–206, May 1992
© 1992 Society for Industrial and Applied Mathematics
004

HAMILTON PATHS IN GRAPHS OF LINEAR EXTENSIONS FOR UNIONS OF POSETS*

GRZEGORZ STACHOWIAK†

Abstract. This paper proves that if a poset Q has an even number of linear extensions and these extensions can be generated by adjacent transpositions, then linear extensions of union of poset Q and an arbitrary poset P can also be generated by adjacent transpositions. This result is then applied to posets P and Q, which are sums of disjoint chains.

Key words. linear extension, Hamilton path, Gray code

AMS(MOS) subject classification. 05C45

1. Introduction. Let P be a partially ordered set (poset) and $\mathscr{L}(P)$ denote the set of all its linear extensions. For two posets P and Q, we define their *parallel* $P|Q$ and *series* PQ compositions. The poset $P|Q$ is the union of P and Q, and the poset PQ is constructed from $P|Q$ by adding the relations that state that each element of P is greater than each element of Q. Let i^n denote a chain of n elements labeled with i; i^0 denotes the empty chain.

A linear extension $L_2 \in \mathscr{L}(P)$ can be obtained from another linear extension $L_1 \in \mathscr{L}(P)$ by an *elementary transformation* if L_1 and L_2 differ by the order of exactly two adjacent elements. We define the graph $GE(P)$ of all linear extensions of P with the elements of $\mathscr{L}(P)$ as its vertices and two vertices form an edge in $GE(P)$ if the corresponding linear extensions differ by a single interchange of adjacent elements. Note that a Hamilton path in $GE(P)$ implies the existence of an algorithm that generates all linear extensions of P by a sequence of elementary transformations. In general, we may consider an arbitrary family of combinatorial objects and ask for such an algorithm, which is called the *Gray code* of these objects. For two posets P and Q on the same ground set, the graph of $GE(P, Q)$ is defined analogously. The vertex set of $GE(P|Q)$ is $\mathscr{L}(P) \cup \mathscr{L}(Q)$, and two vertices are adjacent if they differ by an elementary transformation (i.e., by transposition of two adjacent elements in linear extensions).

In some cases, the problem of generating some combinatorial objects can be transformed to generating linear extensions of a properly defined poset. A motivation for this paper is the following problem: how do we generate all strings containing precisely $n(1) - 1$'s, $n(2) - 2$'s, $\cdots$, $n(k) - k$'s? The existence of Gray code for such strings is equivalent to the existence of a Hamilton path in the graph $H = GE(1^{n(1)}|2^{n(2)}| \cdots |k^{n(k)})$. This problem was considered by Ko and Ruskey [6].

It is well known that the graph $G = GE(P)$ is bipartite and connected for any poset P. Let $V_1 \cup V_2$ be a bipartition of $V(G)$ and $d(G) = \|V_1| - |V_2\|$. There is no Hamilton path in $GE(P)$ if $d(G)$ exceeds 1. Ko and Ruskey gave the formula for $d(H)$, and they posed the following conjecture.

CONJECTURE. *Graph H has a Hamilton path if and only if $d(H)$ is 0 or 1.*

This conjecture has been proved by Ruskey [8] for transpositions of not necessarily adjacent elements as well as for adjacent transpositions in some special cases. We show it for all graphs H.

* Received by the editors June 1, 1990; accepted for publication (in revised form) December 14, 1990.

† Institute of Computer Science, University of Wrocław, Przesmyckiego 20, 51-151 Wrocław, Poland. This research was partially supported by Institute of Informatics, University of Warsaw grant RP.I.09.

It was shown in [6] that $d(H)$ is 1 if $k = 1$ or $k = 2$ and either $n(1)$ or $n(2)$ is equal to 1 and the remaining one is even. It is not difficult to prove the conjecture in this case. The value $d(H)$ is zero if and only if for some i and j ($i \neq j$), $n(i)$ and $n(j)$ are both odd; without loss of generality, we can assume that $i = 1$ and $j = 2$. It is well known that $GE(1^{n(1)}|2^{n(2)})$ has a Hamilton path for $n(1)$ and $n(2)$ odd; see [2], [3], and [7]. For k greater than 2, the conjecture follows from our main result.

THEOREM. *If for a poset Q the graph $GE(Q)$ has an even number of vertices and contains a Hamilton path, then the graph $GE(Q|P)$ has a Hamilton path for every poset P.*

If $Q = 1^{n(1)}|2^{n(2)}$, where $n(1)$ and $n(2)$ are odd, and $P = 3^{n(3)}| \cdots |k^{n(k)}$, then the conditions of the theorem are fulfilled (see [2]); therefore, the conjecture is true.

In what follows, we will make frequent use of the operation of gluing two edges. Edges $e_1 = [a, b]$ and $e_2 = [c, d]$ of a graph G are *parallel* if G contains edges $f_1 = [a, c]$ and $f_2 = [b, d]$. Let us assume that there exist two vertex-disjoint cycles C_1 and C_2 in G such that C_1 contains e_1 and C_2 contains e_2. The *gluing* of edges e_1 and e_2 consists of exchanging e_1 and e_2 for f_1 and f_2. This operation produces one cycle containing all the vertices of cycles C_1 and C_2.

2. The theorem for Q consisting of two isolated elements. In this section we prove the theorem for $Q = 1|2$. An edge in $GE(Q|l^p)$ is called an *i-edge* if both its end vertices correspond to linear extensions of $Q|l^p$ in which the ith and $(i+1)$th elements of l^p are adjacent. A set consisting of i-edges e_i ($1 \leqq i < p$) such that $e_i \neq e_j$ for $i \neq j$ is called a *full set of i-edges*. We first formulate an obvious lemma.

LEMMA 1. *If a vertex $x \in G$ of degree 2 belongs to a cycle in G, then the cycle contains both of the edges incident to x.*

Now we prove some properties of graph $GE((1|2)|l^p)$ that will be useful later.

LEMMA 2. *The graph $G = GE((1|2)|l^p)$ has the following properties:*

1. *If p is odd, then G contains a Hamilton cycle containing a full set of i-edges.*
2. *If p is even, then G contains two cycles. Each of these cycles contains all but two vertices of G. The omitted vertices are either*
   - 2.1. *$12l^p$ and $21l^p$ (see Fig. 1) or*
   - 2.2. *l^p12 and l^p21.*

Moreover, if $p \neq 2$ then both cycles have the same full set of i-edges.

Proof. The graph G has $p+1$ vertex-disjoint paths d_i ($0 \leqq i \leqq p$) of the form

$$d_i = [1l^{p-i}2l^i, l1l^{p-i-1}2l^i, \cdots, l^{p-i}12l^i, l^{p-i}21l^i, \cdots, 2l^{p-i}1l^i],$$

and these paths cover all the vertices of G. Let a_i denote the last edge of d_i, $0 \leqq i \leqq p$. Connecting the ends of d_0 and d_1, d_2 and d_3, $\cdots$ we obtain $\lfloor (p+1)/2 \rfloor$ disjoint cycles in G and if p is even, there exists one additional path d_p, which is the edge $[12l^p, 21l^p]$. Then, by gluing a_1 and a_2, a_3 and a_4, and so on, we obtain a Hamilton cycle if p is odd, and we obtain a cycle omitting only two vertices $12l^p$ and $21l^p$ if p is even.

We obtain case 2.2 in Lemma 2 if the above construction is repeated for paths d'_i ($0 \leqq i \leqq p$) defined as

$$d'_i = [l^i2l^{p-i}1, l^i2l^{p-i-1}1l, \cdots, l^i21l^{p-i}, l^i12l^{p-i}, \cdots, l^i1l^{p-i}2].$$

It remains only to exhibit i-edges in the constructed cycles. Each cycle contains edges $k_i = [l^i12l^{p-i}, l^i21l^{p-i}]$ for $1 \leqq i < p$. As a full set of i-edges, we take $e_1 = k_{p-1}$ and $e_i = k_{i-1}$ for $1 < i$. □

The graph $GE(Q|P)$ for two arbitrary posets P and Q consists of the subgraphs $GE(Q|L)$ ($L \in \mathscr{L}(P)$) connected to each other. All these subgraphs are isomorphic to $GE(Q|l^p)$, where $p = |P|$. We can therefore consider i-edges in $GE(Q|L)$ as those

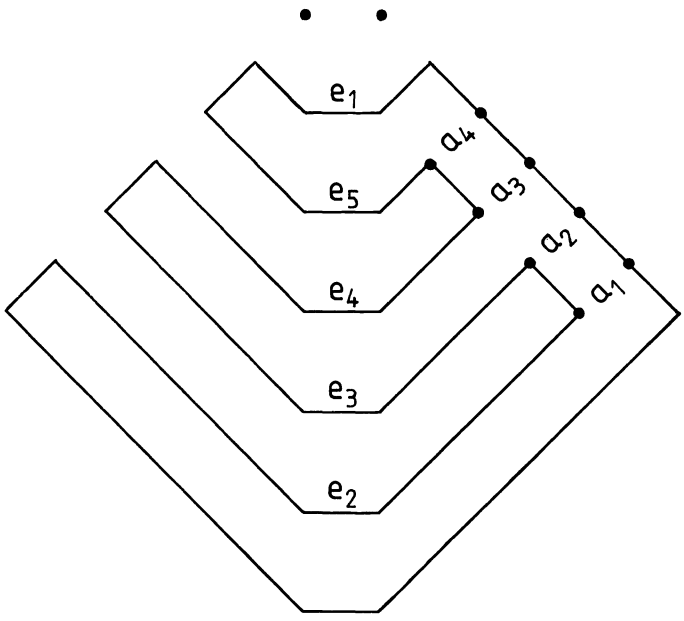


FIG. 1. *A long cycle defined in Lemma* 2, *case* 2.1, *for* $p = 6$.

having ith and $(i + 1)$th places of L adjacent. We can define analogously a full set of i-edges in $GE(Q|L)$.

We now prove a lemma similar to the result of Batagelj and Pisanski [1].

LEMMA 3. *Let a graph* $G = GE(Q|P)$ *contain a family* $\mathscr{C}$ *of cycles such that each* $GE(Q|L)$ *as a subgraph of* G *contains exactly one cycle of* $\mathscr{C}$. *If each of these cycles has a full set of* i*-edges corresponding to a fixed full set in* $GE(Q|l^p)$, *cycles in* $\mathscr{C}$ *can be combined into one cycle containing all their vertices.*

Proof. The graph $GE(P)$ is connected, so it contains a spanning tree, say T. To each edge $[L_1, L_2]$ of T we assign a number i $(1 \leqq i < p)$ such that L_1 can be obtained from L_2 by an elementary transformation on the ith and the $(i + 1)$th elements. Thus, the i-edges in the graphs $GE(Q|L_1)$ and $GE(Q|L_2)$ for a fixed i are parallel. We glue all such edges to obtain one cycle from the cycles of $\mathscr{C}$. These operations of gluing are possible because there is at most one edge labeled by i $(1 \leqq i < p)$ adjacent to a given vertex of $GE(P)$. □

Now we are ready to formulate the first step in the proof of the theorem.

LEMMA 4. *The graph* $G = GE(1|2|P)$ *contains a Hamilton cycle when* $|P|$ *is odd.*

Proof. The graph G consists of subgraphs $GE(1|2|L)$ $(L \in \mathscr{L}(P))$ isomorphic to $GE(1|2|l^p)$. Lemma 2, case 1 gives us Hamilton cycles in $GE(1|2|L)$ with full sets of i-edges. Using Lemma 3, we can connect all these cycles obtaining Hamilton cycle in $GE(1|2|P)$. □

Rewriting the same proof for p even is complicated by the lack of a Hamilton cycle (Lemma 12) in $GE(1|2|l^p)$. We eliminate this difficulty by first using Lemma 3 to connect cycles from Lemma 2 and next applying Lemma 5 to attach the remaining vertices to the obtained cycle.

LEMMA 5. *If* P *is not a chain, then* $GE(P)$ *can be split by removing some edges into paths of number of vertices greater than one.*

Proof. We proceed by induction on $p = |P|$. If $p \leqq 2$, then Lemma 5 is obviously true. Let us assume that it holds for all posets on $(p - 1)$-elements, where $p \geqq 3$ and let P be a p-element poset. There are two possible cases: P either contains or does not contain

a $(p-1)$-element chain. In the former case, $GE(P)$ is a path of number of vertices at least two. In the latter case, $GE(P)$ does not have such a path. Let $x \in P$ be a minimal element in P and consider the poset $R = P - x$. By the inductive assumption, R can be split into paths of number of vertices at least two. Hence the graph $GE(P)$, which consists of graphs $GE(P-x)$ for all minimal $x \in P$, can be split into such paths. □

We can now prove the following lemma.

LEMMA 6. *Let $G = GE(1|2|P)$ where $p = |P|$ is even. If P is a chain, then G contains a Hamilton path, and if P is not a chain, then G contains a Hamilton cycle.*

Proof. If $P = l^p$, then by case 2.1 of Lemma 2 we have in $GE(1|2|l^p)$ the edge $[12l^p, 21l^p]$ and a cycle containing the remaining vertices. The vertex $21l^p$ from the edge is a neighbor of the vertex $2l1l^{p-1}$ belonging to the cycle. So if we have given only this edge and cycle, then by adding the edge $[21l^p, 2l1l^{p-1}]$ and removing one of the edges adjacent to $2l1l^{p-1}$ from the cycle, we obtain a Hamilton path in G.

If P is not a chain, then we consider two cases: $p = 2$ and $p > 2$. For $p = 2$, the only poset to consider is $1|2|3|4$. In this case, a Hamilton cycle can be easily obtained (by the Steinhaus–Johnson–Trotter algorithm [5], [9], [10] or by applying Lemma 10 for $Q = 1|2|3$ and $P = \{4\}$). If $p > 2$, then by Lemma 5, $GE(P)$ can be split into a family of paths $\mathscr{D}$ of number of vertices at least two. Let $D = [L_{D(1)}, L_{D(2)}, \cdots, L_{D(s(D))}] \in \mathscr{D}$. Let $G_{D(i)}$ denote the graph $GE(1|2|L_{D(i)})$. In $G_{D(1)}$, considered as a subgraph of $GE(1|2|P)$, we take a cycle defined in the proof of Lemma 2 (case 2.2). In all other subgraphs $G_{D(i)}$, we take the cycles defined in the proof of case 2.1 of Lemma 2. Note that the cycle from case 2.1 must contain the edge $[l^p12, l^p21]$ and the cycle from case 2.2 must contain the edge $[12l^p, 21l^p]$ because of Lemma 1. We combine these cycles into one cycle applying Lemma 3, and then we replace the edges $[12L_{D(1)}, 21L_{D(1)}]$ by paths $[12L_{D(1)}, 12L_{D(2)}, \cdots, 12L_{D(s(D))}, 21L_{D(s(D))}, \cdots, 21L_{D(1)}]$ and the edges $[L_{D(2)}12, L_{D(2)}21]$ by paths $[L_{D(2)}12, L_{D(1)}12, L_{D(1)}21, L_{D(2)}21]$ (see Fig. 2). This results in a Hamilton cycle in G for $p \neq 2$. □

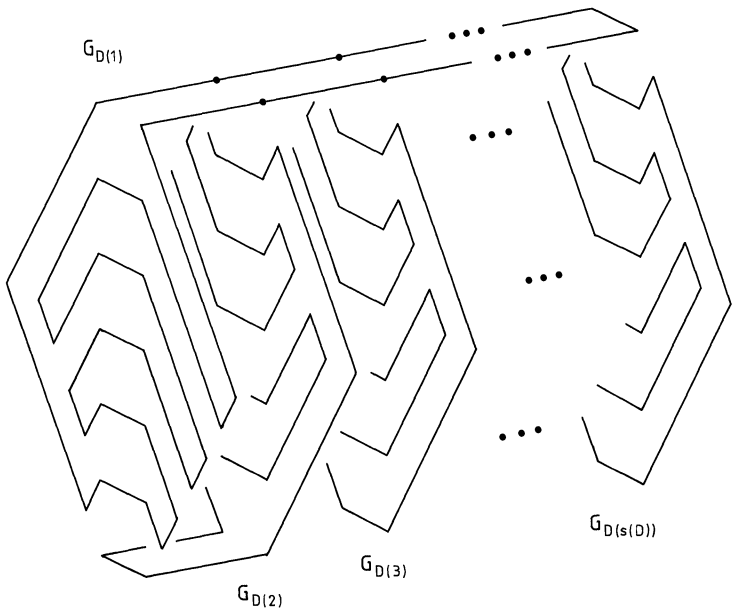


FIG. 2. *The method of connecting to long cycles in $G_{D(i)}$ vertices, which are absent in these cycles, to obtain a Hamilton cycle, as in Lemma* 6.

Lemmas 4 and 6 constitute a proof of the theorem for Q consisting of two isolated elements. This result was proved for not necessarily adjacent transpositions by Göbel [4].

3. The theorem for an arbitrary Q. We first provide another proof of a useful lemma from [2].

LEMMA 7. *The graph* $G = GE((1|2)(k^q|l^p))$ *contains a Hamilton cycle for every* $p, q > 0$.

Proof. We proceed by induction on p. If $p = 1$, then the cycle is of the form $[12k^ql, 12k^{q-1}lk, \cdots, 12lk^q, 21lk^q, \cdots, 21k^ql]$. Let us assume that Lemma 7 holds for $p-1$ and split G into graphs $G_i = GE((1|2)k^{q-i}l(k^i|l^{p-1}))$ $(0 \leqq i \leqq q)$. By the inductive hypothesis, each G_i for $i > 0$ has a Hamilton cycle that contains the following edges:

$$a_i = [21k^{q-i}lk^il^{p-1}, 12k^{q-i}lk^il^{p-1}]$$

and

$$b_i = [12k^{q-i}lk^il^{p-1}, 12k^{q-i}lk^{i-1}lkl^{p-2}],$$

because vertex $12k^{q-i}lk^il^{p-1}$ is of degree 2 in G_i. Then, replacing the edge a_1 in G_1 by the path $[21k^{q-1}lkl^{p-1}, 21k^ql^p, 12k^ql^p, 12k^{q-1}lkl^{p-1}]$ we join G_0 to G_1. Finally, we glue the edges b_1 and b_2, b_3 and b_4, $\cdots$ and a_2 and a_3, a_4 and a_5, $\cdots$ to obtain a Hamilton cycle in G. □

Now we use Lemma 7 to prove the following lemma.

LEMMA 8. *The graph* $G = GE(((1|2)k^q)|l^p)$ *contains a Hamilton cycle for every* $p, q > 0$.

Proof. We split G into subgraphs

$$G_i = GE((1|l^{p-i})2(k^q|l^i),(2|l^{p-i})1(k^q|l^i)) \qquad (0 \leqq i \leqq p)$$

each of which will be shown to contain a Hamilton path with end vertices $x_i = 1l^{p-i}2l^ik^q$ and $y_i = 2l^{p-i}1l^ik^q$. In particular, G_0 contains the path

$$[1l^p2k^q, l1l^{p-1}2k^q, \cdots, l^p12k^q, l^p21k^q, \cdots, 2l^p1k^q].$$

For $i > 0$ we again split G_i into subgraphs G_{ij} $(0 \leqq j \leqq p-i)$ of the form

$$G_{ij} = GE(l^{2j}(1|l)l^{p-i-2j-1}2(l^i|k^q)) \quad \text{for } 0 \leqq j < (p-i)/2,$$

$$G_{ij} = GE(l^{p-i}(1|2)(l^i|k^q)) \quad \text{for } j = (p-i)/2,$$

$$G_{ij} = GE(l^{2(p-i-j)+1}(l|2)l^{i+2j-p-1}1(l^i|k^q)) \quad \text{for } (p-i)/2 < j \leqq p-i.$$

By Lemma 7, each G_{ij} (as isomorphic to $GE((1|2)(l^i|k^q))$) has a Hamilton cycle that contains the edge $[x_{ij}, y_{ij}]$, where

$$x_{ij} = l^{2j}1l^{p-i-2j}2l^ik^q \quad \text{for } 0 \leqq j < (p-i)/2,$$

$$x_{ij} = l^{p-i}12l^ik^q \quad \text{for } j = (p-i)/2,$$

$$x_{ij} = l^{2(p-i-j)}2l^{i+2j-p}1l^ik^q \quad \text{for } (p-i)/2 < j \leqq p-i,$$

and

$$y_{ij} = l^{2j+1}1l^{p-i-2j-1}2l^ik^q \quad \text{for } 0 \leqq j < (p-i)/2,$$

$$y_{ij} = l^{p-i}21l^ik^q \quad \text{for } j = (p-i)/2,$$

$$y_{ij} = l^{2(p-i-j)-1}2l^{i+2j-p+1}1l^ik^q \quad \text{for } (p-i)/2 < j \leqq p-i.$$

By the same argument, G_{i0} contains the edge $a_i = [1l^{p-i}2l^ik^q, 1l^{p-i}2l^{i-1}klk^{q-1}]$. We remove edges $[x_{ij}, y_{ij}]$ from cycles in G_{ij} and then we add edges $[y_{ij}, x_{i(j+1)}]$ to obtain the required Hamilton path in G_i. Finally, we connect x_0 and x_1, x_2 and x_3, $\cdots$ y_0 and y_1, y_2 and y_3, $\cdots$ to obtain $\lfloor (p+1)/2 \rfloor$ cycles in G. If p is even, then there remains a path in G_p and we connect its end vertices to obtain the last cycle. By gluing a_1 and a_2, a_3 and a_4, $\cdots$, we obtain a Hamilton cycle in G. See Fig. 3. □

LEMMA 9. *The graph* $G = GE((k^r(1|2)k^s)|l^p)$ *contains a Hamilton cycle when* $p, r+s>0$.

Proof. If $s = 0$, then the graph G is isomorphic to $G = GE(((1|2)k^r)|l^p)$ and has a Hamilton cycle by Lemma 8. For $s > 0$ we proceed by induction on r. If $r = 0$, then our task can be reduced to Lemma 8. We divide G into subgraphs $G_i = GE(l^{p-i}k((k^{r-1}(1|2)k^s)|l^i))$ for $0 \leqq i \leqq p$. By the inductive hypothesis, there exists a Hamilton cycle C_i in every G_i for $i > 0$. Each cycle C_i contains the edges

$$a_i = [l^{p-i}kl^ik^{r-1}12k^s, l^{p-i}kl^ik^{r-1}21k^s],$$

and, for $r = 1$,

$$b_i = [l^{p-i}kl^i12k^s, l^{p-i}kl^{i-1}1l2k^s],$$

for $r > 1$,

$$b_i = [l^{p-i}kl^ik^{r-1}12k^s, l^{p-i}kl^{i-1}klk^{r-2}12k^s]$$

because vertex $l^{p-i}kl^ik^{r-1}12k^s$ is of degree 2.

In the cycle C_1 we exchange the edge a_1 for the path $[l^{p-1}klk^{r-1}12k^s, l^pk^r12k^s, l^pk^r21k^s, l^{p-1}klk^{r-1}21k^s]$ connecting G_0 to the cycle in G_1. Then we glue b_1 with b_2, b_3 with b_4, and so on, a_2 with a_3, a_4 with a_5, and so on, and we obtain a Hamilton cycle in G. □

LEMMA 10. *If* $q = |Q| > 2$, $|\mathscr{L}(Q)|$ *is even and* $GE(Q)$ *contains a Hamilton path and* $p > 0$ *then* $GE(Q|l^p)$ *has a Hamilton cycle.*

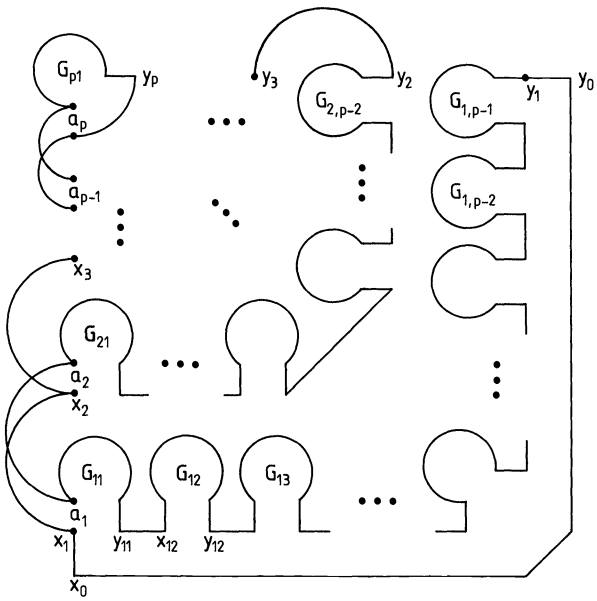


FIG. 3. *A Hamilton cycle defined in Lemma* 8 *for even* p.

Proof. Let $[K_1, K_2, \cdots, K_{2n}]$ be a Hamilton path in $GE(Q)$. We split $GE(Q|l^p)$ into n subgraphs $G_i = GE(K_{2i-1}|l^p, K_{2i}|l^p)$; $1 \leqq i \leqq n$. G_i is isomorphic to $GE((k^r(1|2)k^s)|l^p)$ for some r and s. Thus G_i has a Hamilton cycle C_i by Lemma 9. Let $K_j = k_{j1}k_{j2}\cdots k_{jq}$, and denote the edges

$$a_j = [l^p k_{j1}\cdots k_{jq}, l^{p-1}k_{j1}lk_{j2}\cdots k_{jq}]$$

and

$$b_j = [k_{j1}\cdots k_{jq}l^p, k_{j1}\cdots k_{j,q-1}lk_{jq}l^{p-1}].$$

Since $\deg(l^p k_1 \cdots k_q) = \deg(k_1 \cdots k_q l^p) = 2$, by Lemma 1 the edges a_{2i-1}, a_{2i}, b_{2i-1}, b_{2i} belong to C_i for $1 \leqq i \leqq n$. If K_j and K_{j+1} do not differ in the first pair of elements, then b_j and b_{j+1} are parallel. In the other case, a_j and a_{j+1} are parallel. We then glue parallel edges a_j and a_{j+1} or b_j and b_{j+1} of G_j and G_{j+1} for $j = 2i$ $(1 \leqq i < n)$ obtaining a Hamilton cycle in $GE(Q|l^p)$. See Fig. 4. □

LEMMA 11. *If* $q = |Q| > 2$, $p = |P| > 0$ *and* $GE(Q)$ *has an even number of vertices and contains a Hamiton path then* $GE(Q|P)$ *has a Hamilton cycle.*

Proof. We use Lemma 3 to build a Hamilton cycle in $GE(Q|P)$ from Hamilton cycles in $GE(Q|L)$ $(L \in \mathscr{L}(P))$. These cycles exist by Lemma 10, because $GE(Q|L)$ are isomorphic to $GE(Q|l^p)$. There remains to find a full set of i-edges in $GE(Q|l^p)$. Let $K = k_1k_2\cdots k_q$ be a linear extension of Q. The vertex $x_i = l^i k_1 \cdots k_q l^{p-i}(2 \leqq i \leqq p)$ is a linear extension of Q and has two neighbors in a Hamilton cycle in $GE(Q|l^p)$. At most one of these neighbors can be obtained from x_i by transposition of the ith and $(i+1)$th elements. Moreover, there exists an edge incident to x_i in $GE(Q|l^p)$ not equivalent to this transposition, and we take it as the $(i-1)$-edge e_{i-1}. This way, we obtain i-edges for $1 \leqq i \leqq p-1$. □

Proof of the theorem. For $P \neq \varnothing$ the theorem is obvious. For $|Q| = 2$, it follows from Lemmas 4 and 6, and for $|Q| > 2$, it follows from Lemma 11. □

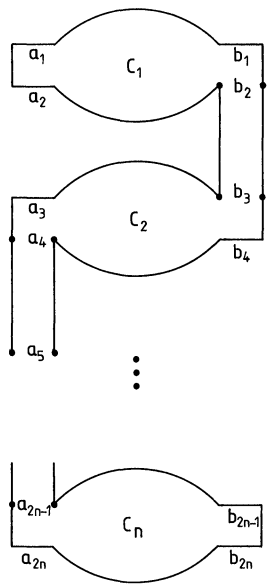


FIG. 4. *A Hamilton cycle constructed in proof of Lemma* 10.

Note that in some cases the lemmas give us more information than the theorem requires. Namely, a graph of type $GE(Q|P)$, where $GE(Q)$ has a Hamilton path and an even number of vertices, contains in many cases not only a Hamilton path, but also a Hamilton cycle. The only exceptions are graphs $GE(Q)$ and $GE((1|2)|l^p)$ where p is even. For the latter case, we prove the following result, which was originally proved by Ruskey [8].

LEMMA 12. *The graph* $G = GE((1|2)|l^p)$, *where* p *is even, has no Hamilton cycle.*

Proof. We can decompose G into two subgraphs $G_1 = GE(12|l^p)$ and $G_2 = GE(21|l^p)$.

These parts are isomorphic to each other. The graph G_i $(i = 1, 2)$ is bipartite, and we can show that $d(G_i)$ is equal to $p/2 + 1$. We proceed by induction on p. If p is equal to 0, then G_i has only one linear extension and $d(G_i) = 1$. If for $p - 2$, $d(G_i) = p/2$, then we can split G_1 for p into three parts (analogous consideration can be carried out for G_2): $H = GE((12|l^{p-2})l^2)$, which is isomorphic to G_1 for $p - 2$; $I = GE((1|l^{p-1})(2|l))$ such that $d(I) = 0$; $J = \mathrm{GE}(l^p12)$, which adds 1 to the result for $p - 2$ finishing our inductive consideration.

As we know, $d(G) = 0$. It follows from that fact that each bipartition of G consists of the greater set in the bipartition of one G_i and the smaller one of the second G_i. There are $p + 1$ edges $a_i = [l^i12l^{p-i}, l^i21l^{p-i}]$ between G_1 and G_2 in G. If C is a Hamilton cycle in G, then C can be split into at most p maximal paths such that each of them is contained in one G_i because every two successive paths must be joined by an edge a_j for some $0 \leqq j \leqq p$ and there are $p + 1$ such edges. Moreover, only p of these edges can be used by C since a cycle must contain an even number of a_i's.

Thus, there are at most $p/2$ paths in each G_i that cover all their vertices. This is, however, impossible because $d(G_i) = p/2 + 1$ $(i = 1, 2)$ and at least this number of paths is necessary to cover G_i for $i = 1, 2$. □

Acknowledgments. The author thanks Professor Maciej M. Sysło for helpful advice and for great patience in reading successive versions of this paper. The author also thanks the referees for remarks that improved the presentation.

REFERENCES

[1] V. BATAGELJ AND T. PISANSKI, *Hamilton cycles in the cartesian product of a tree and a cycle*, Discrete Math., 38 (1982), pp. 311–312.

[2] M. BUCK AND D. WIEDEMAN, *Gray codes with restricted density*, Discrete Math., 48 (1984), pp. 163–171.

[3] P. EADES, M. HICKEY, AND R. READ, *Some Hamilton paths and a minimal change algorithms*, J. Assoc. Comput. Mach., 31 (1984), pp. 19–29.

[4] F. GÖBEL, *On a problem of F. Ruskey*, Memorandum No. 832, Twente University of Technology, Faculty of Applied Mathematics, Enschede, the Netherlands, 1989.

[5] S. M. JOHNSON, *Generation of permutations by adjacent transpositions*, Math. Comp., 17 (1963), pp. 282–285.

[6] C. W. KO AND F. RUSKEY, *Solution of some multidimensional lattice path parity difference recurrence relations*, Discrete Math., 71 (1988), pp. 47–56.

[7] F. RUSKEY, *Adjacent interchange generation of combinations*, J. Algorithms, 9 (1988), pp. 162–180.

[8] ———, *Generating the linear extensions of posets by transpositions*, J. Combin. Theory Ser. B, to appear.

[9] H. STEINHAUS, *One Hundred Problems in Elementary Mathematics*, Dover, New York, 1979.

[10] H. F. TROTTER, *Algorithm* 115: PERM, Comm. ACM, 5 (1962), pp. 434–435.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 207–218, May 1992
© 1992 Society for Industrial and Applied Mathematics
005

ON THE NONMULTIPLICATIVITY OF ORIENTED CYCLES*

HUISHAN ZHOU†

Abstract. Graph homomorphism is an edge-preserving mapping from the vertex set of a graph to the vertex set of another graph, which is the generalization of graph coloring. A graph is multiplicative if the product of two graphs cannot be homomorphically mapped to it whenever the two factor graphs also cannot. The research of multiplicativity can be traced back to the mid-1960s, and has become active again in the past five years. There are some partial results about the multiplicativity of oriented cycles. The so-called even-deleting operation is now explored to prove that after implementing an even-deleting operation to a core-oriented cycle, both the resulting cycle and the original cycle are nonmultiplicative if the resulting cycle is not a special type of basic cycle. Finally, this paper proves that almost all oriented cycles are nonmultiplicative.

Key words. (non)multiplicative, oriented cycle, basic path, even-deleting operation

AMS(MOS) subject classification. 05C

1. Introduction. Hedetniemi conjectured in 1966 [8] that $\chi(G \times H) = \min(\chi(G), \chi(H))$, where χ is the chromatic number and $G \times H$ is the categorical product. This conjecture was verified only in a very few special cases. The cases $\min(\chi(G), \chi(H)) = 1, 2$, and 3 are easy [8]. The proof for $\min(\chi(G), \chi(H)) = 4$ was a major breakthrough [2]. To gain insights relevant for eventually solving Hedetniemi's conjecture, Haggkvist, Hell, Miller, and Lara [6] proposed the concept of multiplicativity and proved that undirected odd cycles, transitive tournaments, directed paths, and prime-power directed cycles are multiplicative. As a consequence, they also derived the multiplicativity of a large class of undirected graphs defined by Gerards (cf. [5], [6] for the exact definition of this class). We continue this task and obtain more classes of graphs and digraphs, some of which are multiplicative [17] and some of which are nonmultiplicative [18]. We now go further to prove that a large class of oriented cycles are nonmultiplicative.

For the definitions not given here, see [7]. In particular, *all graphs considered here are simple and finite. Furthermore, graphs G, H, etc., could be graphs or diagraphs; similarly, the edge gg', could mean the undirected edge* $\{g, g'\}$ *or the directed arc* gg'. The vertex set of G is denoted by $V(G)$ and the edge (arc) set of G is denoted by $E(G)$. The *product* $G \times H$ (also known as the categorical product [9], [13], conjunction [7], cardinal product, Kronecker product [3], [16], or weak direct product) has the vertex set $V(G) \times V(H)$ and the edges $(g, h)(g', h')$, where $gg' \in E(G)$ and $hh' \in E(H)$.

An *n-colouring* φ of G is a map of $V(G)$ to $\{1, 2, \cdots, n\}$ such that $gg' \in E(G)$ implies $\varphi(g) \neq \varphi(g')$. We say that G is *n-chromatic* and write $\chi(G) = n$ if n is the minimum k for which there is a k-colouring of G. A *homomorphism* $f: G \to H$ is a mapping $f: V(G) \to V(H)$ for which $f(g)f(g') \in E(H)$ whenever $gg' \in E(G)$. The existence, respectively, nonexistence, of a homomorphism $f: G \to H$ will be denoted by $G \to H$, respectively, $G \not\to H$. Note that there is an n-colouring of G just if $G \to K_n$, where K_n is a complete undirected graph of n vertices. Two graphs G and H are *homomorphically equivalent* if both $G \to H$ and $H \to G$. A *core* is a graph H with $H \not\to G$ for any proper subgraph G of H; i.e., a graph H in which each homomorphism $H \to H$ is an isomorphism (see [17]). Cores were studied in [4], [10], [17].

A directed (respectively, undirected) connected graph W is *multiplicative* [6] if $G \not\to W$ and $H \not\to W$ imply $G \times H \not\to W$ for all directed (respectively, undirected) graphs

* Received by the editors July 25, 1990; accepted for publication (in revised form) January 11, 1991.

† Department of Mathematics and Statistics, Simon Fraser University, Burnaby, British Columbia, V5A1S6, Canada.

G and H. It is important to note here that the graphs G and H are taken to be undirected graphs if W is undirected, and directed graphs if W is directed.

Let K_n denote the *complete undirected graph* with n vertices. A *complete digraph* with n vertices, denoted by $\vec{K}_n$, is a digraph with n vertices and an arc between every ordered pair of vertices. Thus each $\vec{K}_n$ has $n(n-1)$ arcs.

The *directed path* $\vec{P}_n$ has a sequence of different vertices $v_0, v_1, \cdots, v_n$ and arcs $v_0v_1, v_1v_2, \cdots, v_{n-1}v_n$. The *directed walk* $\vec{P}_n$ has a sequence of vertices, $v_0, v_1, \cdots, v_n$ and arcs $v_0v_1, v_1v_2, \cdots, v_{n-1}v_n$, where both vertices and arcs may not be different. An *oriented path* (*walk*) $P[x_0, x_1, \cdots, x_n]$ in a digraph G is a sequence of distinct (not necessarily distinct) vertices $x_0, x_1, \cdots, x_n$ and arcs $(x_0, x_1), (x_1, x_2), \cdots, (x_{n-1}, x_n)$, where (x_i, x_{i+1}) denotes either x_ix_{i+1} or $x_{i+1}x_i$. The *directed cycle* $\vec{C}_n$ has a sequence of different vertices $v_0, v_1, \cdots, v_{n-1}$ and arcs $v_0v_1, v_1v_2, \cdots, v_{n-2}v_{n-1}, v_{n-1}v_0$. An *oriented cycle* is a digraph obtained from a simple undirected cycle by choosing one direction for each edge.

Given a graph G, a colouring φ of G, and any graph H, there is a natural induced colouring φ' of $G \times H$, namely, $\varphi'(g, h) = \varphi(g)$. Thus we see that $\chi(G \times H) \leqq \min(\chi(G), \chi(H))$. Hedetniemi [8] conjectured that equality holds for all graphs G and H. An equivalent formulation of this conjecture is the following [1], [6]:

(i) For all positive integers $n+1$, $\chi(G) = \chi(H) = n+1$ implies that $\chi(G \times H) = n+1$ for all graphs G and H.

Or, using the general terminology introduced above,

(ii) K_n is multiplicative.

There are quite a few attacks on the original Hedetniemi problem [1], [2], [6] and also some research on the generalized version of Hedetniemi's problem—multiplicativity [6], [11], [12], [17], [18], [19].

In the literature there are several examples of (undirected or directed) graphs that are not multiplicative. However, in most instances, they were constructed by taking two connected graphs G and H with $G \not\to H$, $H \not\to G$, and by taking $W = G \times H$. (For instance, $\vec{C}_{pq} = \vec{C}_p \times \vec{C}_q$ is nonmultiplicative for this reason when p and q are relatively prime [6], [14].) The core $\vec{K}_n$ is the first known example [15], where a nonmultiplicative graph $W = \vec{K}_n$ is not constructed to be (homomorphically equivalent to) some $W = G \times H$, i.e., by the method explained above. Other examples are provided in [18].

We have already completely characterized the oriented paths with respect to multiplicativity. Thus, we have completely characterized the noncore oriented cycles, since noncore oriented cycles are homomorphically equivalent to some oriented subpath, and two homomorphically equivalent graphs have the same multiplicative or nonmultiplicative property [18]: Any oriented path and oriented cycle that is homomorphically equivalent to a directed path is multiplicative [6], [18]; any oriented path and oriented cycle that is not homomorphically equivalent to a directed path but is homomorphically equivalent to an oriented path is nonmultiplicative [17], [18]; any directed cycle with length of prime power is multiplicative [6], [11]; any directed cycle with length being not prime power is nonmultiplicative [6], [14].

For core oriented cycles we have also obtained some nonmultiplicative examples [18].

Now we apply the concept of basic b-path and the operation of even-deleting, to explore by the method of construction that many core oriented cycles are nonmultiplicative, which is the Main Theorem in § 3. Thus we have "almost" characterized all oriented cycles with respect to multiplicativity, with very few exceptions remaining open. We also prove in § 5 that almost all oriented cycles are nonmultiplicative. In an upcoming work [11], the only remaining exceptions of this paper will be characterized as multi-

plicative. Therefore we have completely characterized the oriented cycles with respect to multiplicativity. There are still many problems of characterizing other classes of graphs that remain open.

2. Preliminaries. The *level* of a vertex x in an oriented path P with respect to a chosen vertex a of P, denoted by $L_{P,a}(x)$, or simply $L(x)$ with $L(a) \equiv 0$ if no confusion results, is the difference between the number of edges directed forward and the number of edges directed backward on the subgraph from the chosen vertex a to x. The *net length* of an oriented path (cycle), denoted by nl, is the absolute value of the difference between the number of edges directed forward, and the number of edges directed backward along the path (cycle).

For the oriented cycle $C = P[x_0, x_1, x_2, x_0]$ in Fig. 2.1, we have $nl(C) = 1$.

Let $x_0, x_1, \cdots, x_n$ be the vertices of an oriented path P in a fixed *traversal order* or P. Then $P[x_i, x_{i+1}, \cdots, x_j]$, or, briefly, $P[x_i, x_j]$ is an *interval* of P consisting of the subgraph induced by the vertices $x_i, x_{i+1}, \cdots, x_j$. If all edges in the interval $P[x_i, x_j]$ are going in one direction (forward, backward), we call $P[x_i, x_j]$ a *one-directional* (*forward*, *backward*, respectively) *interval* of P. Furthermore, if the one-directional interval $P[x_i, x_j]$ cannot be extended to a larger one-directional interval of P, then we call $P[x_i, x_j]$ a *maximal one-directional* (*forward*, *backward*, respectively) *interval*. Let P_1 and P_2 be two oriented paths with the specified orders of traversal. Then the *concatenation* of P_1 and P_2, denoted by P_1P_2, is the oriented path obtained by identifying the last vertex of P_1 and the first vertex of P_2. Sometimes we write $v_1P_1v_2P_2v_3P_3v_4$ for $P_1P_2P_3$ to specify that the first vertex of P_1 is v_1, the last vertex of P_3 is v_4, the identified vertex of P_1 and P_2 is v_2, and the identified vertex of P_2 and P_3 is v_3. Furthermore, if P_1 is a maximal forward interval of length a, P_2 is a maximal backward interval of length b, and P_3 is a maximal forward interval of length c, then we write $v_1P_1v_2P_2v_3P_3v_4$ as $v_1\vec{P}_av_2\overleftarrow{P}_bv_3\vec{P}_cv_4$, or simply $\vec{P}_a\overleftarrow{P}_b\vec{P}_c$.

Let $b, a_1, a_2, \cdots, a_n, m_1, m_2, \cdots, m_{n-1}$ be positive integers where $n \geqq 2$ and all $a_i > b$ for $i = 1, 2, \cdots, n$. A *basic forward b-path* with parameters $(a_1, 2m_1 - 1; a_2, 2m_2 - 1; \cdots; a_{n-1}, 2m_{n-1} - 1; a_n)$ is an oriented path consisting of a $\vec{P}_{a_1}$ followed by $\overleftarrow{P}_b, \vec{P}_b, \cdots, \overleftarrow{P}_b$ ($2m_1 - 1$ repetitions), $\vec{P}_{a_2}$, then $\vec{P}_b, \overleftarrow{P}_b, \cdots, \vec{P}_b$ ($2m_2 - 1$ repetitions), etc., and ending with $\vec{P}_{a_n}$ (cf. Fig. 2.2). When we write $S = \cdots \overleftarrow{P}_i \cdots \vec{P}_j \cdots$, we always assume that each $\overleftarrow{P}_i$ (and $\vec{P}_j$) is a maximal backward (and forward, respectively) interval in S unless otherwise stated. We also use the terms "$(a_1, a_2, \cdots, a_m)$ basic forward b-path," if the exact numbers of the repetitions of $\overleftarrow{P}_b, \vec{P}_b$ need not be specified, or just "basic forward b-path" if the parameters need not be specified. *Basic backward b-paths* are defined similarly. A *basic b-path* is a basic forward or backward b-path. In the following, we allow $n = 1$ in the definition of basic b-path. In fact, a basic b-path for $n = 1$ becomes a directed path.

A $(b + 1, 2m - 1)^n$ *basic b-path* is a basic b-path with parameters $(a_1, 2m_1 - 1; \cdots; a_{n-1}, 2m_{n-1} - 1; a_n)$, where $a_1 = \cdots = a_n = b + 1$ and $m_1 = \cdots = m_{n-1} = m$.

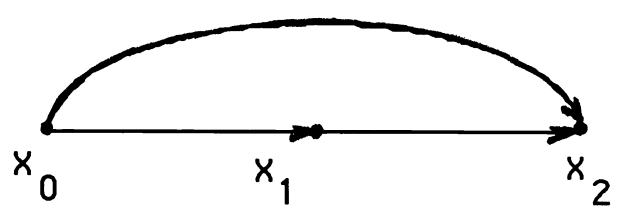


FIG. 2.1

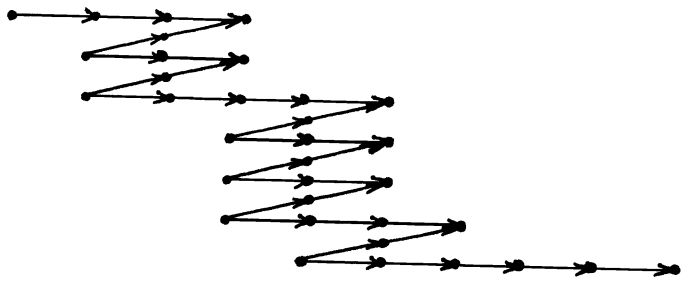


FIG. 2.2. (3, 4, 3, 5) *basic (forward) 2-path or* (3, 3; 4, 5; 3, 1; 5) *basic (forward) 2-path.*

A *degenerate* $(b+1, 2m-1)^n$ *basic b-path* is defined exactly the same way, except that $a_1 = b$ or $a_n = b$ (or both).

We give examples for these definitions in Figs. 2.2–2.3.

A *basic b-cycle* in a graph G is an oriented cycle as follows:

$$v\vec{P}_a\overleftarrow{P}_b\vec{P}_b\cdots\overleftarrow{P}_bv \quad \text{or} \quad v\vec{P}_a\overleftarrow{P}_b\vec{P}_b\cdots\overleftarrow{P}_bW\overleftarrow{P}_b\vec{P}_b\cdots\overleftarrow{P}_bv,$$

where W is a basic forward b-path and $a > b$; in this notation, v is the first vertex of $\vec{P}_a$ as well as the last vertex of the last $\overleftarrow{P}_b$.

The digraph given in Fig. 2.1 is a basic 1-cycle $x_0\vec{P}_2\overleftarrow{P}_1x_0$.

The following three lemmas, whose proofs are given in [18], are important to obtain the Main Theorem.

LEMMA 2.1. *Let* $P_2[y_0, y_1, \cdots, y_k]$ *be a directed path of length* k $(k > 1)$, *and* $P_1[x_0, x_1, \cdots, x_t]$ *a* $(b+1, 2m-1)^n$ *basic forward b-path. Then any component C of* $P_1 \times P_2$ *is isomorphic to a subpath of* P_1 *with net length between* 0 *and* k.

For readers not familiar with [18], we provide the details of various components in the product graph as follows:

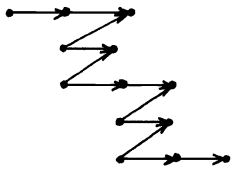
(1) C is an isolated vertex located in $\{(x, y_0): x \in P_1\}$ or $\{(x, y_k): x \in P_1\}$;

(2) C is a directed path $\vec{P}_{b'}$ $(1 \leqq b' \leqq b)$ either starting at some (x_0, y) $(y \in P_2)$ and ending at some (x, y_k) $(v \in P_1)$, or starting at some (x, y_0) $(x \in P_1)$ and ending at some (x_t, y) $(y \in P_2)$;

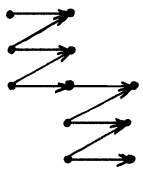
(3) C is $\vec{P}_{b'}\overleftarrow{P}_{b'}$ $(1 \leqq b' \leqq b)$ starting at some (x, y_0) and ending at some (x', y_0) $(x, x' \in P_1)$;

(4) C is $\overleftarrow{P}_{b'}\vec{P}_{b'}$ $(1 \leqq b' \leqq b)$ starting at some (x, y_k) and ending at some (x', y_k) $(x, x' \in P_1)$; or

(5) C is a $(b+1, 2m-1)^{n_1}$ basic or degenerate basic b-path for some n_1 $(2 \leqq n_1 \leqq n)$ starting at some (x, y_0) $(x \in P_1)$ and ending at some (x', y_k) $(x' \in P_1)$, or starting



$(2,3)^3$ basic (forward)1-path



$(2,3)^3$ degenerate basic (forward)1-path

FIG. 2.3. (*Degenerate*) *basic b-path.*

at some (x_0, y) $(y \in P_2)$ and ending at some (x, y_k) $(x \in P_1)$, or starting at some (x, y_0) $(x \in P_1)$ and ending at some (x_t, y) $(y \in P_2)$.

LEMMA 2.2. *Let G be a $(b+1, 2m-1)^n$ basic b-path and W any oriented path. Then there exists a homomorphism from G onto W if and only if W is a basic b-path with parameters $(a_1, 2m_1 - 1; \cdots; a_{t-1}, 2m_{t-1} - 1; a_t)$, where $1 \leqq t \leqq n$, all $m_i \leqq m$, and $nl(W) = nl(G)$.*

LEMMA 2.3. *Let G be a $(b+1, 2m-1)^n$ basic or degenerate basic b-path and W a directed path or a $(a_1, 2m_1 - 1; \cdots; a_{t-1}, 2m_{t-1} - 1; a_t)$ basic b-path with $m \geqq m_i$ for $i = 1, \cdots, t-1$. Assume that $nl(G) \leqq nl(W)$. Then*

(1) $G \to W$;

(2) *We can map the vertex of G with smallest (or greatest) level to the vertex of W with smallest (or greatest) level in the homomorphic mapping of G to W; and*

(3) *If $nl(G) = nl(W)$, then the two end vertices of G will be mapped to the two end vertices of W in the homomorphic mapping of G to W.*

3. Even-deleting operations and Main Theorem. Let C be a core oriented cycle. Let b_0 be the minimum length of any maximal one-directional intervals in C. A *b_0-run* is a subpath of C consisting of alternating consecutive $\vec{P}_{b_0}$ and $\overleftarrow{P}_{b_0}$ (i.e., $\vec{P}_{b_0}\overleftarrow{P}_{b_0}\vec{P}_{b_0}\overleftarrow{P}_{b_0}\cdots$, or $\overleftarrow{P}_{b_0}\vec{P}_{b_0}\overleftarrow{P}_{b_0}\vec{P}_{b_0}\cdots$), where each $\vec{P}_{b_0}$ and $\overleftarrow{P}_{b_0}$ is maximal in C, as we recall from the comments on the notation. A *maximal b_0-run* is a *b_0-run* not included in a larger b_0-run. Any maximal b_0-run must be preceded and followed by a one-directional interval of length greater than b_0 from the minimality of b_0 and the fact that C is a core. If there exists a maximal b_0-run that has an odd number of maximal one-directional intervals, then we stop; i.e., we do not need to do any operation on C. Now suppose that all maximal b_0-runs have an even number of maximal one-directional intervals. Let I be one such maximal b_0-run.

Write $C = vA_1xIyA_2v$. We can perform an *I-deleting operation* on the interval $[x, y]$ of C to obtain an oriented cycle $A = vA_1xA_2v$ (i.e., delete I and identify x and y). Conversely, we can perform an I-*squeezing operation* at x of A to return to C. If we only want to specify the parameter b_0, but do not wish to specify I, we call these two operations a *b_0-even-deleting operation* and a *b_0-even-squeezing operation*. Let C^1 be the oriented path obtained from C after performing b_0-even-deleting operations on all maximal b_0-runs.

Let b_1 be the minimum length of all maximal one-directional intervals in C^1. Clearly, $b_1 > b_0$. Let I be one of the deleted intervals of C during the b_0-even-deleting operation. Then I has the following three properties:

(i) $nl(I) = 0$;

(ii) the level of any vertex of I is between the levels of the two end vertices of either one of the neighboring maximal one-directional intervals in C (for any fixed level function on C); and

(iii) any subpath of I has net length smaller than b_1. (In fact, any subpath of I has net length not greater than b_0.)

Now it is easy to see that $C \to C^1$ by a homomorphism that maps the deleted interval to either one of its neighboring maximal one-directional intervals.

If in C^1 there exists a maximal b_1-run that has an odd number of maximal one-directional intervals, then we stop; i.e., we do not need to do any even-deleting operation of C^1. Now suppose that all maximal b_1-runs have an even number of maximal one-directional intervals. We again do all the b_1-even-deleting operations on C^1. Continuing this process, let the oriented cycle C^k be obtained from C^{k-1} by doing all the b_{k-1}-even-deleting operations, provided all maximal b_{k-1}-runs have an even number of

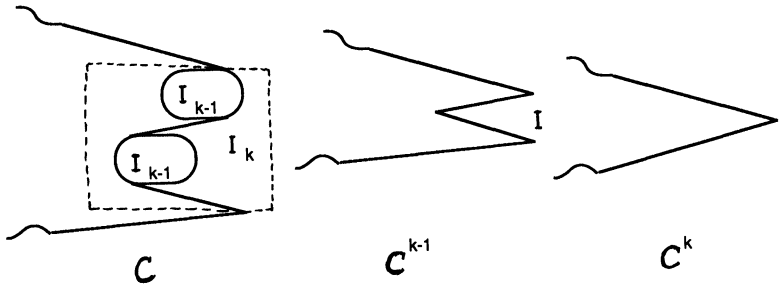


FIG. 3.1

maximal one-directional intervals in C^{k-1}, where b_{k-1} is the minimum length of all maximal one-directional intervals of C^{k-1}. Furthermore, let I^k be one of the intervals that was removed from C in the process of constructing C^k and I the corresponding interval of C^{k-1} (each maximal one-directional interval of I has length b_{k-1}) and let b_k the minimum length of all maximal one-directional intervals of C^k. We can prove by induction on k that I^k has the following three properties (see Fig. 3.1 for an illustration):

(iv) $nl(I^k) = 0$;

(v) the level of any vertex of I^k is between the levels of two end vertices of either one of the neighboring maximal one-directional intervals in C (for any fixed level function of C); and

(vi) any subpath of I^k has net length smaller than b_k.

Let C^k be the first oriented cycle obtained in this process in which there is a maximal b_k-run I with an odd number of maximal one-directional intervals of length b_k, where b_k is the minimum length of all the maximal one-directional intervals of C^k. Therefore C^k contains a basic b_k-path.

Let I^k be any one of the intervals that was removed from C to construct C^k. We have known that I^k satisfies (iv)–(vi). It is easy to see that $C \to C^k$ by a homomorphism that maps each deleted interval to either one of its two neighboring maximal one-directional intervals.

Now we can propose the Main Theorem.

MAIN THEOREM. *Let C be a core oriented cycle. Let C^k be the oriented cycle obtained by doing the above even-deleting operations. If $C^k \neq v_0\vec{P}_{b_k+1}\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}v_0$, then C (as well as C^k) is nonmultiplicative.*

COROLLARY. *Let C be a core oriented cycle and b, a positive integer. If C is not a basic b-cycle and C contains a basic b-path, then C is nonmultiplicative.*

We use this corollary in the next section.

By a totally different and quite complicated technique, we have proved in [11] that if $C^k = v_0\vec{P}_{b_k+1}\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}v_0$, then C is multiplicative. Therefore we have completely characterized the oriented cycles with respect to multiplicativity.

We prove the Main Theorem according to the following two cases in § 4.

Case 1. C^k contains a basic b_k-path, but C^k itself is not a basic b_k-cycle.

Case 2. C^k is a basic b_k-cycle and $C^k \neq v_0\vec{P}_{b_k+1}\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}v_0$.

4. Proof of the Main Theorem.

Case 1. C^k contains a basic b_k-path, but C^k itself is not a basic b_k-cycle.

Let $C^k = v_{k0}U^kv_{k1}W^kv_{k0}$ and $C = v_0Uv_1Wv_0$, where W^k is a $(b_k + 1, 2m - 1; b_k + 1)$ basic b_k-path, and U^k and W^k are obtained from U and W, respectively, by

even-deleting operations. If we need to delete a copy of some I neighboring to v_0 or v_1, then the copy of I is assumed to be in U. Let

$$b^* = \max\{nl(S): S \text{ is a basic } b_k\text{-path of } C^k\}.$$

(Note: we regard a directed path as a basic b_k-path if the length of this directed path is at least b_k.)

We now construct G^k and H^k as follows: G^k is a $(b_k + 1, 2m - 1)^{b^{*+1-b_k}}$ basic b_k-path, and $H^k = d_{k0}U^k d_{k1}\vec{P}_{b_k+2}d_{k0}$, where H^k is obtained from C^k by replacing W^k by $\vec{P}_{b_k+2}$, but now the two end vertices of U^k are denoted by d_{k0} and d_{k1} for our convenience.

If we write

$$W^{k1} = w_1\vec{P}_{b_k}w_2\cdots\vec{P}_{b_k}w_{2m}, \qquad \text{and}$$

$$W^k = \vec{P}_{b_k+1}w_1\vec{P}_{b_k}w_2\cdots\vec{P}_{b_k}w_{2m}\vec{P}_{b_k+1}$$

$$= \vec{P}_{b_k+1}W^{k1}\vec{P}_{b_k+1},$$

where w_i $(i = 1, 2, \cdots, 2m)$ denotes the end vertex of the corresponding $\vec{P}_{b_k}$, then G^k can be written as

$$G^k = \vec{P}_{b_k+1}W^{k1}\vec{P}_{b_k+1}W^{k1}\cdots W^{k1}\vec{P}_{b_k+1} \text{ (there are } (b^* - b_k + 1)\vec{P}_{b_k+1}).$$

Now we can construct G and H as follows. (See Fig. 4.1 for illustration.)

Suppose that the interval I of C was removed in the process of constructing C^k. If the vertex of C^k where this occurred is in U^k, then we insert a copy of I in the corresponding vertex of H^k. If the vertex was some w_i $(i = 1, \cdots, 2m)$, then we insert a copy of I at the corresponding vertex w_i of each copy of W^{k1} in G^k. Note that there are $b^* - b_k$ copies of W^{k1} in G^k.

In this way we obtain graphs G and H. $\vec{P}_{b_k+2}$ is a subpath of H. We write $H = d_0Ud_1\vec{P}_{b_k+2}d_0$, where U corresponds to U^k.

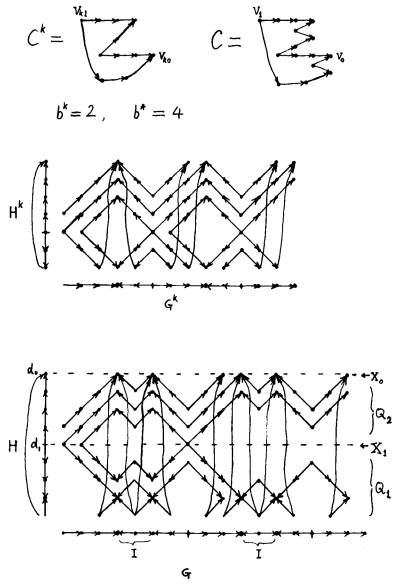


FIG. 4.1. *Illustration of the lemma.*

Claim 1. $H \not\to C$.

Assume on the contrary that $H \to C$. Since we have $C^k \to H^k$, because $U^k \to U^k$ and $W^k \to \vec{P}_{b_k+2}$, and we have inserted the same copies of I at the corresponding vertices of C^k and H^k, we have $C \to H$. The mapping from H to C is not onto since H has fewer vertices than C. Thus the composition of $C \to H \to C$ is a homomorphism of C onto a proper subgraph, contradicting the fact that C is a core.

Claim 2. $G \not\to C$.

Assume on the contrary that f: $G \to C$ is a homomorphism. If we write

$$G^k = \vec{P}_{s_1}\overleftarrow{P}_{s_2}\cdots \quad \text{and} \quad C^k = \vec{P}_{t_1}\overleftarrow{P}_{t_2}\cdots,$$

then

$$G = \vec{P}_{s_1}I_1\overleftarrow{P}_{s_2}I_2\cdots \quad \text{and} \quad C = \vec{P}_{t_1}J_1\overleftarrow{P}_{t_2}J_2\cdots,$$

where the intervals I_i, J_i ($i = 1, 2, \cdots$) satisfy (iv)–(vi) of § 3 and contain only one-directional paths of lengths smaller than all s_i (t_i). The homomorphism f must therefore map each P_{s_i} to some P_{t_j}. Moreover, the images under f of the endpoints of any I_i are the endpoints of some J_m. This ensures that f can be used to define (in the natural way) a homomorphism f^k: $G^k \to C^k$. Since G^k is a $(b_k + 1, 2m - 1)^{b^*+1-b_k}$ basic b_k-path, then the homomorphic image of G^k must be a basic b_k-walk of net length $nl(G^k) = b^* + 1$ by Lemma 2.2. This is impossible.

Claim 3. $G \times H \to C$.

The product $G \times H$ has two parts as follows:

$$Q_1 = G \times d_0 U d_1 \quad \text{and} \quad Q_2 = G \times d_1 \vec{P}_{b_k+2} d_0.$$

Recall that

$$H = d_0 U d_1 \vec{P}_{b_k+2} d_0 \quad \text{and} \quad C = v_0 U v_1 W v_0.$$

Let

$$X_0 = \{(x, d_0) : x \in G\} \quad \text{and} \quad X_1 = \{(x, d_1) : x \in G\}.$$

The projection π defined by $\pi(x, y) = y$ is a homomorphism of $G \times H$ onto H, which maps Q_1 to U and Q_2 to $\vec{P}_{b_k+2}$, with the set X_i mapped to v_i ($i = 0, 1$). Like Lemma 2.1, we have the following lemma [18] to classify the possible components of Q_2. Then we can modify π to a homomorphism $G \times H \to C$ by noting that each component A of Q_2 may be homomorphically mapped to W so that the image of $X_i \cap A$ are v_i ($i = 0, 1$) by Lemma 2.3.

Remark. The proof for the multiplicativity of C^k is exactly the same as the proof for the multiplicativity of C, but instead of using G, H, and the following lemma, we use the simpler form G^k, H^k, and Lemma 2.1. Similarly, in Case 2, we only give the proof for the multiplicativity of C.

LEMMA. *Let* $P_2[y_0, y_1, \cdots, y_{b_k+2}]$ *be a directed path of length* $b_k + 2$, *and* $P_1[x_0, \cdots, x_t]$ *be the graph G constructed as above. Then any component C of* $P_1 \times P_2$ *is isomorphic to a subpath of* P_1 *with net length between* 0 *and* k.

As in Lemma 2.1, for readers not familiar with [18], we provide the detail of various components in the product graph as follows. We also provide the illustrations in Fig. 4.2.

(1) C is an isolated vertex located in $P_1 \times \{y_0\}$ or $P_1 \times \{y_{b_k} + 2\}$;

(2) C is a directed path $\vec{P}_a$ ($1 \leqq a \leqq b_k$) starting at (x_0, y_i) ($i > 1$) and ending at (x, y_{b_k+2}) ($x \in P_1$), or starting at (x, y_0) ($x \in P_1$) and ending at (x_t, y_j) ($j \leqq b_k$);

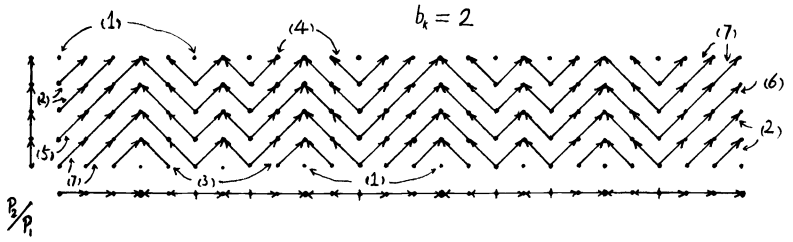


FIG. 4.2. *Illustration of Case* 1.

(3) C is $\vec{P}_a I_i \overleftarrow{P}_a$ $(0 \leqq a < b_k)$ or its subpath for some i with both end vertices at $P_1 \times \{y_0\}$;

(4) C is $\overleftarrow{P}_a I_i \vec{P}_a$ $(0 \leqq a < b_k)$ or its subpath for some i with both end vertices at $P_1 \times \{y_{b_k+2}\}$;

(5) C is $\vec{P}_{b_k+1} I_1 \overleftarrow{P}_{b_k} I_2 \vec{P}_{b_k} \cdots I_{2m} \vec{P}_{b_k}$ starting at (x_0, y_1) and ending at (x, y_{b_k+2}) $(x \in P_1)$;

(6) C is $\vec{P}_{b_k} I_1 \overleftarrow{P}_{b_k} I_2 \cdots I_{2m} \vec{P}_{b_k+1}$ starting at (x, y_0) $(x \in P_1)$ and ending at (x_t, y_{b_k+1}); or

(7) C is $\vec{P}_{s_1} I_1 \overleftarrow{P}_{s_2} I_2 \cdots \vec{P}_{s_m}$ starting at (x, y_0) and ending at (x', y_{b_k+2}) $(x, x' \in P_1)$, where $\vec{P}_{s_1} \overleftarrow{P}_{s_2} \cdots \vec{P}_{s_m}$ is a basic or degenerate basic b_k-path with net length $b_k + 2$ and $s_i = b_k$ or $b_k + 1$ for $i = 1, 2, \cdots, m$.

In (3)–(7) $I_1, I_2, \cdots$ are oriented paths satisfying properties (iv)–(vi) in § 3.

Case 2. C^k is a basic b_k-cycle.

Since $C^k \neq v_0 \vec{P}_{b_k+1} \overleftarrow{P}_{b_k} \vec{P}_{b_k} \cdots \overleftarrow{P}_{b_k} v_0$, then C^k must be in the following two subcases.

Case 2.1. $C^k = v_0 \vec{P}_a \overleftarrow{P}_{b_k} \vec{P}_{b_k} \cdots \overleftarrow{P}_{b_k} v_0$ $(a \geqq b_k + 2)$.

Case 2.2. $C^k = v_0 \vec{P}_a \overleftarrow{P}_{b_k} \vec{P}_{b_k} \cdots \overleftarrow{P}_{b_k} W^k \overleftarrow{P}_{b_k} \vec{P}_{b_k} \cdots \overleftarrow{P}_{b_k} v_0$, where $a > b_k$ and W^k is a nonempty basic forward b_k-path.

First we prove Case 2.1, below.

Proof of Case 2.1. We may write

$$C^k = v_0 \vec{P}_a v_1 \overleftarrow{P}_{b_k} v_2 \vec{P}_{b_k} v_3 \cdots v_{2n-1} \overleftarrow{P}_{b_k} v_0 \ (a \geqq b_k + 2)$$

$$= v_0 \vec{P}_a H^{k1} v_0, \quad \text{where } H^{k1} = v_1 \overleftarrow{P}_{b_k} v_2 \vec{P}_{b_k} v_3 \cdots v_{2n-1} \overleftarrow{P}_{b_k} v_0.$$

Then

$$C = v \vec{P}_a I_1 \overleftarrow{P}_{b_k} I_2 \vec{P}_{b_k} I_3 \cdots I_{2n-1} \overleftarrow{P}_{b_k} I_0 v \ (a \geqq b_k + 2)$$

$$= v \vec{P}_a H^1 v, \quad \text{where } H^1 = I_1 \overleftarrow{P}_{b_k} I_2 \vec{P}_{b_k} I_3 \cdots I_{2n-1} \overleftarrow{P}_{b_k} I_0,$$

where C is the original cycle, and C (as well as H^1) can also be obtained by squeezing I_i at vertices v_i of C^k (and H^{k1}, respectively) for $i = 0, 1, \cdots, 2n - 1$. Construct oriented paths G and H as follows:

$$G = \vec{P}_{a+1}, \qquad H = \vec{P}_a H^1 \vec{P}_{b_k+1} H^1 \vec{P}_a.$$

See Fig. 4.3 for illustration.

Claim. $G \not\to C$, $H \not\to C$, and $G \times H \to C$.

It is obvious that $G \not\to C$ since the length of the directed path G is greater than the length of any maximal one-directional interval of C.

Suppose that $H \to C$. Then the first $\vec{P}_a$ of H must be mapped to the unique $\vec{P}_a$ of C, and the next H^1 of H must be mapped onto the H^1 of C, otherwise the next $\vec{P}_{b_k+1}$ could not be homomorphically mapped to C (cf. above). Then the last vertex of the

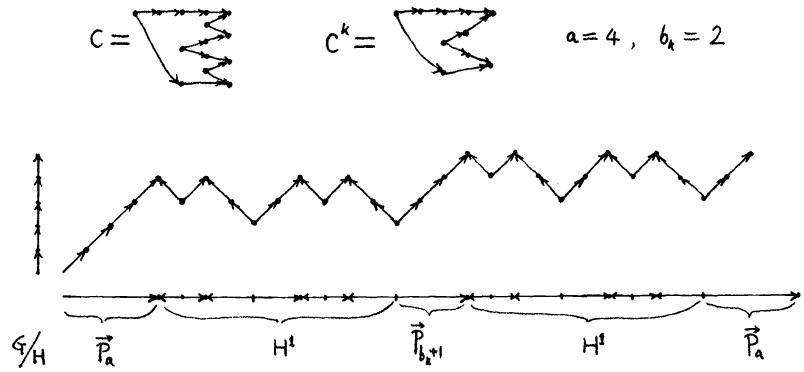


FIG. 4.3. *Illustration of Case* 2.1.

path $\vec{P}_{b_k+1}$ in H must be mapped to some inner vertex of the $\vec{P}_a$ in C, and so must the last vertex of the following H^1 in H. Therefore the last $\vec{P}_a$ in H cannot be properly mapped.

Now we show that $G \times H \to C$. It is easy to see that $nl(H) > a + 1$, any proper subpath of H admits a homomorphism to C and $nl(G') \leqq a + 1$ for any directed subpath G' of G. Let π_1 and π_2 be the projections $G \times H \to G$ and $G \times H \to H$, respectively. For any component A of $G \times H$, it is easy to see that $\pi_1(A)$ is a directed subpath of G and $nl(\pi_2(A)) = nl(\pi_1(A))$ is at most $nl(G) = a + 1$. Therefore $\pi_2(A)$ is a proper subpath of H with net length at most $a + 1$ and so admits a homomorphism to C. By composition, $A \to C$. Therefore $G \times H \to C$.

Proof of Case 2.2. $C^k = v_0\vec{P}_a\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}W^k\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}v_0$, where $a > b_k$ and W^k is a nonempty basic forward b_k-path with parameters $(a_1, 2m_1 - 1; a_2, 2m_2 - 1; \cdots; a_{n-1}, 2m_{n-1} - 1; a_n)$. Without loss of generality, let $a \geqq \max\{a_i: i = 1, 2, \cdots, n\}$.

We may write

$$C^k = v_0\vec{P}_a v_1\overleftarrow{P}_{b_k}v_2\vec{P}_{b_k}v_3\cdots v_{2m-1}\overleftarrow{P}_{b_k}v_{2m}W^k\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}v_0$$

$$= v_0\vec{P}_a H^{k1}W^k\overleftarrow{P}_{b_k}\vec{P}_{b_k}\cdots\overleftarrow{P}_{b_k}v_0,$$

where $H^{k1} = v_1\overleftarrow{P}_{b_k}v_2\vec{P}_{b_k}v_3\cdots v_{2m-1}\overleftarrow{P}_{b_k}v_{2m}$. Now we can construct the directed path G and the basic b_k-cycle H^k as follows:

$$G = \vec{P}_{a+1} \quad \text{and} \quad H^k = v_{2m}\vec{P}_{b_k+1}H^{k1}v_{2m}.$$

Suppose that we can obtain C from C^k by squeezing I_i at v_i for $i = 0, 1, \cdots, 2m$, etc. Then we can construct H^1 and H from H^{k1} and H^k by squeezing I_i at v_i for $i = 0, 1, \cdots, 2m$ as follows:

$$H^1 = I_1\overleftarrow{P}_{b_k}I_2\vec{P}_{b_k}I_3\cdots I_{2m-1}\overleftarrow{P}_{b_k}I_{2m} \quad \text{and} \quad H = h_0\vec{P}_{b_k+1}H^1h_0.$$

See Fig. 4.4 for illustration.

Claim. $G \not\to C$, $H \not\to C$, and $G \times H \to C$.

The reason for $G \not\to C$ and $H \not\to C$ are obvious. The main component of $G \times H$ is the oriented path isomorphic to

$$A = H^1\vec{P}_{b_k+1}H^1\vec{P}_{b_k+1}H^1\cdots H^1\vec{P}_{b_k+1}H^1, \qquad \text{there are } (a - b_k + 1)\vec{P}_{b_k+1},$$

which is obtained by squeezing I_i at v_i, for $i = 1, 2, \cdots, 2m$, of each copy H^{k1} of the following degenerate $(b_k + 1, 2m - 1)^{a-b_k+3}$ basic b_k-path

$$H^{k1}\vec{P}_{b_k+1}H^{k1}\vec{P}_{b_k+1}H^{k1}\cdots H^{k1}\vec{P}_{b_k+1}H^{k1}, \qquad \text{there are } (a - b_k + 1)\vec{P}_{b_k+1}.$$

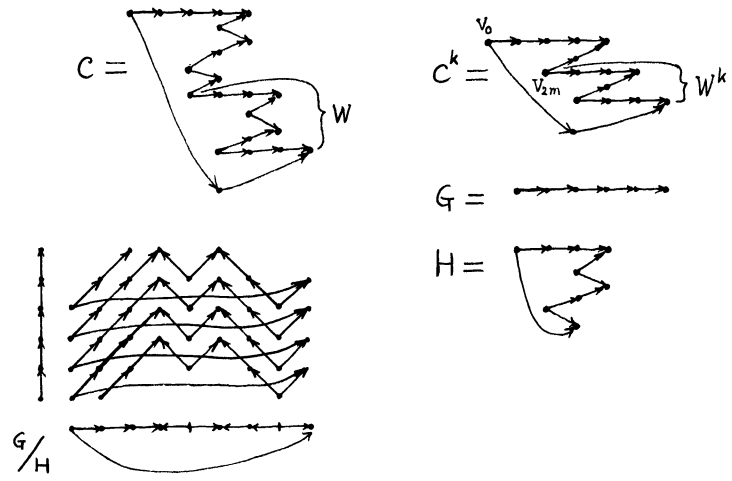


FIG. 4.4. *Illustration of Case* 2.2.

Obviously, A can be homomorphically mapped to

$$v_0\vec{P}_a I_1 \overleftarrow{P}_{b_k} I_2 \vec{P}_{b_k} I_3 \cdots I_{2m-1} \overleftarrow{P}_{b_k} I_{2m} \vec{P}_{b_k+1},$$

which is isomorphic to a subpath of C. Other components of $G \times H$ are $\vec{P}_j$, $\vec{P}_j I \overleftarrow{P}_j$, and $\overleftarrow{P}_j I \vec{P}_j$ or their subpath for $0 < j < b$ and I is some I_i for $i = 1, 2, \cdots, 2m$, which can also be homomorphically mapped to C.

5. Almost all oriented cycles are nonmultiplicative. Let C_n be the undirected cycle with n vertices. We say that almost all oriented cycles have property P, if $\lim_{n\to\infty}(p_n/v_n) = 1$, where p_n is the number of orientations of C_n that have property P, and v_n is the total number of orientations of C_n.

LEMMA 5.1. *Almost all oriented cycles are cores.*

Proof. If an oriented cycle is not a core, then it has net length zero since it has a homomorphically equivalent oriented subpath. The number of arcs directed forward is the same as the number of arcs directed backward in an oriented cycle with net length zero. If n is odd, then each oriented cycle of n vertices is a core since it has net length greater than zero. If $n = 2m$ is even, then the proportion of orientations of C_n with net length zero is $\binom{2m}{m}/2^{2m}$, which goes to zero when m goes to infinity. Therefore almost all oriented cycles are cores. □

LEMMA 5.2. *Almost all oriented cycles contain* $\vec{P}_2\overleftarrow{P}_1\vec{P}_2$. (*Note*: *Here we specify that two* $\vec{P}_2$ *are not necessarily maximal in the cycle.*)

Proof. Consider all orientations of some C_n, and consider $q = \lfloor n/6 \rfloor$ edge-disjoint subintervals $I_1, I_2, \cdots, I_q$ of C_n, each being an undirected path of length five. The proportion of orientations of C_n, which have I_j oriented to be $\vec{P}_2\overleftarrow{P}_1\vec{P}_2$ is $1/2^5 = 1/32$. Therefore the proportion of all orientations of C_n that have none of the I_j oriented to be $\vec{P}_2\overleftarrow{P}_1\vec{P}_2$ is $(31/32)^q \to 0$ as $n \to \infty$. Hence the proportion of all orientations of C_n that contain $\vec{P}_2\overleftarrow{P}_1\vec{P}_n$ somewhere tends to 1, as $n \to \infty$. □

THEOREM 5.3. *Almost all oriented cycles are nonmultiplicative.*

Proof. Apply the corollary of the Main Theorem, Lemmas 5.1 and 5.2, and observe that the proportion of orientations of C_n that are basic 1-cycles goes to zero. (By an argument analogous to the one above, almost all orientations contain $\vec{P}_2\overleftarrow{P}_2\vec{P}_2$ and hence cannot be simple basic 1-cycles.) □

Acknowledgment. I thank Professor Pavol Hell for many valuable suggestions and comments.

REFERENCES

[1] D. DUFFUS, B. SANDS, AND R. WOODROW, *On the chromatic number of the product of graphs*, J. Graph Theory, 9 (1985), pp. 487–495.
[2] M. EL-ZAHAR AND N. W. SAUER, *The chromatic number of the product of two* 4*-chromatic graphs is* 4, Combinatorica, 5 (1985), pp. 121–126.
[3] M. FARZAN AND D. A. WALKER, *Kronecker products and local joins of graphs*, Canad. J. Math., 29 (1977), pp. 255–269.
[4] W. D. FELLNER, *On minimal graphs*, Theoret. Comp. Sci., 17 (1982), pp. 103–110.
[5] A. M. H. GERARDS, *Homomorphisms of graphs into odd cycles*, J. Graph Theory, 12 (1988), pp. 73–83.
[6] R. HAGGKVIST, P. HELL, D. J. MILLER, AND N. LARA, *On multiplicative graphs and the productive conjecture*, Combinatorica, 8 (1988), pp. 71–81.
[7] F. HARARY, *Graph Theory*, Addison-Wesley, Reading, MA, 1972.
[8] S. T. HEDETNIEMI, *Homomorphisms of graphs and automata*, University of Michigan Tech. Report 03105-44-T, 1966.
[9] P. HELL, *An introduction to the category of graphs*, Ann. N.Y. Acad. Sci., 328 (1979), pp. 120–136.
[10] ———, *Retracts in graphs*, Lecture Notes in Math., 406 (1974), pp. 291–301.
[11] P. HELL, X. ZHU, AND H. ZHOU, *On the multiplicativity of oriented cycles*, submitted.
[12] P. KOMAREK, *Some new good characterizations for directed graphs*, Časopsis Pëst. Mat., 51 (1984), pp. 348–354.
[13] D. J. MILLER, *The categorical product of graphs*, Canad. J. Math., 20 (1968), pp. 1511–1521.
[14] J. NESETRIL AND A. PULTR, *On classes of relations and graphs determined by subobject and factorobjects*, Discrete Math., 22 (1978), pp. 297–300.
[15] S. POLJAK AND V. RODL, *On the arc-chromatic number of a digraph*, J. Combin. Theory Ser. B, 31 (1981), pp. 190–198.
[16] P. M. WEICHSEL, *The Kronecker product of graphs*, Proc. Amer. Math. Soc., 13 (1962), pp. 47–52.
[17] H. ZHOU, *Multiplicativity (part* I*): variations and multiplicative graphs*, J. Graph Theory, to appear.
[18] ———, *Multiplicativity (part* II*): non-multiplicative digraphs and characterization of oriented paths*, J. Graph Theory, to appear.
[19] X. ZHU, *Multiplicative structures*, Ph.D. thesis, University of Calgary, Calgary, Alberta, Canada, 1990.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 219–248, May 1992
© 1992 Society for Industrial and Applied Mathematics
006

NORMAL LIMITING DISTRIBUTIONS FOR PROJECTION AND SEMIJOIN SIZES*

DANIÈLE GARDY†

Abstract. This paper presents classes of bivariate generating functions associated with the probability distributions of parameters on sets of points (sizes of derived relations in a relational data base) that correspond to asymptotically normal distributions. These results are extended to give some conditions under which the numbers $a_{n,k}$ defined by $\sum_{n,k} a_{n,k}x^k y^n = \phi(x,y)^d$ follow a Gaussian limiting distribution.

Key words. central limit theorem, generating function, urn models

AMS(MOS) subject classifications. 05A16, 60F05, 68P15, 68Q25

1. Introduction.

The aim of this paper is to study some parameters that can be defined on sets of points obtained by random sampling without replacement from an initial domain Δ. We examine the probability distribution of these parameters, under the hypothesis that the size of the domain Δ and the sample size grow to infinity. More precisely, we assume that we know the probability distribution on Δ, and we want to show that, for a large class of these distributions, the probability distributions of the parameters that we consider are asymptotically normal.

Our tools for proving this convergence are the multivariate generating function of the parameter under study and the size(s) of the set(s) of points, and classical results on analytic functions and the Laplace transform. Bivariate generating functions for which the convergence toward a normal distribution holds are studied, for example, in [2], [4], [8]. Our approach differs from these works mostly in that the generating functions considered here depend on d, one of the parameters that grow to infinity. We consider *probability* or *counting* generating functions, which are themselves of the form "dth power of a function."

The plan of the paper is as follows: We present the parameters that we intend to study in §2. There we give several interpretations of these parameters, both probabilistic and related to data bases in computer science. We formally introduce our modelization and notations in §3. We next give our results in §§4 and 5 and prove them in §6.

2. Sets of points, relations, and sums of random variables.

2.1. Sets of points.

We first define the set Δ of "legal" points. A point in a two-dimensional space is an ordered pair (x, y). We assume that each coordinate takes its value in a finite domain, denoted, respectively, by D_X and D_Y, on which a probability distribution is defined. Throughout the paper, d_X represents the size of the set D_X, and d_Y the size of the set D_Y. We also assume that the values of a point on its first and second coordinates are independent; i.e., the probability distribution on $\Delta = D_X \times D_Y$ is the product of the probability distributions on D_X and D_Y.

We consider a random subset R of Δ built in one of the following ways:

* Received by the editors December 2, 1988; accepted for publication (in revised form) April 30, 1991. This work was partially supported by the Programme de Recherches Concertées Mathématique-Informatique and by ESPRIT-II Basic Research Action No. 3075 (project ALCOM).

† LRI, Université de Paris-Sud, Centre National de la Recherche Scientifique U.A. 410, 91405 Orsay, France.

- In the first case, we obtain R by drawing n independent random points without replacement from Δ;
- We may also draw n independent random samples without replacement from D_X, then complete each pair by drawing independently the y-value from D_Y. In this case, each value of D_X appears at most once in a set R, but a given value of D_Y may be present in several pairs of R.

Of course, the symmetrical rule also exists: We can draw a sample without replacement from D_Y, then complete it by sampling from D_X. We define on R a first parameter $f(R)$: "number of distinct x-values" (or "number of distinct y-values"). The second parameter $g(R,S)$ in which we are interested is the size of the set of points obtained by drawing two independent sets R and S, then suppressing from R all the points (x,y) whose value x on the first coordinate does not appear in a pair of S. The sets R and S may take their values in the same sample space $D_X \times D_Y$ or in two different spaces $D_X \times D_Y$ and $D_X \times D_U$.

We want to investigate the relationship between the *size* of R (number of points in R) and $f(R)$, and between the sizes of R and S and $g(R,S)$, for different probability distributions on the domains D_X, D_Y, and D_U. More precisely, we are interested in the conditional probability distribution of $f(R)$ for a given size of R, and in that of $g(R,S)$ for given sizes of R and S. We study these distributions when the size of the domain D_X and the numbers of samplings (sizes of the sets of points R and S) grow to infinity, and we show that they become asymptotically normal in many cases.

2.2. Relational data bases and sizes of relations. Those who are familiar with that part of computer science that deals with relational data bases may have noticed that the sets of points R and S defined in §2.1 are instances of relations of a particular type. The coordinates X and Y, or X and U, are the so-called *attributes* of relations R and S, and the points are the tuples of the relations. The parameters $f(R)$ and $g(R,S)$ are, respectively, the sizes of the *projection* of relation R on attribute X (or attribute Y) and of the *semijoin* of relations R and S on attribute X. These sizes are important parameters in query optimization, which aims at minimizing the cost of executing a query on the data base. We refer to [18], [20] for general texts on relational data base theory, to [15], [19] for surveys on query optimization and on the evaluation of relation sizes, and to [9], [11], [12] for a complete presentation of the problem of relation sizes and its modelization in terms of generating functions. We mention in [12] that the probability distributions of the sizes of relations obtained by a projection or a semijoin were (empirically) found to follow asymptotically normal distributions. Here we make precise the conditions under which this convergence holds and give the mathematical proofs. We also prove that complete knowledge of the probability distributions on all attributes is not necessary to characterize the asymptotic distribution of the derived size, and that it often suffices to know the distribution on the domain of the attribute on which the projection or the semijoin takes place.

The classical operations defined on relational data bases are the set operations (intersection, union, and symmetrical difference), the projection, and several types of join, mostly the equijoin and the semijoin [18], [20]. We restrict this paper to the *projection* and *semijoin*. The *intersection* is related to a special case of semijoin, and the sizes of the *union* and *difference* are very easily computed from the sizes of the intersection and of the initial relations. We do not consider the *equijoin* in this paper. One justification is that query optimizers often use a sequence of semijoins to reduce data before computing a "full" equijoin, and that an important part of

the cost of the operation comes from the semijoin part. We must also admit that the generating functions associated with the equijoins are less easy to study than the functions associated with the semijoins, and we defer them to a forthcoming paper [10].

We assume that the relations we consider have two (sets of) attribute(s): X and Y or U. Throughout the paper, X denotes the join or projection attribute. We restrict ourselves to the following three schemes of relations:

- In the case of a *free* relation, there is total independence between the values taken by the different tuples. This is the first case of §2.1;
- We may also consider relations where attribute X is a *key*, i.e., in a given instance of the relation the x-value of a tuple uniquely determines its y-value. This is the second case of §2.1;
- Finally, we consider the symmetrical case, where attribute Y is key of relation R.

Of course, there are many more possible schemes of relations. We give in [9], [12] generating functions for several of them.

2.3. Sums of random variables. It can be recognized that the parameters $f(R)$ and $g(R,S)$ defined in §2.1 are instances of a common problem: We study the limiting distribution of a sum of identically distributed *dependent* random variables when the number of variables grows to infinity.

Given two sets R and S built as described in §2.1, we define two random variables for each i in D_X: v_i and w_i are, respectively, the number of points of R or S whose value on the first coordinate is i. These variables take their values in $\{0 \cdots d_Y\}$, and the case where i does not appear in a pair of, say, R corresponds to $v_i = 0$. The sizes of R and S can be expressed as

$$\sum_{1 \le i \le d_X} v_i \quad \text{and} \quad \sum_{1 \le i \le d_X} w_i.$$

The size of the projection of R on the first coordinate is

$$\sum_{1 \le i \le d_X} u_i, \quad \text{with } u_i = 1_{v_i > 0}.$$

The size of the semijoin of relations R and S can also be written as

$$\sum_{1 \le i \le d_X} u_i', \quad \text{with } u_i' = v_i \cdot 1_{w_i > 0}.$$

If we assume a uniform probability distribution on the domain D_X, then the random variables $u_i, 1 \le i \le d_X$ (or the u_i') follow an identical distribution. Our problem, then, is to study the sum of the u_i, or the sum of the u_i', under the conditions that the sums of the v_i and w_i are known and when the total number d_X of variables grows to infinity.

In the case of *independent* random variables u_i, the central limit theorem, or some extensions of it when the variables are not uniformly distributed (see, for example, [13]), allows us to prove that the distribution of the sum $\sum_{1 \le i \le n} u_i$ is asymptotically normal for large n. We see here that, although the random variables are no longer independent, the correlation between them is weak enough that the limiting distribution is still Gaussian.

3. Models and notations.

3.1. Probability distributions on attribute domains. We consider two classes of distributions on a finite domain D of size d and denote by $p_{i,d}$ the probability that the ith element of the domain is selected when choosing at random an element of D. Hence we have that $\sum_i p_{i,d} = 1$. The subscript emphazises the fact that the probability distribution depends on the number d of elements in the domain. Without loss of generality, we can assume that the $p_{i,d}$ are decreasing when i grows, for fixed d. The two classes are defined as follows:

(Z) $\sum_{1 \le i \le d} {p_{i,d}}^2 \to 0$ for $d \to +\infty$;

(G) For each fixed i, $p_{i,d} \to p_i$ for $d \to +\infty$, and the $\{p_i\}$ define a probability distribution.

Class (Z) is named after the *Zipf* distribution: $p_{i,d}$ proportional to $1/i^C$ for $1 \le i \le d$ and fixed d. The uniform probability distribution, the so-called "80% – 20%" distributions and Zipf distributions for $0 < C \le 1$ are members of this class. A probability distribution on a domain of fixed size d_0 ($p_{i,d} = p_{i,d_0}$ for $i \le d_0$ and $p_{i,d} = 0$ for $i > d_0$), Zipf distributions for $C > 1$, and geometric distributions belong to class (G). Intuitively, distributions of class (Z) are not too far from the equiprobable case, and distributions of class (G) are those for which the probability of the "diagonal" $\{(i,i)\}$ has a nonnull limit.

Distributions of classes (Z) and (G) share the uniform convergence, for bounded t, of the generating function $\lambda(t) = \prod_{1 \le i \le d}(1 + p_{i,d}t)$ associated with the probabilities of the sets of distinct items, toward a function $\varphi(t)$.[1] Probability distributions of class (Z) are simply characterized by $\varphi(t) = e^t$. Anticipating the results presented below, we can see that the distributions on the attributes that do not participate in the projection or in the join (attributes Y and U) matter only as long as they belong either to class (Z) or to class (G). In particular, all distributions of class (Z) give distributions for the projection or semijoin size that converge asymptotically to the same normal distribution, uniquely characterized by its moments.

3.2. Probability distributions on relations. We recall the independence assumptions of §2.1, translated in terms of the following relations:

(i) The two coordinates of a tuple (point) are independent;

(ii) The tuples (points) of a given relation (set) are independent, as far as this is compatible with the constraints on the relation (free relation or relation with a key);

(iii) When we consider two relations R and S, these two relations are independent.

Condition (i) ensures that the probability distribution on a domain $\Delta = D_X \times D_Y$ is the product of the probability distributions on domains D_X and D_Y, and condition (iii) merely states that the probability distribution of a couple (R, S) is the product of the probabilities of R and S. Condition (ii) was detailed in §2.1 and deserves further explanation.

The underlying idea is that the probability distribution on a relation R is proportional to the probability of each of its points: $\mathrm{Prob}(R) = k \cdot \prod_{t \in R} \mathrm{Prob}(t)$. The constant k is independent of R and is chosen to obtain a probability distribution on relations; it varies according to the rule for building R, which may restrict the set of

[1] The generating function that gives the probabilities of the finite sets of elements is actually $\lambda_0(t) = \prod_{1 \le i \le d} \big((1 + p_{i,d}t)/(1 + p_{i,d})\big)$. It differs from the function $\lambda(t)$ that we use in the paper by a constant multiplicative factor $\prod_{1 \le i \le d}(1 + p_{i,d})$, which disappears when we study conditional distributions; see §3.3.

admissible relation instances. For example, assume that the probability distribution on attribute X is given by $\{q_j, 1 \leq j \leq d_X\}$ and that the distribution on attribute Y is given by $\{p_i, 1 \leq i \leq d_Y\}$; in the case of a free relation (first case of §2.1), we have that $k = 1/\prod_{i,j}(1 + p_i q_j)$; in the case of a relation with key X (second case of §2.1), we have that $k = 1/\prod_j(1 + q_j)$ [12].

3.3. Limiting distributions. We recall that we want to investigate the limiting distribution of the size of a relation obtained either by the projection of a relation of known size r or by the join of two relations of known sizes r and s. Thus the problems presented in this paper can be cast into a common frame: given a doubly indexed sequence of real positive numbers $(a_{l,r})$,[2] our goal is to study the limiting distribution of the normalized sequence $(b_{l,r} = a_{l,r}/(\sum_l a_{l,r}))$ when r goes to infinity. We assume that we know the function $\Phi(x, y) = \sum_{l,r} a_{l,r} x^l y^r$. The problem can be reformulated using the probability distribution defined by the generating function $f(x) = [y^r]\Phi(x, y)/[y^r]\Phi(1, y)$: This is the conditional probability distribution of the parameter "marked" by x in Φ, knowing that the parameter "marked" by y in Φ (usually the size of some structure) has value r. We study the limit of this conditional distribution when r and d go to infinity.

For example, we define $a_{l,r}$ as the number of relations of size r whose projection is of size l, and we want to estimate the size of the projection of a relation of known size r; d is the size of the domain on which we project the relation. We see in §4 that the generating function that appears in the study of the projection size has the general form $\Phi(x, y) = (1 - x + x\lambda(y))^d$.

In this form, it is obvious that, at least for uniform distributions on the underlying domains, it does not matter if Φ is a probability or counting generating function in the variable y: This corresponds to an extra factor in the term $[y^r]\Phi(x, y)$, which cancels in $f(x)$. In our example, the generating function for the projection sizes might be a probability generating function with respect to x, and a counting generating function with respect to y. For the same reason, we may indifferently use an ordinary or exponential function in y, according to the underlying structure (this holds even if the distribution on the attribute domains are not uniform). Finally, we use probability generating functions Φ either for joint probabilities or for conditional probabilities (we assume that we know the size of the parameter marked by y) as the need arises: The generating function for the conditional probability satisfies $[y^r]\Phi(1, y) = 1$, which gives $f(x) = [y^r]\Phi(x, y)$.

We give our theorems in the case where *the probability distribution on the domain D_X of the projection or join attribute* X *is uniform,* and its size d_X [3] is related in a simple way to the sizes of the relevant relations. For example, in Corollary 1 the size d_X of D_X is of the order of the size r of relation R. We also assume that, when the sizes d_Y and d_U of the domains D_Y and D_U grow to infinity, they do so without relation to d_X. However, the exact relation between these parameters is not strict: The proofs can be adapted in many cases to show the convergence toward normal distributions with suitably modified moments.

3.4. Analytic functions with positive coefficients. For easy reference, we introduce here a property relative to an analytic function that we need to prove

[2] The numbers $a_{l,r}$ actually depend on a third parameter d in such a way that the function $\Phi(x, y) = \sum_{l,r} a_{l,r} x^l y^r$ is of the form $\phi(x, y)^d$. See §§4 and 5.

[3] This is the parameter d such that $\Phi = \phi^d$.

our results and which is satisfied in all the cases studied in this paper; we call it Property $\mathcal{P}$, shown below.

PROPERTY $\mathcal{P}$. *A function, say $\lambda(y)$, is entire and not affine, with positive coefficients, such that $\lambda(0) = 1$, and such that there exists no entire function Λ and integer $m \geq 2$ with $\lambda(y) = \Lambda(y^m)$.*

The last part of Property $\mathcal{P}$, $\lambda(y) \neq \Lambda(y^m)$, is introduced for technical reasons, but is in no way a restriction: If $\lambda(y) = \Lambda(y^m)$, we just change the variable y into y^m for the greatest such m, and the function Λ satisfies Property $\mathcal{P}$. It can be reformulated as in [4]: The greatest common divisor of the $\{r : [y^r]\lambda \neq 0\}$ is 1. Likewise, the important condition on $\lambda(0)$ is simply $\lambda(0) \neq 0$; requiring that $\lambda(0) = 1$ merely simplifies some computations.

In Theorem 1 of §4 and in Theorems 3 and 4 of §5, we use an auxiliary function $g(y)$, obtained from the function $\lambda(y)$ by $g(y) = y\lambda'(y)/\lambda(y)$.

LEMMA A. *Let Property $\mathcal{P}$ be satisfied and define $g(y) = y\lambda'(y)/\lambda(y)$. Then g is increasing on the interval $[0, +\infty[$.*

Proof of Lemma A. As function λ is entire with positive coefficients and $\lambda(0) = 1$, λ has no zeros on $[0, +\infty[$, and the function g is well defined on this interval. Let us define the function

$$D(y) = \lambda(y)(\lambda'(y) + y\lambda''(y)) - y\lambda'^2(y).$$

We have that $g'(y) = D(y)/\lambda^2(y)$. The definition of D in terms of λ and its derivatives can be used to get an expansion of D as a series with positive terms. This shows that g' is positive on the interval $[0, +\infty[$ and that g is increasing on this interval. □

When the function λ satisfies Property $\mathcal{P}$, then, by Lemma A, the function g is increasing; hence $g(y)$ either has a finite limit or tends to infinity when $y \to +\infty$. Henceforth, we use the expression $\lim_{y\to+\infty} g(y)$ either for a finite or an infinite limit; in the last case, the condition that the limit is greater than some positive number A is trivially satisfied.

4. Asymptotic distributions for projection sizes. Given a relation R with two attributes X and Y, we want to study the size of the *projection of* R *on attribute* X. We recall that this projection is computed by suppressing the attribute Y, then eliminating the redundant values of attribute X: We just keep one instance of each x-value that appears in the initial relation. We assume that the domains D_X and D_Y, where X and Y take their values, are of finite sizes d_X and d_Y and that the relation has r elements, where r is of the order of d_X. We are interested in the probability distribution of the size of the projection of R, conditioned by the initial size r of R, when the parameters r, d_X, and d_Y grow to infinity.

To be consistent with the schemes of relations defined in §2.1, we study a relation with a key on the attribute Y eliminated by the projection and a relation without a key. The case of the projection of a relation R with a key on attribute X is without any difficulty: Each pair of R has for X-component a distinct value; as a consequence, the projection on attribute X is composed of all the values x that appear as the first coordinate of a pair (x, y), counted once, and has exactly the same size as the initial relation R.

4.1. R has Y as key. Define $p(l/r)$ as the conditional probability that the projection of R on attribute X is of size l when the size of R is itself equal to r, for a uniform probability distribution on attribute X and a general probability distribution on attribute Y given by $\{p_{i,d_Y}, 1 \leq i \leq d_Y\}$. To study the distribution

of the projection size, it is convenient to use the following generating function, exponential in y: $\Phi(x,y) = \sum_{l,r} p(l/r)x^l y^r/r!$. As an intermediate step, we use the auxiliary bivariate generating function $\Psi(x,y) = \sum_{l,r} p(l,r)x^l y^r$. In both functions Φ and Ψ, the variable x marks the size of the projection on attribute X, and the variable y the size of the initial relation; $p(l,r)$ is the joint probability that relation R is of size r and its projection on attribute X is of size l; it is related to $p(l/r)$ by $p(l/r) = p(l,r)/(\sum_k p(k,r)) = [x^l y^r]\Psi(x,y)/[y^r]\Psi(1,y)$. We have [9], [11] that

$$\Psi(x,y) = \sum_{k=0}^{d_X} \binom{d_X}{k} \lambda_0(ky/d_X)\, x^k\, (1-x)^{d_X-k}.$$

In this formula, $\lambda_0(t) = \prod_{1\le i\le d_Y} ((1+p_{i,d_Y}t)/(1+p_{i,d_Y}))$ is the generating function describing all sets of y-values, with their associated probability. By extracting the coefficient of y^r in Ψ and computing the (exponential in y) generating function of the conditional probabilities $\Phi(x,y) = \sum_{l,r} p(l/r)x^l y^r/r!$, we get [11] that

$$\Phi(x,y) = (1 + x(e^{y/d_X} - 1))^{d_X}. \tag{1}$$

Let us mention that there exists a closed-form expression for the conditional probabilities: $p(l/r) = l!\binom{d_X}{l} d_X^{-r} S(r,l)$, with $S(r,l)$ a Stirling number of the second kind.[4]

Equation (1) shows that the evaluation of the projection size for a relation, with a key on the attribute Y suppressed by the projection, is equivalent to the classical *occupancy problem in urn models* [16]. This problem can be summarized as follows: Given d urns and r balls, the balls are thrown independently and at random into the urns, and we study the number of empty urns, or, equivalently, the number of urns containing at least one ball. The appropriate generating function in this case is a counting generating function, exponential in the number of balls (marked by the variable y) and ordinary in the number of urns with at least one ball (marked by x). Let us denote by $N_{l,r}$ the number of ways of throwing r balls into l urns, with each urn containing at least one ball; then [16] it follows that

$$\sum_{l,r} N_{l,r} x^l y^r/r! = (1 - x + xe^y)^d.$$

It is obvious from the expression of $\Phi(x,y)$ in (1) (but not from that of Ψ) that the probability distribution on attribute Y does not matter. Moreover, a whole spectrum of limiting results is known for urn models (see [16], [17] for surveys) and can be directly applied to the projection of a relation with a key.

4.2. R is a free relation. The bivariate generating function $\Phi(x,y)$ of the joint probabilities $p(l,r)$, where x marks the size of the projection on attribute X and y the size of the initial relation, is [9], [11]

$$\Phi(x,y) = \sum_{l,r} p(l,r)x^l y^r = (1 - x + x\lambda(y))^{d_X},$$

[4] As the notation for Stirling numbers is not standardized, we use here the notation of Comtet [5].

R	$\Phi(x,y)$	Asymptotic result
$X \dagger Y$	$(1-x+x\lambda(y))^{d_X}$	§4.2, Cor. 1
$Y \to X$	$(1-x+xe^{y/d_X})^{d_X}$	[16], [17] or §4.2, Thm. 1

FIG. 1. *Generating function for the size of the projection* $\pi_X(R)$ *of* R *on attribute* X : $\pi_X(R) = \{x | \exists y : (x,y) \in R\}$.

with

$$\lambda(y) = \prod_{1 \le i \le d_Y} (1 + p_{i,d_Y}\, y).^5$$

In this formula, as in §4.1, p_{i,d_Y} denotes the probability of the ith value of domain D_Y, which depends on the type of distribution and on the size of the domain.

Figure 1 sums up the generating function and the asymptotic results, either previously known or proved in this paper, for the two types of relations: a free relation and a relation with a key. Here and in Fig. 2 in §5.1, "X † Y" means that neither attribute X nor attribute Y is key of R (free relation), $Y \to X$ means that the attribute Y is key of R, and $X \to Y$ (in Fig. 2) means that X is key of R.

We first give a general theorem (Theorem 1) pertaining to functions that have the general form $(1-x+x\lambda(y))^d$. We then deduce from it a corollary dealing with the case of a free relation. Theorem 1 can also be used to get the classical result on urn models, or, equivalently, the result pertaining to a relation where attribute Y is key: This is simply the case where the function $\lambda(y)$ is equal to e^y or to e^{y/d_X}.

THEOREM 1. *Let Property* $\mathcal{P}$ *be satisfied. Define* $\Phi(x,y) = (1-x+x\lambda(y))^d$. *Let* $d, r \to +\infty$ *in such a way that* $r = Ad + o(d)$ *for some positive constant* A, *and that* $g(y) = y\lambda'(y)/\lambda(y)$ *satisfies* $\lim_{y \to +\infty} g(y) > A$. *Then the probability distribution defined by the generating function* $f(x) = [y^r]\Phi(x,y)/[y^r]\Phi(1,y)$ *is asymptotically Gaussian when* $d \to +\infty$. *The asymptotic values of the mean and variance are defined in terms of the unique real positive solution* ρ *of the equation* $g(y) = A$ *as follows* :

$$\mu = d\left(1 - \frac{1}{\lambda(\rho)}\right); \qquad \sigma^2 = d\left(\frac{1}{\lambda(\rho)} - \frac{1}{\lambda^2(\rho)} - \frac{\rho\lambda'^2(\rho)}{g'(\rho)\lambda^4(\rho)}\right).$$

COROLLARY 1. *Let* $R[X,Y]$ *be a free relation with a uniform probability distribution on the domain of attribute* X *Then the probability distribution of the size of the projection of* R *on attribute* X, *conditioned by the size* $r = Ad_X + o(d_X)$ *of relation* R

[5] Actually, $\Phi(x,y)$ is obtained from the function

$$\lambda_0(t) = \lambda(t)/\lambda(1) = \prod_i \left((1 + p_{i,d_Y} y)/(1 + p_{i,d_Y})\right)$$

by marking the tuples of R and their projection on X; this gives

$$\Phi(x,y) = \left((1-x+x\prod_i (1+p_{i,d_Y} y))/\prod_i (1+p_{i,d_Y})\right)^{d_X} = \frac{(1-x+x\lambda(y))^{d_X}}{\lambda(1)^{d_X}}.$$

As we are interested in $f(x) = [y^r]\Phi(x,y)/[y^r]\Phi(1,y)$, the multiplicative factor $\lambda(1)^{d_X}$ cancels in $f(x)$, and we can use the simpler expression given in the text.

with A a positive constant is asymptotically normal when $d_X \to +\infty$. The asymptotic mean and variance are given by $\mu = \mu_0\, d_X$ and $\sigma^2 = \sigma_0^2 d_X$ where μ_0 and σ_0^2 are constants that depend on the probability distribution on attribute Y. We now assume that $d_Y \to +\infty$ and is independent of d_X.

If the distribution on D_Y satisfies hypothesis (Z), *then $\mu_0 = 1 - e^{-A}$ and $\sigma_0^2 = (e^A - 1 - A)/e^{2A}$. If the distribution on D_Y satisfies hypothesis* (G), *let $\varphi(t) = \prod_{i\geq 1}(1 + p_i t)$, where the $\{p_i\}$ define the limiting distribution on attribute Y, and $g(t) = t\,\varphi'(t)/\varphi(t)$. Let ρ be the unique real positive solution of the equation $g(t) = A$. The constants μ_0 and σ_0^2 are*

$$\mu_0 = 1 - 1/\varphi(\rho); \qquad \sigma_0^2 = \frac{\mu_0}{\varphi(\rho)} - \frac{\rho\varphi'^2(\rho)}{g'(\rho)\varphi^4(\rho)}.$$

Moreover, μ_0 satisfies $1 - e^{-A} \leq \mu_0 \leq 1$.

The proof of Theorem 1 and the derivation of Corollary 1 are postponed until §6.2. As an application of Corollary 1, we deduce that the exact probability distribution on attribute Y has no influence on the limiting distribution as long as it stays in class (Z) and $d_Y \to +\infty$.

Relation to some urn models. When the probability distribution on the domain of attribute Y belongs to class (Z), i.e., when the function

$$\lambda(y) = \prod_{1\leq i\leq d_Y} (1 + p_{i,d_Y} y)$$

has for limit e^y for any fixed y and for $d_Y \to +\infty$, the generating function $\Phi(x, y) = (1 - x + x\prod_{1\leq i\leq d_Y}(1 + p_{i,d_Y} y))^{d_X}$ converges pointwise toward the function $(1 + x(e^y - 1))^{d_X}$ when d_Y grows to infinity and d_X is constant. This function is the generating function $\sum_{i,j} N_{i,j} x^i y^j / j!$ of the number i of urns containing at least one ball when we throw j balls independently in d_X urns, and it has already appeared in the study of a relation with a key (see §4.1). This can be explained intuitively as follows: For large d_Y, the probability $\sum_i p_{i,d_Y}^2$ that we twice draw the same point in successive trials with replacements is close to zero. Hence we may assume that the successive trials that give the points of the relation are "asymptotically" independent, and we get the classical urn model.

However, when the probability distribution on attribute Y belongs to class (G), the successive trials giving the points of the relation are *not* independent: The probability of twice drawing the same point in random sampling with replacement is definitely not null! (Asymptotically, it is close to $\sum_{i\geq 1} p_i^2 > 0$.) This is reflected in the limiting generating function $(1 + x(\varphi(y) - 1))^{d_X}$, which we obtain by letting d_Y grow to infinity and by keeping d_X constant.

Alternatively, the size of the projection on attribute X can be related to the number of nonempty urns when we throw the balls in *complexes*. Again, we refer to [17] for asymptotic results when complexes are of fixed size. In our approach, a complex is the number of points (x_0, y) in a given instance of a relation for a fixed value x_0 of attribute X, and its size is a random variable taking its values in $\{0 \cdots d_Y\}$. Ammann [1] studies such a case when the size of a complex is bounded and for various conditions on the numbers r of balls and d of urns: If r is of order $\sqrt{d}$, the number of empty urns asymptotically follows a compound Poisson distribution; for larger r, but still with $r = o(d)$, the asymptotic distribution becomes Gaussian. In our framework, this means that the size of domain D_Y is fixed and that the order of the size of the relation is either $\sqrt{d_X}$ or $o(d_X)$.

5. Asymptotic distributions for semijoin sizes. In this section, we consider two initial relations R and S and their semijoin on a common attribute X. The values taken by the two relations are assumed to be independent of each other. We consider that R and S are each built on two attributes, respectively, $R[X,Y]$ and $S[X,U]$. The *semijoin* of R and S on attribute X is a subset of the relation R; it is computed by keeping in relation R only those tuples whose value on X appears in the X-column of relation S. This operation is not symmetrical: The semijoin of R and S is *not* equal to the semijoin of S and R. We recall that we assume a uniform probability distribution on the join attribute X.

5.1. Generating functions. We use the following notation: $p(t,r,s)$ is the joint probability that the relations R and S have respective sizes r and s and that their semijoin is of size t; $p(t,s/r)$ is the joint probability that the relation S is of size s, and that the semijoin of R and S is of size t, conditioned by the fact that the relation R is of size r; and so forth. These probabilities are trivially related to one another; for example, $p(t/r,s) = p(t,r,s)/(\sum_i p(i,r,s))$. We define two functions λ_R and λ_S as in §3.1. For example, λ_R is a generating function associated with the sets of elements of R whose first value is fixed. If R is a free relation and if the probability distribution on attribute Y is given by $\{p_{i,d_Y}\}$, then $\lambda_R(y) = \prod_{1\le i\le d_Y}(1+p_{i,d_Y}y)$. When X is the key of R, then $\lambda_R(y) = 1+y$. Once again, we do not consider the probability generating function, which only differs from λ_R by a constant factor. Function λ_S describes in a similar way the legal sets of points in S.

Our aim is to study the conditional probability distribution $p(t/r,s)$, which gives the probability that the join has size t, knowing that the initial relations are, respectively, of sizes r and s. As in §4.1, we often use as an intermediary step the generating function of another, related distribution. We use whatever probability distribution has a generating function of a kind convenient for asymptotic study, namely $\phi(x,y,z)^d$. The rule of thumb is that, if an attribute Y or U is the key of the relation in which it appears (R or S), we should use a probability distribution conditioned by the size of this relation; moreover, the generating function should be exponential in the variable "marking" the relation. This is formalized in Theorem 2, below.

THEOREM 2. *Let R and S be two independent relations. The generating function $\Phi(x,y,z)$ of the sizes of relations R and S, and of the semijoin of R and S, is given by the table of Fig. 2, with the conventions that "X † Y" or "X † U" corresponds to a free relation, $Y \to X$ or $U \to X$ to a relation with key Y or U as applies, and $X \to Y$ or $X \to U$ means that X is key of R or S, and with the following definition of Φ:*

- *If each of the two relations* R *and* S *is either free or with a key X,*

$$\Phi(x,y,z) = \sum_{t,r,s} \mathrm{Proba}(t,r,s)\; x^t y^r z^s;$$

- *If attribute Y is key of relation R, and relation S is either free or with key X,*

$$\Phi(x,y,z) = \sum_{t,r,s} \mathrm{Proba}(t,s/r)\; x^t \cdot \frac{y^r}{r!} \cdot z^s;$$

- *If relation R is free or has key X, and attribute U is key of relation S,*

$$\Phi(x,y,z) = \sum_{t,r,s} \mathrm{Proba}(t,r/s)\; x^t \cdot y^r \cdot \frac{z^s}{s!};$$

R	S	$\Phi(x,y,z)$	Asymptotic result
$X \dagger Y$	$X \dagger U$	$(\lambda_R(y) + \lambda_R(xy)[\lambda_S(z) - 1])^{d_X}$	[10]
$X \dagger Y$	$X \to U$	$(\lambda_R(y) + z\lambda_R(xy))^{d_X}$	§5.4, Cor. 4
$X \dagger Y$	$U \to X$	$(\lambda_R(y) + \lambda_R(xy)[e^z - 1])^{d_X}$	[10]
$X \to Y$	$X \dagger U$	$(1 + y + (1 + xy)[\lambda_S(z) - 1])^{d_X}$	§5.3, Cor. 2
$X \to Y$	$X \to U$	$(1 + y + z + xyz)^{d_X}$	§5.4, Thm. 5
$X \to Y$	$U \to X$	$(1 + y + (1 + xy)[e^z - 1])^{d_X}$	§5.3, Cor. 3
$Y \to X$	$X \dagger U$	$(e^y + e^{xy}[\lambda_S(z) - 1])^{d_X}$	[10]
$Y \to X$	$X \to U$	$(e^y + ze^{xy})^{d_X}$	§5.4, Cor. 5
$Y \to X$	$U \to X$	$(e^y + e^{xy}[e^z - 1])^{d_X}$	[10]

FIG. 2. *Generating function for the size of the relations R, S, and their semijoin on attribute* X: $\{(x,y) | (x,y) \in R; \exists u : (x,u) \in S\}$.

- *If attributes Y and U are, respectively, keys of relations R and S,*

$$\Phi(x,y,z) = \sum_{t,r,s} \text{Proba}(t/r,s)\; x^t \cdot \frac{y^r}{r!} \cdot \frac{z^s}{s!}.$$

Proof of Theorem 2. The ordinary counting generating functions for the cases when none of the attributes Y and U is key can be found in [12]. The computation of the *joint probability* generating functions when neither attribute Y nor attribute U is key of its relation is straightforward, and we do not detail it. We give below the computation of $\Phi(x,y,z)$ when attribute Y is key of relation R and relation S is free. The cases where relation S has for key either X or U can be dealt with in a similar way.

We first assume that the probability distributions on attributes Y and U are uniform. It is simpler in this case, and it has no effect on function Φ, to use the *counting* generating function of the sets of elements on attribute X, that is, to take $\lambda_S(y) = (1+z)^{d_Z}$ instead of $(1 + z/d_Z)^{d_Z}$. Let us denote by $N(t,r,s)$ the number of couples of relations (R,S) with given sizes r and s, whose semijoin has size t, and by $N(r)$ the number of relations R of size r. The ordinary counting generating function $\Psi(x,y,z) = \sum_{r,s,t} N(t,r,s)\; x^t y^r z^s$ is [12]

$$\Psi(x,y,z) = \sum_k \binom{d_X}{k} (1 + d_X y + ky(x-1))^{d_Y} (\lambda_S(z) - 1)^k.$$

The conditional probability $p(t,s/r)$ is equal to $N(t,r,s)/N(r)$. We want to compute $\Phi(x,y,z) = \sum_{r,s,t} p(t,s/r)\; x^t \cdot y^r/r! \cdot z^s = \sum_{t,r,s} N(t,r,s)/N(r) \cdot x^t \cdot y^r/r! \cdot z^s$. Substituting the value $d_X^r \binom{d_Y}{r}$ for $N(r)$ in the expression of $p(t,s/r)$ gives

$$p(t,s/r) = \frac{[x^t y^r z^s]\Psi(x,y,z)}{d_X^r \binom{d_X}{r}} = \sum_k \binom{d_X}{k} d_X^{-r} [x^t](d_X - k + kx)^r [z^s](\lambda_S(z) - 1)^k.$$

Substituting this value into the definition of Φ, we get that

$$\Phi(x,y,z) = \sum_{k,r,s,t} \binom{d_X}{k} [x^t]\{(d_X - k + kx)^r\} x^t \cdot y^r/(d_X^r r!) \cdot [z^s]\{(\lambda_S(z) - 1)^k\} \cdot z^s$$

$$= e^y \sum_k \binom{d_X}{k} e^{k(x-1)y/d_X} \cdot (\lambda_S(z)-1)^k = (e^{y/d_X} + e^{xy/d_X} \cdot [\lambda_S(z)-1])^{d_X}.$$

The computation when at least one of the probability distributions on attributes Y or U is not uniform is in the same vein and presents no real difficulty. □

As in Theorem 1, we ignore in Theorem 2 constant multiplicative factors of the type $\lambda(1)$; we also ignore the coefficients $1/d_X$ of variables y or z. These factors might hide the global structure of Fig. 2, above, and do not serve any useful purpose: From §3.3, we know that the asymptotic study concerns the probability distribution generated by the function $f(x) = [y^r z^s]\Phi(x,y,z)/[y^r z^s]\Phi(1,y,z)$, and neither the elimination of a multiplicative factor in Φ nor the substitution of y for y/d_X have any effect on the function f. For example, the case detailed in the proof of Theorem 2, starting from the probability generating function for a uniform distribution, not from the counting generating function that we used, actually leads to the function $\Phi_0(x,y,z) = \left(e^{y/d_X} + e^{xy/d_X}(\lambda_S(z)-1)\right)^{d_X} \cdot \lambda_S(1)^{-d_X}$, and the function given in Fig. 2 is $\Phi(x,y,z) = (e^y + e^{xy}(\lambda_S(z)-1)^{d_X} = \lambda_S(1)^{d_X}\ \Phi_0(x, d_X y, z)$. Both functions lead to the same conditional probability distribution.

5.2. Limiting distributions. We can show that the semijoin size is asymptotically normal in several cases and that, as in the case of a projection, the probability distributions on attributes Y and U have almost no importance. There are three cases for each relation: It is free, or it has attribute X for key, or the other attribute is key (attribute Y for R, and attribute U for S). The choices for relations R and S are independent. As we see in §6.1, our method for proving results of asymptotic normality requires the evaluation of the coefficient $[y^r z^s]\Phi(x,y,z)$, and we classify the different cases according to the ease with which either one of the intermediate coefficients $[y^r]\Phi(x,y,z)$ or $[z^s]\Phi(x,y,z)$ can be computed. The possible cases for the two relations are listed below.

1. *None of relations R and S has X for key.* There is no direct way to evaluate $[y^r z^s]\Phi$, and we must use twice Cauchy's formula to compute it. We defer it to a future paper [10].
2. *R has X for key, but not S.* The extraction of $[y^r]\Phi$ gives a function in x and z: $\binom{d_X}{r}(1-x+x\lambda_S(z))^r \lambda_S(z)^{d_X - r}$. If, moreover, attribute U is key of relation S, then $\lambda_S(z)$ is the exponential function e^z. See Theorem 3 and Corollaries 2 and 3.
3. *S has X for key, but not R.* We compute the coefficient of z^s in the function Φ: $[z^s]\Phi = \binom{d_X}{s}\lambda_R(xy)^s \lambda_R(y)^{d_X - s}$. If relation R has attribute Y for key, then $\lambda_R(y) = e^y$. We then must study a bivariate function in x and y. This is done in Theorem 4 and Corollaries 4 and 5.
4. *R and S each have X for key.* The generating function has the simple form $\Phi(x,y,z) = (1+y+z+xyz)^{d_X}$, and either one of the coefficients $[y^r]\Phi$ and $[z^s]\Phi$ is easily computed. In this case, the semijoin of R and S has exactly the same number of elements as the intersection of relation R and the projection of relation S on attribute X, which has the same size as S. Conversely, the intersection of two relations can be seen as a semijoin of two relations with no other attribute than the one that is used in the join. Here again, we have a Gaussian limiting result (Theorem 5), which we state for the intersection of two relations built on the same attributes. Theorem 5 is also valid for the semijoin of two relations with a uniform probability distribution on the join attribute and without restriction on the distributions on the domains of

attributes Y and U.

The corollaries to the theorems below are valid for probability distributions in classes (Z) or (G) on the attributes Y and U, as indicated. In Theorems 3–5 and Corollaries 2–5, A and B are strictly positive constants. In the case of a probability distribution on either attribute Y or attribute U belonging to class (G), the domain sizes d_Y or d_U grow large (they are assumed to do so independently of each other and of d_X), and $\varphi_R(t)$ or $\varphi_S(t)$ denotes the limiting function $\prod_{i\geq 1}(1+p_i t)$; ρ is the unique real positive solution of the associated equation $t\varphi_R'(t)/\varphi_R(t) = A$ or $t\varphi_S'(t)/\varphi_S(t) = B$. The existence and uniqueness of ρ results from Lemma A in §3.4. The function g is defined by $g(t) = t\varphi_S'(t)/\varphi_S(t)$ in §5.3 and $g(t) = t\varphi_R'(t)/\varphi_R(t)$ in §5.4.

5.3. X key of R. We first give the general result pertaining to generating functions of the kind $\Phi(x,y,z) = (1+y+(1+xy)(\lambda_S(z)-1))^d$, then the corollaries dealing with the different cases for relation S. The proofs of Theorem 3 and of Corollaries 2 and 3 are given in §6.3.

THEOREM 3. *Let Property $\mathcal{P}$ be satisfied (see §3.4). Define $\Phi(x,y,z) = (1+y+(1+xy)(\lambda(z)-1))^d$. Let $d,r,s \to +\infty$ in such a way that $r < d$, $d = o(r^{3/2})$, and $s = Bd + o(d)$ for some positive constant B, and that $g(y) = y\lambda'(y)/\lambda(y)$ satisfies $\lim_{y\to+\infty} g(y) > B$. Then the probability distribution defined by the generating function $f(x) = [y^r z^s]\Phi(x,y,z)/[y^r z^s]\Phi(1,y,z)$ is asymptotically Gaussian. The asymptotic values of the mean and variance are defined below in terms of the solution ρ of the equation $g(y) = B$:*

$$\mu = r\left(1 - \frac{1}{\lambda(\rho)}\right), \qquad \sigma^2 = r\left(\frac{\lambda(\rho)-1}{\lambda^2(\rho)} - \frac{r}{d}\frac{\rho\lambda'^2(\rho)}{\lambda^4(\rho)g'(\rho)}\right).$$

COROLLARY 2. *Let $R[X,Y]$ be a relation with a key X, and $S[X,U]$ a free relation. We assume that the probability distribution on D_X is uniform; the probability distribution on D_Y is arbitrary. The sizes r and s of the relations R and S are assumed to satisfy $r < d_X$, $d_X = o(r^{3/2})$, and $s = Bd_X(1+o(1))$. Then the probability distribution of the size of the semijoin of R and S on attribute X, conditioned by the sizes of R and S, is asymptotically normal.*

If the distribution on D_U satisfies hypothesis (Z), *the mean and variance have for asymptotic values $\mu = (1-e^{-B})r$ and $\sigma^2 = r((e^B-1)/e^{2B} - (rB)/d_X e^{2B})$. If the probability distribution on D_U satisfies hypothesis* (G), *the asymptotic values of the mean and variance are*

$$\mu = \mu_0 r = (1 - 1/\varphi_S(\rho))\, r, \qquad \sigma^2 = r\left(\frac{\varphi_S(\rho)-1}{\varphi_S^2(\rho)} - \frac{r}{d_X}\frac{\rho\varphi_S'^2(\rho)}{\varphi_S^4(\rho)g'(\rho)}\right).$$

Moreover, the constant μ_0 satisfies $1 - e^{-B} \leq \mu_0 \leq 1$.

COROLLARY 3. *Let $R[X,Y]$ be a relation with a key X, and $S[X,U]$ a relation with a key U. We assume that the probability distribution on D_X is uniform. The probability distributions on D_Y and D_U are arbitrary. The sizes r and s of the relations R and S are assumed to satisfy $r < d_X$, $d_X = o(r^{3/2})$, and $s = Bd_X(1+o(1))$. Then the probability distribution of the size of the semijoin of R and S on attribute X, conditioned by the sizes of R and S, is asymptotically normal. The mean and variance have for asymptotic values $\mu = (1-e^{-B})r$ and $\sigma^2 = r((e^B-1)/e^{2B} - rB/d_X e^{2B})$.*

The comparison of Corollary 2 in the case of a distribution of class (Z) on attribute U and of Corollary 3 shows that the two asymptotic distributions of the projection size are the same. As in the case of a projection, the exact distribution of the values of attribute U for a free relation, as long as it is not too far from uniform, or the existence of a key on U, are of no importance with respect to the asymptotic size of the semijoin when d_X and d_U go to infinity.

5.4. X key of S. This part deals with the cases where the generating function is of the kind $\Phi(x,y,z) = (\lambda_R(y) + z\lambda_R(xy))^d$. Here again, we first give the general result (Theorem 4), then the applications to the semijoin size (Corollaries 4 and 5), and we defer the proofs until §6.4.

THEOREM 4. *Let Property $\mathcal{P}$ be satisfied (see §3.4). Define $\Phi(x,y,z) = (\lambda(y) + z\lambda(xy))^d$. Let $d,r,s \to +\infty$ in such a way that $s < d$, $d = o(s^{3/2})$ and $r = Ad + o(d)$, and that $g(y) = y\lambda'(y)/\lambda(y)$ satisfies $\lim_{y\to+\infty} g(y) > A$. Then the probability distribution defined by the generating function $f(x) = [y^r z^s]\Phi(x,y,z)/[y^r z^s]\Phi(1,y,z)$ is asymptotically Gaussian. The asymptotic value of the mean is $\mu = rs/d$. The asymptotic value of the variance is defined in terms of the solution ρ of the equation $g(y) = A$: $\sigma^2 = s(1-s/d)\rho g'(\rho)$.*

COROLLARY 4. *Let $R[X,Y]$ be a free relation, and $S[X,U]$ a relation with a key X. We assume that the probability distribution on D_X is uniform and that the probability distribution on D_Y is in class* (Z) *or* (G); *the probability distribution on D_U is arbitrary. The sizes r and s of the initial relations are assumed to satisfy $r = Ad_X + o(d_X)$, $s < d_X$, and $d_X = o(s^{3/2})$. Then the probability distribution of the size of the semijoin of R and S on attribute X, conditioned by the sizes r and s of the initial relations, is asymptotically normal. The asymptotic mean is $\mu = rs/d_X$. If the distribution on attribute Y belongs to class* (Z), *the asymptotic variance is equal to $\sigma^2 = (1-s/d_X)rs/d_X$. If the distribution on attribute Y belongs to class* (G), *the asymptotic variance becomes $\sigma^2 = s(1-s/d_X)\sigma_0^2$ for a positive constant σ_0^2.*

COROLLARY 5. *Let $R[X,Y]$ be a relation with key Y, and $S[X,U]$ a relation with a key X. We assume that the probability distribution on D_X is uniform. The probability distributions on D_Y and D_U are arbitrary. The sizes r and s of the initial relations are assumed to satisfy $r = Ad_X + o(d_X)$, $s < d_X$, and $d_X = o(s^{3/2})$. Then the probability distribution of the size of the semijoin of R and S on attribute X, conditioned by the sizes $r = Ad_X(1+o(1))$ of R and s of S, is asymptotically normal. The asymptotic mean and variance are given by $\mu = rs/d_X$ and $\sigma^2 = (1-s/d_X)rs/d_X$.*

Once again, a comparison of Corollaries 4 and 5 shows that the existence of a key, or a probability distribution of class (Z), on attribute Y have no influence on the asymptotic behaviour of the semijoin size.

In the case where both relations R and S have attribute X for key, we have the following result, which is proved in §6.5.

THEOREM 5. *Let R and S be two free relations, of sizes r and s. We assume that the probability distribution on the set of size d of possible tuples is uniform. We take $r = Ad(1+o(1))$ and $s = Bd(1+o(1))$, where A and B are constants in $]0,1[$. Then the probability distribution of the size of the intersection of R and S, conditioned by the sizes r and s of the initial relations, is asymptotically normal. The mean and variance are given by $\mu = rs/d$ and $\sigma^2 = rs/d(1-r/d)(1-s/d)$; their asymptotic values are, respectively, ABd and $AB(1-A)(1-B)d$.*

6. Proofs of theorems.

6.1. Sketch of the proofs. Theorems 1, 3–5 have a common flavor: We are interested in a function $\Phi(x,y)$ or $\Phi(x,y,z)$, which defines a conditional probability distribution, and we want to know if this distribution has a limit when some parameters r and s go to infinity. Moreover, the function Φ is of the kind ϕ^d, where the exponent d also grows to infinity. The generating function for the conditional distribution is $f(x) = [y^r]\Phi(x,y)/[y^r]\Phi(1,y)$ or $f(x) = [y^r z^s]\Phi(x,y,z)/[y^r z^s]\Phi(1,y,z)$.

Let us sketch the method that we use to study the limit of the distribution defined by $f(x)$ when function Φ is bivariate. When the initial function is $\Phi(x,y,z)$, we restrict ourselves in this paper to cases where at least one of the coefficients $[y^r]\Phi(x,y,z)$ or $[z^s]\Phi(x,y,z)$ can be computed by the binomial formula, and the evaluation of a limiting distribution defined by function $f(x)$ proceeds in a similar way.

We first evaluate $\psi(x) = [y^r]\Phi(x,y)$, for x real. By Cauchy's formula for an analytic function, $\psi(x)$ can be written as an integral on a closed contour around the origin as follows:

$$\psi(x) = \frac{1}{2i\pi}\oint \Phi(x,y)\frac{dy}{y^{r+1}}. \tag{2}$$

In all the cases that we consider in this paper, the function Φ has no singularity, and we use the saddlepoint method [3], [6], [14] to approximate this integral. We take for integration path in (2) a circle $y = \rho(x)e^{i\theta}$, centered at the origin, whose radius $\rho(x)$ is chosen in such a way that only a small part of the circle contributes to the integral and that the integral on the rest of the circle just gives an error term. The point $\rho(x)$ is a *saddlepoint*; it is defined from the function $h(x,y) = \log\Phi(x,y) - (r+1)\log y$ as the solution of the equation $(\partial h/\partial y)(x,y) = 0$. The saddlepoint approximation then gives the following approximate value of $\psi(x)$:

$$\psi(x) = \frac{e^{h(x,\rho(x))}}{\sqrt{2\pi\partial^2 h/\partial y^2(x,\rho(x))}}(1+o(1)).$$

We next show the pointwise convergence of the Laplace transform $e^{t\mu/\sigma}\psi(e^{-t/\sigma})/\psi(1)$ of the normalized random variable associated with the probability generating function $f(x) = \psi(x)/\psi(1)$, toward $e^{t^2/2}$, for suitably chosen values of μ and σ and for any fixed t in the interval $[0,+\infty[$. Classical results on the convergence of probability distributions (see, for example, [7, Chap. XIII, Thm. 2]) allow us to conclude to the convergence of the probability distribution defined by $f(x)$ toward a normal distribution of mean μ and variance σ^2.

We give in some detail the proof of Theorem 1 and, more quickly, the proofs of Theorems 3–5 and of the corollaries.

6.2. Proof of Theorem 1 for projections. We recall that the bivariate function we consider is $\Phi(x,y) = (1 - x + x\lambda(y))^d$, with Property $\mathcal{P}$ of §3.4 satisfied: It is entire and not affine, with positive coefficients, such that $\lambda(0) = 1$, and such that there exists no entire function Λ and integer $m \geq 2$ with $\lambda(y) = \Lambda(y^m)$. Let us define

$$h(x,y) = d\log(1 - x + x\lambda(y)) - (r+1)\log y. \tag{3}$$

Equation (2) becomes

$$\psi(x) = [y^r]\Phi(x,y) = \frac{1}{2i\pi}\oint e^{h(x,y)}dy.$$

6.2.1. Choice of the integration contour. The equation in y defining the saddlepoint is $\partial h/\partial y(x,y) = 0$, which can be rewritten as

$$\frac{xy\lambda'(y)}{1-x+x\lambda(y)} = \frac{r+1}{d}. \tag{4}$$

The solution of this equation in y is a function of x, r, and d. We must take x equal to 1, or near 1: $x = e^{-t/\sigma}$, for fixed t and large σ. We solve the equation for $x = 1$, then give an approximate solution for x close to and smaller than 1. For $x = 1$, (4) becomes

$$\frac{y\lambda'(y)}{\lambda(y)} = \frac{r+1}{d}. \tag{5}$$

We first show that (5) has a unique real positive solution ρ_0.

Lemma A of §3.4 and the fact that $g(0) = 0$ together show that ρ_0 exists and is unique if and only if $\lim_{y\to+\infty} y\lambda'(y)/\lambda(y) > (r+1)/d$, which can be simplified into a condition independent of d and r, below (we recall that $d, r \to +\infty$ and that $r = Ad + o(d)$):

$$\lim_{y\to+\infty} \frac{y\lambda'(y)}{\lambda(y)} > A. \tag{6}$$

We henceforth assume that the condition (6) holds: For r and d large enough, (5) has a unique real positive solution ρ_0. We look for a solution of (4) under the form $\rho(x) = (1+u)\rho_0$ with $u = o(1)$, for $x = 1+\varepsilon$ and $\varepsilon = o(1)$, with $\varepsilon < 0$.

Functions λ and λ' can be expanded near the origin as follows:

$$\lambda(y) = \lambda(\rho_0) + u\lambda'(\rho_0)\rho_0 + O(u^2),$$
$$\lambda'(y) = \lambda'(\rho_0) + u\lambda''(\rho_0)\rho_0 + O(u^2).$$

We rewrite (4) as $xy\lambda'(y)/(1-x+x\lambda(y)) = g(\rho_0)$ then plug the expansions of λ and λ' into it. In terms of the function

$$D(y) = \lambda(y)(\lambda'(y) + y\lambda''(y)) - y\lambda'^2(y),$$

this gives the following equation on u and $\varepsilon = x - 1$:

$$\lambda'(\rho_0)\varepsilon + D(\rho_0)u + O(u^2) + O(\varepsilon u) = 0.$$

The coefficient of u in this equation is $D(\rho_0) > 0$, and we can compute the following approximate value of u:

$$\rho(x) = (1+u)\rho_0; \qquad u = -\frac{\lambda'(\rho_0)}{D(\rho_0)}\varepsilon + O(\varepsilon^2). \tag{7}$$

6.2.2. Approximation of $\psi(x)$. In this section, x is fixed and real near 1. As we will need, in §6.2.3, to take $x = e^{-t/\sigma}$ for $t > 0$ and $\sigma > 0$, we can restrict ourselves to $x \le 1$. We also abbreviate $\partial^2 h/\partial y^2$ into h'' in this section; this should cause no ambiguity.

We choose for integration contour in the integral (2) a circle with radius $\rho(x)$:

$$\psi(x) = \frac{1}{2i\pi}\oint e^{h(x,y)}dy = \frac{1}{2i\pi}\int_{\theta\in[-\pi,+\pi]} e^{h(x,\rho(x)e^{i\theta})}d(\rho(x)e^{i\theta}).$$

We divide this integral in two parts: I_1 is the part of the integral dealing with a restricted piece of the path near point $\rho(x)$ and will give the main term (11); the complement I_2 will give an exponentially smaller term (12). We first choose $\alpha \in]0, \pi[$ (we see in the following the conditions that α must satisfy), then we formally define I_1 and I_2 by

$$I_1 = \frac{1}{2i\pi} \int_{\theta \in]-\alpha, +\alpha[} e^{h(x, \rho(x)e^{i\theta})} d(\rho(x)e^{i\theta}),$$
$$I_2 = \frac{1}{2i\pi} \int_{\alpha \le |\theta| \le \pi} e^{h(x, \rho(x)e^{i\theta})} d(\rho(x)e^{i\theta}).$$

We have that $\psi(x) = I_1 + I_2$, and our goal is to prove that we can find a value of α such that

$$\psi(x) = \frac{e^{h(x,\rho(x))}}{\sqrt{2\pi h''(x, \rho(x))}}(1 + o(1)). \tag{8}$$

Evaluation of I_1. We can always assume that x varies near 1 in such a way that $\rho(x)$ belongs to a compact set near ρ_0. Actually, ρ_0 itself is a function of d and r but can be restricted to a compact neighbourhood of $g^{-1}(A)$ (which is a constant). This means that we can restrict $\rho(x)$ to a compact interval around $g^{-1}(A)$ independently of r, d, and x. This will henceforth be used implicitly to prove that the error terms that we consider are uniform with respect to r, d, and x. We abbreviate $\rho(x)$ into ρ for the evaluation of I_1; again, this should cause no confusion. The evaluation of I_1 is similar to the corresponding ones in the proofs of Theorems 3 and 4. This leads us to state the following lemma, which we use again in the next proofs.

LEMMA B. *Let $h_d(x, y)$ be a function that depends on a parameter d, defined and twice differentiable for (x, y) in a compact neighbourhood of the point $(1, \rho_0)$, where ρ_0 satisfies the equation $\partial h_d/\partial y(1, \rho_0) = 0$. We assume furthermore that, as d varies, ρ_0 stays in a compact subset of $]0, +\infty[$. Define ρ as the solution (dependent on x and d) of $\partial h_d/\partial y(x, \rho) = 0$. Assume that*

- *$\partial^2 h_d/\partial y^2(1, \rho_0)$ is of order exactly d;*
- *for x near 1, $\partial^2 h_d/\partial y^2(x, \rho) = \partial^2 h_d/\partial y^2(1, \rho_0)(1 + o(1))$ with an error term uniform in x and independent of d;*
- *the function $\theta \mapsto h_d(x, \rho e^{i\theta})$ has a Taylor expansion near zero satisfying*

$$h_d(x, \rho e^{i\theta}) = h_d(x, \rho) + \frac{\rho^2}{2}(e^{i\theta} - 1)^2 \partial^2 h_d/\partial y^2(x, \rho) + O(d\theta^3),$$

with the $O()$ term such that the implied constant can be chosen independently of x and d.

Then there exist constants $\alpha_0 > 0$, γ_0, and γ_1 such that, for any $\alpha \in]0, \alpha_0[$, and with implied constants in the $O()$ terms independent of x and d,

$$\frac{1}{2i\pi} \int_{\theta \in]-\alpha, +\alpha[} e^{h_d(x, \rho e^{i\theta})} d(\rho e^{i\theta}) = \frac{e^{h_d(x,\rho)}}{\sqrt{2\pi \partial^2 h_d/\partial y^2(x, \rho)}} \cdot (1 + O(\alpha^2 \sqrt{d} e^{-\gamma_0 d\alpha^2}) + O(e^{-\gamma_1 d\alpha^2}) + O(d\alpha^3)).$$

The proof of Lemma B is given in Appendix A. We now check that its assumptions are satisfied for the function h defined by (3). The function $h(x, \rho e^{i\theta})$ can be expanded

in the variable θ around the origin as follows:

$$h(x,\rho e^{i\theta}) = h(x,\rho) + \rho(e^{i\theta}-1)\partial h/\partial y(x,\rho) + \frac{\rho^2}{2}(e^{i\theta}-1)^2 h''(x,\rho) + O(||h'''||(e^{i\theta}-1)^3). \tag{9}$$

Define $\phi(x,y) = 1-x+x\lambda(y)$. Definition (3) gives $h''(x,y) = d\cdot\partial^2(\log\phi)/\partial y^2(x,y) + (r+1)/y^2$. For x and y near 1 and ρ_0, respectively, we have that $\partial^2(\log\phi)/\partial y^2(x,y) = \partial^2(\log\phi)/\partial y^2(1,\rho_0) + O(x-1) + O(y-\rho_0)$ and $1/y^2 = 1/\rho_0^2 + O(y-\rho_0)$. This and the fact that $r = \Theta(d)$ give $h''(x,y) = h''(1,\rho_0) + d\,O(x-1) + d\,O(y-\rho_0)$. We next check that $h''(1,\rho_0)$ has order exactly d: $h''(1,\rho_0) = d\,g'(\rho_0)/\rho_0$. Hence we can factor it out of the expression of $h''(x,y)$, and we get that

$$h''(x,y) = h''(1,\rho_0)(1 + O(x-1) + O(y-\rho_0)). \tag{10}$$

The $O()$ terms in this expansion are independent of r and d. We deduce from it that, for x close to 1 and $y = \rho(x) = \rho$, $h''(x,\rho) = h''(1,\rho_0)(1+o(1))$. The error term in this relation is uniform for $x \to 1$ and $r, d \to +\infty$. A similar argument shows that the term $||h'''||$ is actually $O(d)$. We also have, by definition of ρ, that $\partial h/\partial y(x,\rho) = 0$. Equation (9) then becomes

$$h(x,\rho e^{i\theta}) = h(x,\rho) + \frac{\rho^2}{2}(e^{i\theta}-1)^2 h''(x,\rho) + O(d\theta^3).$$

Lemma B finally gives the following approximation of I_1:

$$I_1 = \frac{e^{h(x,\rho)}}{\sqrt{2\pi h''(x,\rho)}}(1 + O(\alpha^2\sqrt{d}e^{-\gamma_0 d\alpha^2})O(e^{-\gamma_1 d\alpha^2}) + O(d\alpha^3)). \tag{11}$$

Upper bound on I_2. We recall that

$$I_2 = \frac{1}{2i\pi}\int_{\alpha\le|\theta|\le\pi} e^{h(x,\rho(x)e^{i\theta})} d(\rho(x)e^{i\theta}).$$

We extract from the integral the main term of I_1: $e^{h(x,\rho(x))}$; this gives

$$I_2 = \frac{\rho(x)e^{h(x,\rho(x))}}{2\pi}\int_{\alpha\le|\theta|\le\pi} e^{-ir\theta} k_x(\theta)^d d\theta,$$

with $k_x(\theta) = (1-x+x\lambda(\rho(x)e^{i\theta}))/(1-x+x\lambda(\rho(x)))$. We now want an upper bound on $|k_x(\theta)|$, for $|\theta| \in [\alpha,\pi]$. The term $1-x$ is $o(1)$, of smaller order than the term $\lambda(\rho(x))$, and we get that

$$|k_x(\theta)| \le \frac{1-x+x|\lambda(\rho(x)e^{i\theta})|}{1-x+x\lambda(\rho(x))}.$$

Lemma C, below (proved in Appendix B), gives a bound on $|\lambda(\rho(x)e^{i\theta})|$, which, in turn, gives an inequality on $|k_x(\theta)|$: $|k_x(\theta)| \le 1-C_1\alpha^2$, for a strictly positive constant C_1, independent of r, d, x, and α. We finally get that

$$|I_2| \le \rho(x)e^{h(x,\rho(x))}(1-C_1\alpha^2)^d = e^{h(x,\rho(x))}O(e^{-C_1 d\alpha^2}),$$

which gives

$$|I_2| = \frac{e^{h(x,\rho(x))}}{\sqrt{h''(x,\rho(x))}} O(\sqrt{d}\, e^{-C_1 d\alpha^2}). \tag{12}$$

LEMMA C. *Let λ be a function satisfying Property $\mathcal{P}$ of* §3.4, *and $\alpha \in]0,\pi[$. Let y vary in a compact subset of $]0,+\infty[$. Then there exists a constant $C > 0$ such that, for all θ satisfying $\alpha < |\theta| < \pi$, and for all y in the compact subset, the following inequality holds*:

$$|\lambda(ye^{i\theta})| \leq \lambda(y)(1 - C\alpha^2).$$

Choice of α. We obtain the following approximation of $\psi(x)$:

$$\psi(x) = \frac{e^{h(x,\rho(x))}}{\sqrt{2\pi h''(x,\rho(x))}}(1+O(e^{-\gamma_1 d\alpha^2})+O(\sqrt{d}e^{-C_1 d\alpha^2})+O(\alpha^2\sqrt{d}e^{-\gamma_0 d\alpha^2})+O(d\alpha^3)).$$

Approximation (8) holds if we can choose α such that the error terms are negligible. For $d \to +\infty$, this is a consequence of

$$\frac{d\alpha^2}{\log d} \to +\infty, \qquad d\alpha^3 \to 0.$$

For $\alpha = (\log d)/\sqrt{d}$, it is easy to check that $d\alpha^2/\log d = \log d$ and $d\alpha^3 = \log^3 d/\sqrt{d} = o(1)$, and we have that

$$\psi(x) = \frac{e^{h(x,\rho(x))}}{\sqrt{2\pi h''(x,\rho(x))}}(1 + o(1)).$$

6.2.3. Convergence of the normalized Laplace transform. We show here that we can choose μ and σ in such a way that the function $e^{t\mu/\sigma} f(e^{-t/\sigma}) = e^{t\mu/\sigma}\psi(e^{-t/\sigma})/\psi(1)$ converges toward $e^{t^2/2}$ when $d \to +\infty$ and for every $t > 0$. Taking the logarithm, we must prove the convergence of $\Xi(t) = t\mu/\sigma + \log(\psi(e^{-t/\sigma})/\psi(1))$ toward $t^2/2$.

Equation (8) shows that

$$\log\left(\frac{\psi(x)}{\psi(1)}\right) = h(x,\rho(x)) - h(1,\rho_0) - \frac{1}{2}\log\left(\frac{h''(x,\rho(x))}{h''(1,\rho_0)}\right) + o(1),$$

which gives

$$\Xi(t) = t\mu/\sigma + \delta h(e^{-t/\sigma}, \rho(e^{-t/\sigma})) + o(1), \tag{13}$$

with

$$\delta h(x,y) = h(x,y) - h(1,\rho_0) - \frac{1}{2}\log\frac{h''(x,y)}{h''(1,\rho_0)}. \tag{14}$$

We formerly proved in (10) that h'' can be expanded near the point $(1,\rho_0)$, $h''(x,y) = h''(1,\rho_0)(1 + O(x-1) + O(y-\rho_0))$. This shows that, for $x = 1+\varepsilon$,

$$\log\left(\frac{h''(x,\rho(x))}{h''(1,\rho_0)}\right) = O(\varepsilon).$$

Evaluation of $h(x,\rho(x)) - h(1,\rho_0)$. The function h can be expanded near the point $(1,\rho_0)$ as follows:

$$\begin{aligned} h(x,y) = h(1,\rho_0) &+ (x-1)\frac{\partial h}{\partial x}(1,\rho_0) + (y-\rho_0)\frac{\partial h}{\partial y}(1,\rho_0) \\ &+ \frac{1}{2}(x-1)^2\frac{\partial^2 h}{\partial x^2}(1,\rho_0) + (x-1)(y-\rho_0)\frac{\partial^2 h}{\partial x \partial y}(1,\rho_0) + \frac{1}{2}(y-\rho_0)^2\frac{\partial^2 h}{\partial y^2}(1,\rho_0) \\ &+ O(d(x-1)^3) + O(d(y-\rho_0)^3). \end{aligned}$$

For $x-1=\varepsilon$ and $y-\rho_0 = u\rho_0$, and substituting the values of the derivatives of h at point $(1,\rho_0)$, we get that

$$\begin{aligned} h(1+\varepsilon,(1+u)\rho_0) = h(1,\rho_0) &+ d\frac{\lambda(\rho_0)-1}{\lambda(\rho_0)}\varepsilon - \frac{d}{2}\cdot\frac{(\lambda(\rho_0)-1)^2}{\lambda^2(\rho_0)}\varepsilon^2 \\ &+ d\frac{\lambda'(\rho_0)}{\lambda(\rho_0)}\varepsilon u\rho_0 + \frac{d}{2}\,g'(\rho_0)u^2\rho_0 + O(d\varepsilon^3). \end{aligned}$$

We now use the value of u computed in (7): $u = -\varepsilon\lambda'(\rho_0)/D(\rho_0) + O(\varepsilon^2)$, which gives

$$h(1+\varepsilon,\rho_0(1+u)) = h(1,\rho_0) + d\alpha_1\varepsilon - \frac{d}{2}\alpha_2\varepsilon^2 + O(d\varepsilon^3).$$

The coefficients α_1 and α_2 in this formula are defined by

$$\alpha_1 = 1 - \frac{1}{\lambda(\rho_0)}, \qquad \alpha_2 = \alpha_1^2 + \frac{\lambda'^2(\rho_0)\rho_0}{\lambda^2(\rho_0)D(\rho_0)}.$$

The values of h and h'' in (14) are now replaced and we get that

$$\delta h(1+\varepsilon,(1+\varepsilon)\rho_0) = d\alpha_1\varepsilon - \frac{d}{2}\alpha_2\varepsilon^2 + O(d\varepsilon^3) + O(\varepsilon). \tag{15}$$

We next define $x = e^{-t/\sigma} = 1 - t/\sigma + t^2/2\sigma^2 + O(t^3/\sigma^3)$, i.e., $\varepsilon = -t/\sigma + t^2/2\sigma^2 + O(t^3/\sigma^3)$. Equations (13) and (15) show that

$$\Xi(t) = (\mu - d\alpha_1)\frac{t}{\sigma} + d(\alpha_1-\alpha_2)\frac{t^2}{2\sigma^2} + O\left(\frac{d}{\sigma^3}\right) + O\left(\frac{1}{\sigma}\right) + o(1).$$

Determination of the mean μ *and the variance* σ. Let us define $\mu = d\alpha_1$ and $\sigma^2 = d(\alpha_1-\alpha_2)$. They can be written as

$$\mu = d\frac{\lambda(\rho_0)-1}{\lambda(\rho_0)}, \qquad \sigma^2 = d\left(\frac{\lambda(\rho_0)-1}{\lambda^2(\rho_0)} - \frac{\lambda'^2(\rho_0)\rho_0}{\lambda^2(\rho_0)D(\rho_0)}\right).$$

Here ρ_0 is equal to $g^{-1}(r+1)/d$ and has for asymptotic value the solution of $g(y) = A$. Hence the mean and variance are asymptotically equal, respectively, to $d\mu_0$ and $d\sigma_0^2$, for μ_0 and σ_0 defined in function of $\rho = g^{-1}(A)$ as follows:

$$\mu_0 = \frac{\lambda(\rho)-1}{\lambda(\rho)}, \qquad \sigma_0^2 = \frac{\lambda(\rho)-1}{\lambda^2(\rho)} - \frac{\rho\lambda'^2(\rho)}{\lambda^2(\rho)D(\rho)}.$$

We now check that σ_0^2 is strictly positive. In terms of the functions $g_1(y) = y\lambda'(y)/(\lambda(y) - 1)$ and $g(y) = y\lambda'(y)/\lambda(y)$, we have that $\sigma_0^2 = (\lambda(\rho) - 1)^2 g_1'(\rho)/(\lambda^3(\rho) g'(\rho))$. By an argument similar to that used in §3.4 to prove that g is an increasing function (see Lemma A and its proof), we can show that the value $g_1'(\rho)$ is positive, which in turn shows that $\sigma_0^2 > 0$. The error terms $O(d/\sigma^3)$ and $O(1/\sigma)$ both become $O(1/\sqrt{d}) = o(1)$, and we finally have that $\Xi(t) = t^2/2 + o(1)$, which ends the proof of Theorem 1. □

6.2.4. Proof of Corollary 1. Checking that the function $\lambda_{d_Y}(y) = \prod_{1 \le i \le d_Y}(1 + p_{i,d_Y} y)$ satisfies Property $\mathcal{P}$ presents no difficulty. If the size d_Y of D_Y is fixed, Corollary 1 is a direct consequence of Theorem 1. If d_Y grows to infinity independently of d_X and r, we must adjust the proof as indicated below. We recall that we assume the independence of d_X and d_Y.

We work with a *sequence* of functions $\lambda_{d_Y}(y) = \prod_{1 \le i \le d_Y}(1 + p_{i,d_Y} y)$. When the probability distribution on attribute Y is in class (Z) or (G), this sequence converges normally toward a function $\varphi(y)$ for any y in a compact subset of the complex plane and for $d_Y \to +\infty$. The saddlepoint ρ_0 for $x = 1$ has a finite, nonnull limit ρ when $r, d_X, d_Y \to +\infty$. This limit ρ also satisfies the limiting equation $t\varphi'(t)/\varphi(t) = A$. We solve the equation, giving the general saddlepoint $\rho(x)$ exactly as in §6.2.1. The solution now also depends on d_Y, and it is important to note that $\rho(x)$ can be restricted to a compact neighbourhood of ρ for $x \to 1$ and $r, d_X, d_Y \to +\infty$. The rest of the proof is then the same as the corresponding part of the proof of Theorem 1, with uniform error terms in our approximations.

When the distribution on attribute Y belongs to class (G), the inequality $\mu_0 \ge 1 - e^{-A}$ is equivalent to $e^A \le \varphi(\rho)$ or (by $g(\rho) = A$) to $g(\rho) < \log \phi(\rho)$. As $g(y) = y(\log \phi)'(y)$ and $\log \varphi(y) = \sum_{i \ge 1} \log(1 + p_i y)$, we have that $g(y) = \sum_{i \ge 1} p_i y/(1 + p_i y)$. We can then rewrite the former inequality as $\sum_{i \ge 1}(\log(1 + p_i\rho) - p_i\rho/(1 + p_i\rho)) \ge 0$. The function $t \mapsto \log(1 + t) - t/(1 + t)$ is positive for all $t \in]0, +\infty[$, and each of the terms of the global inequality is positive, which proves the lower bound on μ_0. □

6.3. Proof of Theorem 3 for semijoins: X key of R. Theorem 3 is an extension of Theorem 1, when we multiply the function $\Phi(x, y) = (1 - x + x\lambda(y))^d$ by a term $\varphi(z) = \lambda(z)^{d-r}$. The proof of Theorem 3 is similar to that of Theorem 1, and we mainly indicate the points where it differs.

The coefficient $[y^r]\Phi(x, y, z)$ is $\binom{d}{r}(1 - x + x\lambda(z))^r \lambda(z)^{d-r}$. Let us define

$$\psi(x) = [y^r z^s]\Phi(x, y, z)/\binom{d}{r} = \frac{1}{2i\pi}\oint e^{h(x,z)} dz,$$

with

$$h(x, z) = r \log(1 - x + x\lambda(z)) + (d - r)\log \lambda(z) - (s + 1)\log z.$$

6.3.1. Evaluation of the saddlepoint $z(x)$. We have that $h(1, z) = d \log \lambda(z) - (s + 1)\log z$. Define $g(z) = z\lambda'(z)/\lambda(z)$; the equation $\partial h/\partial z(1, z) = 0$ becomes $g(z) = (s + 1)/d$. We assume that $\lim_{s,d \to +\infty}(s + 1)/d$ exists and is equal to B. As function λ satisfies Property $\mathcal{P}$ of §3.4, Lemma A of that section shows that the equation $g(z) = (s + 1)/d$ has a unique real positive solution ρ_0 if and only if $\lim_{z \to +\infty} g(z) > B$. We then solve the equation $\partial h/\partial z(1, z) = 0$, for $x = 1 + \varepsilon$ and $z = (1 + u)\rho_0$. We first rewrite it into

$$\left(1 + \frac{r(x - 1)}{d(1 - x + x\lambda(z))}\right) g(z) = \frac{s + 1}{d}.$$

Using expansions of the functions λ and g near ρ_0, we get the approximate equation

$$\frac{rg(\rho_0)}{d\lambda(\rho_0)}\varepsilon + g'(\rho_0)u\rho_0 + O\left(\frac{r}{d}\varepsilon^2\right) + O\left(\frac{r}{d}\varepsilon u\right) + O(u^2) = 0.$$

We solve it and get that

$$u = -\alpha\varepsilon(1+O(\varepsilon)), \qquad \alpha = \frac{r\lambda'(\rho_0)}{d\lambda^2(\rho_0)g'(\rho_0)}. \tag{16}$$

We need the values of the derivatives of function h near point $(1,\rho_0)$ in §§6.3.2 and 6.3.3, so we give them below: $\partial h/\partial z(1,\rho_0) = 0$, the derivatives of order 3 of h are $O(d)$, and

$$\frac{\partial h}{\partial x}(1,\rho_0) = r\left(1-\frac{1}{\lambda(\rho_0)}\right), \qquad \frac{\partial^2 h}{\partial x^2}(1,\rho_0) = -r\left(1-\frac{1}{\lambda(\rho_0)}\right)^2,$$
$$\frac{\partial^2 h}{\partial x \partial z}(1,\rho_0) = r\frac{\lambda'(\rho_0)}{\lambda^2(\rho_0)}, \qquad \frac{\partial^2 h}{\partial z^2}(1,\rho_0) = d\frac{g'(\rho_0)}{\rho_0}.$$

6.3.2. Evaluation of $\psi(x)$. In the formula $\psi(x) = (1/2i\pi)\oint e^{h(x,z)}dz$, we take for integration path a circle centered at the origin and with radius $z(x) = (1+u)\rho_0$, with u defined by (16). We choose $\alpha \in]0,\pi[$ and divide the integral in two parts: $I_1 = (1/2i\pi)\int_{|\theta|\le\alpha} e^{h(x,z)}dz$ and $I_2 = (1/2i\pi)\int_{\alpha\le|\theta|\le\pi} e^{h(x,z)}dz$. Lemma B of §6.2.2 gives an approximation of I_1 as follows:

$$I_1 = \frac{e^{h(x,z(x))}}{\sqrt{2\pi h''_{z^2}(x,z(x))}}(1+O(\alpha^2\sqrt{d}e^{-\gamma_0 d\alpha^2}) + O(e^{-\gamma_1 d\alpha^2}) + O(d\alpha^3)).$$

Lemma C of §6.2.2 then gives an upper bound on $|\lambda(z(x)e^{i\theta})|$ for $\alpha \le |\theta| \le \pi$; it is easy to show from it that

$$I_2 = \frac{e^{h(x,z(x))}}{\sqrt{h''_{z^2}(x,z(x))}}O(\sqrt{d}e^{-\gamma_2 d\alpha^2}).$$

By choosing $\alpha = (\log d)/\sqrt{d}$, we obtain that

$$\psi(x) = \frac{e^{h(x,z(x))}}{\sqrt{h''_{z^2}(x,z(x))}}(1+o(1)).$$

6.3.3. Laplace transform and determination of moments. Always following the same path as in the proof of Theorem 1, we now compute

$$\log\frac{\psi(x)}{\psi(1)} = h(x,z(x)) - h(1,\rho_0) - \frac{1}{2}\log\frac{\partial^2 h/\partial z^2(x,z(x))}{\partial^2 h/\partial z^2(1,\rho_0)} + o(1).$$

It is easy to check that $\log(\partial^2 h/\partial z^2(x,z(x))/\partial^2 h/\partial z^2(1,\rho_0))$ is $O(x-1)$. We then expand the function $h(x,z(x))$ near the point $(1,\rho_0)$ as follows:

$$\begin{aligned} h(x,z(x)) = h(1,\rho_0) &+ (x-1)\frac{\partial h}{\partial x} + (z(x)-\rho_0)\frac{\partial h}{\partial z} + \frac{1}{2}(x-1)^2\frac{\partial^2 h}{\partial x^2} \\ &+ (x-1)(z(x)-\rho_0)\frac{\partial^2 h}{\partial x\partial z} + \frac{1}{2}(z(x)-\rho_0)^2\frac{\partial^2 h}{\partial z^2} \\ &+ O(d(x-1)^3) + O(d(z(x)-\rho_0)^3). \end{aligned}$$

The values of the derivatives of h in this expansion are taken at point $(1, \rho_0)$. For $x = 1 + \varepsilon$ and $z(x) = (1 + u)\rho_0$ (see (16)), we get that

$$h(x, z(x)) - h(1, \rho_0) = \frac{\partial h}{\partial x}\varepsilon + \frac{1}{2}\left(\frac{\partial^2 h}{\partial x^2} - 2\alpha\rho_0\frac{\partial^2 h}{\partial x \partial z} + \alpha^2\rho_0^2\frac{\partial^2 h}{\partial z^2}\right)\varepsilon^2 + O(d\varepsilon^3).$$

Define $\mu = \partial h/\partial x$ and $\sigma^2 = \mu + \partial^2 h/\partial x^2 - 2\alpha\rho_0\partial^2 h/\partial x\partial z + \alpha^2\rho_0\partial^2 h/\partial z^2$; the values of the derivatives in μ and σ^2 are taken at point $(1, \rho_0)$. We have that

$$\log\frac{\psi(x)}{\psi(1)} = \mu(x-1) + \frac{1}{2}(\sigma^2 - \mu)(x-1)^2 + O(d(x-1)^3) + O(x-1).$$

For $s, d_X \to +\infty$, we have that $\mu = r\mu_0(1 + o(1))$ and $\sigma^2 = r\sigma_0^2(1 + o(1))$, with the constants μ_0 and σ_0 defined in function of $\rho = g^{-1}(B)$ as follows:

$$\mu_0 = 1 - \frac{1}{\lambda(\rho)}, \qquad \sigma_0^2 = \frac{\lambda(\rho) - 1}{\lambda^2(\rho)} - \frac{r}{d}\frac{\rho\lambda'^2(\rho)}{\lambda^4(\rho)g'(\rho)}.$$

We again note that σ_0^2 is strictly positive: $\sigma_0^2 \geq (\lambda(\rho)-1)/\lambda^2(\rho) - \rho\lambda'^2(\rho)/\lambda^4(\rho)g'(\rho)$, and we proved in §6.2.3 that this term is strictly positive. We finally get that $t\mu/\sigma + \log\psi(e^{-t/\sigma})/\psi(1) = t^2/2 + O(d/r^{3/2}) + O(1/r) + o(1)$. The error term becomes $o(1)$ for r such that $r \to +\infty$ and $r^{3/2}/d \to +\infty$. □

6.3.4. Proofs of Corollaries 2 and 3. The proof of Corollary 2 is adapted from the proof of Theorem 3, as Corollary 1 was obtained from Theorem 1 in §6.2.4: Take a sequence of functions $\lambda_{d_Z}(t)$ and note that $z(x)$ can be restricted to a compact subset near $g^{-1}(B)$. Corollary 3 is simply Theorem 3 applied to the function $\lambda(z) = e^z$.

6.4. Proof of Theorem 4 for semijoins: X key of S. The generating function $\Phi(x, y, z)$ has the following general form:

$$\Phi(x, y, z) = (\lambda(y) + z\lambda(xy))^d.$$

We first extract the coefficient of z^s in $\Phi(x, y, z)$ as follows:

$$[z^s]\Phi(x, y, z) = \binom{d}{s}\lambda(xy)^s\lambda(y)^{d-s}.$$

Cauchy's formula then gives the following coefficient of y^r:

$$\psi(x) = \frac{1}{\binom{d}{s}}[y^r z^s]\Phi(x, y, z) = \frac{1}{2i\pi}\oint e^{h(x,y)}dy,$$

with

$$h(x, y) = (d - s)\log\lambda(y) + s\log\lambda(xy) - (r + 1)\log y.$$

Here again, we choose for integration path a circle centered at the origin and that has for radius the root $y(x)$ of equation $\partial h/\partial y(x, y) = 0$.

6.4.1. Evaluation of the saddlepoint $y(x)$. For $x = 1$, $y(x)$ is solution of $\partial h/\partial y(1, y) = 0$. We have that $h(1, y) = d \log \lambda(y) - (r+1) \log y$. This gives

$$y \frac{\lambda'(y)}{\lambda(y)} = \frac{r+1}{d}. \tag{17}$$

Define again $g(y) = y\lambda'(y)/\lambda(y)$. Equation (17) can be written as $g(y) = (r+1)/d$. As function λ satisfies Property $\mathcal{P}$, the equation $g(y) = (r+1)/d$ has a unique solution ρ_0 if and only if (cf. Lemma A in §3.4)

$$\lim_{y \to +\infty} y \frac{\lambda'}{\lambda}(y) > A.$$

We then solve (17) for $x = 1 + \varepsilon$ and $y(x) = (1+u)\rho_0$. They satisfy the equation

$$(d-s)g(y) + s\, g(xy) - (r+1) = 0. \tag{18}$$

Function g can be expanded near ρ_0, as follows:

$$g(y) = g(\rho_0) + (y - \rho_0) g'(\rho_0) + O(||g''||(y-\rho_0)^2).$$

The error term is simply $O((y-\rho_0)^2)$, and we have for $y = (1+u)\rho_0$ that

$$g(y) = g(\rho_0) + g'(\rho_0) u \rho_0 + O(u^2).$$

As $xy = (1 + \varepsilon + u + \varepsilon u)\rho_0$, we get that

$$g(xy) = g(\rho_0) + g'(\rho_0)(\varepsilon + u)\rho_0 + O(\varepsilon^2) + O(u^2).$$

Equation (18) can be simplified by using $r + 1 = d g(\rho_0)$, and we get the following approximate equation between ε and u:

$$du + s\varepsilon + O(du^2) + O(s\varepsilon^2) = 0.$$

We have that $s < d$, and we can solve this equation in u. This gives the following approximate value of the saddlepoint for $x = 1 + \varepsilon$:

$$y(x) = (1+u)\rho_0, \qquad u = -\frac{s}{d}\varepsilon(1 + O(\varepsilon)). \tag{19}$$

We indicate below the values of derivatives of h that we need later: $\partial h/\partial y(1, \rho_0) = 0$, the derivatives of order 3 of h near $(1, \rho_0)$ are $O(d)$, and

$$\frac{\partial h}{\partial x}(1, \rho_0) = s g(\rho_0), \qquad \frac{\partial^2 h}{\partial x^2}(1, \rho_0) = s(\frac{\lambda'}{\lambda})'(\rho_0)\rho_0^2 = s(g'(\rho_0)\rho_0 - g(\rho_0)),$$
$$\frac{\partial^2 h}{\partial x \partial y}(1, \rho_0) = s g'(\rho_0), \qquad \frac{\partial^2 h}{\partial y^2}(1, \rho_0) = d\frac{g'(\rho_0)}{\rho_0}.$$

6.4.2. Approximation of $\psi(x)$. We take here x fixed, real, and smaller than 1. The function $\psi(x) = [y^r z^s]\Phi(x, y, z)/\binom{d}{s}$ can be written as an integral along a circle of center the origin and radius $y(x) = (1+u)\rho_0$, below:

$$\psi(x) = \frac{1}{2i\pi} \oint e^{h(x,y)} dy = \frac{1}{2i\pi} \int_{\theta \in [-\pi, +\pi]} e^{h(x, y(x)e^{i\theta})} d(y(x)e^{i\theta}).$$

For α in $]0,\pi[$, we define

$$I_1 = \frac{1}{2i\pi}\int_{\theta\in]-\alpha,+\alpha[} e^{h(x,y(x)e^{i\theta})}d(y(x)e^{i\theta}),$$
$$I_2 = \frac{1}{2i\pi}\int_{\alpha\le|\theta|\le\pi} e^{h(x,y(x)e^{i\theta})}d(y(x)e^{i\theta}).$$

Evaluation of I_1. We check that the assumptions of Lemma B of §6.2.2 are satisfied: The conditions on the derivatives of h hold, and $h(x,y)$ has the following expansion for y near the saddlepoint $y(x)$:

$$h(x,y) = h(x,y(x)) + (y-y(x))\frac{\partial h}{\partial y}(x,y(x)) + 1/2\,(y-y(x))^2\frac{\partial^2 h}{\partial y^2}(x,y(x))$$
$$+O(||h'''||(y-y(x))^3).$$

This gives

$$h(x,y(x)e^{i\theta}) = h(x,y(x)) + \frac{y(x)^2}{2}(e^{i\theta}-1)^2 h''_{y^2}(x,y(x)) + O(d\theta^3).$$

Lemma B then proves that

$$I_1 = \frac{e^{h(x,y(x))}}{\sqrt{2\pi h''_{y^2}(x,y(x))}}(1+O(\alpha^2\sqrt{d}e^{-\gamma_0 d\alpha^2}) + O(e^{-\gamma_1 d\alpha^2}) + O(d\alpha^3)). \tag{20}$$

Upper bound on I_2. We have that

$$I_2 = \frac{e^{h(x,y(x))}}{2i\pi}\int_{\alpha\le|\theta|\le\pi}\left(\frac{\lambda(y(x)e^{i\theta})}{\lambda(y(x))}\right)^{d-s}\cdot\left(\frac{\lambda(xy(x)e^{i\theta})}{\lambda(y(x))}\right)^{s}\cdot e^{-i(r+1)\theta}d\theta.$$

Lemma C in §6.2.2 shows that there exists a suitable constant $\gamma > 0$ such that the following inequalities hold:

$$|\lambda(y(x)e^{i\theta})| \le \lambda(y(x))e^{-\gamma\alpha^2}, \qquad |\lambda(xy(x)e^{i\theta})| \le \lambda(xy(x))e^{-\gamma\alpha^2}.$$

As a consequence,

$$\int_{\alpha\le|\theta|\le\pi}\left|\frac{\lambda(y(x)e^{i\theta})}{\lambda(y(x))}\right|^{d-s}\cdot\left|\frac{\lambda(xy(x)e^{i\theta})}{\lambda(y(x))}\right|^{s}\cdot d\theta \le 2\pi e^{-\gamma d\alpha^2}.$$

We get that

$$I_2 = \frac{e^{h(x,y(x))}}{\sqrt{2\pi h''_{y^2}(x,y(x))}}O(\sqrt{d}e^{-\gamma d\alpha^2}). \tag{21}$$

Choice of α. As usual, we choose $\alpha = (\log d)/\sqrt{d}$; the error terms in (20) and (21) then become $o(1)$ for $r,d\to+\infty$. Hence

$$\psi(x) = \frac{e^{h(x,y(x))}}{\sqrt{2\pi h''_{y^2}(x,y(x))}}(1+o(1)).$$

6.4.3. Convergence of the Laplace transform. We show here that $e^{t\mu/\sigma}\psi(e^{-t/\sigma})/\psi(1)$ converges toward $e^{t^2/2}$ for $d \to +\infty$ and for all fixed real positive t. Its logarithm is $\Xi(t) = t\mu/\sigma + \log(\psi(e^{-t/\sigma})/\psi(1))$. For $x = 1+\varepsilon$ and $y = (1+u)\rho_0$, and using the information on the order of the derivatives of h, we get that

$$h(x,y) = h(1,\rho_0) + \frac{\partial h}{\partial x}(1,\rho_0)\varepsilon + \frac{\partial h}{\partial y}(1,\rho_0)u\rho_0 + \frac{1}{2}\frac{\partial^2 h}{\partial x^2}(1,\rho_0)\varepsilon^2 + \frac{\partial^2 h}{\partial x \partial y}(1,\rho_0)\varepsilon u\rho_0 + \frac{1}{2}\frac{\partial^2 h}{\partial y^2}(1,\rho_0)u^2\rho_0^2 + O(du^3) + O(d\varepsilon^3).$$

We substitute $-s\varepsilon/d\cdot(1+O(\varepsilon))$ for u (see (19)), and the values computed above for the derivatives of h, and we get that

$$h(1+\varepsilon,\rho_0(1+u)) - h(1,\rho_0) = sg(\rho_0)\varepsilon + \frac{s}{2}\left(\left(1-\frac{s}{d}\right)g'(\rho_0)\rho_0 - g(\rho_0)\right)\varepsilon^2 + O(d\varepsilon^3).$$

We have, as usual, that $(\partial^2 h/\partial y^2(x,\rho_0(x)))/(\partial^2 h/\partial y^2(1,\rho_0)) = O(\varepsilon)$. For $x = e^{-t/\sigma}$, we then substitute $-t/\sigma + t^2/2\sigma^2 + O(1/\sigma^3)$ for $\varepsilon = x-1$, and we obtain that

$$\Xi(t) = (\mu - sg(\rho_0))\frac{t}{\sigma} + s(1-s/d)g'(\rho_0)\rho_0\frac{t}{2\sigma^2} + O\left(\frac{d}{\sigma^3}\right) + O\left(\frac{1}{\sigma}\right).$$

Define $\mu = sg(\rho) = As \approx rs/d$ and $\sigma^2 = s(1-s/d)\rho g'(\rho)$, with $\rho = g^{-1}(A)$. The conditions on s and d show that the error terms are $o(1)$ and we have that $\Xi(t) \to e^{t^2/2}$. □

6.4.4. Proofs of Corollaries 4 and 5. The proof of Corollary 4 is adapted from that of Theorem 4, as indicated in §6.2.4 for Corollary 1 and Theorem 1. Corollary 5 is an instance of Theorem 4 in the case when $\lambda(y) = e^y$.

6.5. Proof of Theorem 5. Theorem 5 cannot be deduced from either Theorem 3 or Theorem 4: The functions $\lambda_R(t)$ and $\lambda_S(t)$ are both equal to $1+t$. However, the function $\Phi(x,y,z)$ is simple enough that it is possible to write a direct proof. We can express $[y^r z^s]\Phi(x,y,z)$ as a sum of binomial coefficients: For $\Phi(x,y,z) = (1+y+z+xyz)^d$, we have that

$$[y^r z^s]\Phi(x,y,z) = \sum_{i+j=s}\binom{d}{r}\binom{d-r}{i}\binom{r}{j}x^j.$$

We can then try a direct study based on properties of the binomial coefficients. We do not follow this idea, but rather indicate briefly how Theorem 5 can be proved by our approach.

We compute $[y^r]\Phi$, then apply Cauchy's formula to get $[y^r z^s]\Phi$ as follows:

$$[y^r z^s]\Phi(x,y,z) = \frac{\binom{d}{r}}{2i\pi}\oint e^{h(x,z)}dz,$$

with $h(x,z) = (d-r)\log(1+z) + r\log(1+xz) - (s+1)\log z$.

The saddlepoint for $x=1$ is $\rho_0 = (s+1)/(d-s-1)$. We must assume that $s = Bd + o(d)$ if we want ρ_0 to stay in a compact subset of $]0,+\infty[$. For $x = 1+\varepsilon$, the saddlepoint is $z(x) = \rho_0(1 - r\varepsilon/d + O(r\varepsilon/d))$.

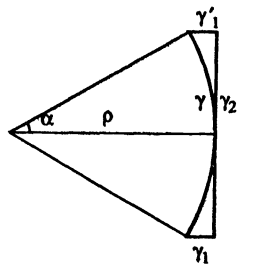
As usual, the computation of the integral $\int_{[-\pi,+\pi]} e^{h(x,z(x)e^{i\theta}}d(z(x)e^{i\theta})$ is performed in two parts. The computation of the main part, on interval $[-\alpha,+\alpha]$, is

straightforward, and we do not detail it. The upper bound on the remainder of the integral once again relies on the bound on a function of θ, for $\alpha \leq |\theta| \leq \pi$. In the present case, this function is simply $(1+z(x)e^{i\theta})^{d-r}(1+xz(x)e^{i\theta})^r/(1+z(x))^d$, and the desired inequality presents no difficulty. The evaluation of the normalized Laplace transform and its convergence toward $e^{t^2/2}$ are then easily proved. □

Appendix A. Proof of Lemma B. We write h instead of h_d, and h'' instead of $\partial^2 h_d/\partial y^2$. Let $\alpha_0 > 0$ be such that the Taylor expansion of h is valid for $|\theta| \leq \alpha_0$. Then, for any $\alpha \leq \alpha_0$, the assumptions on h show that

$$\frac{1}{2i\pi}\int_{\theta\in]-\alpha,+\alpha[} e^{h(x,\rho e^{i\theta})}d(\rho e^{i\theta}) = \frac{e^{h(x,\rho)}}{2i\pi}\int_{|\theta|<\alpha} e^{(\rho^2/2)(e^{i\theta}-1)^2 h''(x,\rho)+O(d\theta^3)}d(\rho e^{i\theta})$$
$$= \frac{e^{h(x,\rho)}}{2i\pi}\int_{|\theta|<\alpha} e^{(\rho^2/2)(e^{i\theta}-1)^2 h''(x,\rho)}d(\rho e^{i\theta})\ (1+O(d\alpha^3)).$$

Let us define $J = \int_{|\theta|<\alpha} e^{(\rho^2/2)\cdot(e^{i\theta}-1)^2 h''(x,\rho)}d(\rho e^{i\theta})$. The integration path $\gamma = \{|\theta| < \alpha\}$ is part of a circle of radius ρ. We replace it by the path $\gamma_1 \cup \gamma_2 \cup \gamma'_1$ defined as follows: $\gamma_1 = \{y = \rho(1-v) - i\rho\sin\alpha\}$, $\gamma'_1 = \{y = \rho(1-v) + i\rho\sin\alpha\}$ for $v \in [0, 1-\cos\alpha]$, and $\gamma_2 = \{y = \rho + i\rho t\}$, for $t \in [-\sin\alpha, +\sin\alpha]$. See the figure below:



Let J_1, J'_1, and J_2 be the integrals on γ_1, γ'_1, and γ_2: $J = J_1 + J_2 + J'_1$. We first show, below, that the integrals J_1 and J'_1 can be neglected as follows:

$$J_1 = \int_{\gamma_1} e^{(1/2)(y-\rho)^2 h''(x,\rho)}dy$$
$$= -\rho\int_{1-\cos\alpha}^{0} e^{(\rho^2/2)(v+i\sin\alpha)^2 h''(x,\rho)}dv$$
$$= \rho\int_{0}^{1-\cos\alpha} e^{(\rho^2/2)(v+i\sin\alpha)^2 h''(1,\rho_0)(1+o(1))}dv.$$

Hence $|J_1| \leq \rho\int_0^{1-\cos\alpha} e^{\Re\{\rho^2/2\cdot(v+i\sin\alpha)^2 h''(1,\rho_0)(1+o(1))\}}dv$.

The former integral is $O(\int_0^{1-\cos\alpha} e^{\rho^2/2\cdot(v^2-\sin^2\alpha)h''(1,\rho_0)}dv)$, and

$$\int_0^{1-\cos\alpha} e^{(\rho^2/2)(v^2-\sin^2\alpha)h''(1,\rho_0)}dv = e^{-(\rho^2/2)h''(1,\rho_0)\sin^2\alpha}\int_0^{1-\cos\alpha} e^{(\rho^2/2)v^2 h''(1,\rho_0)}dv.$$

We also have that $\int_0^{1-\cos\alpha} e^{\rho^2/2\cdot v^2 h''(1,\rho_0)}dv \leq (1-\cos\alpha)e^{\rho^2/2\cdot h''(1,\rho_0)(1-\cos\alpha)^2}$. As $h''(1,\rho_0) = \Theta(d)$, this gives for a suitable constant $\gamma_0 > 0$: $J_1 = O(\alpha^2 e^{-\gamma_0 d\alpha^2})$. The

majoration of J_1' is done in the same way. We now show that J_2 gives the main term of J, as follows:

$$\begin{aligned} J_2 &= \int_{\gamma_2} e^{(1/2)(y-\rho)^2 h''(x,\rho)} dy \\ &= i\rho \int_{|t|\le \sin\alpha} e^{-(\rho^2/2)t^2 h''(x,\rho)} dt \\ &= \frac{i}{\sqrt{h''(x,\rho)}} \int_{|v|\le \rho\sqrt{h''(x,\rho)}\sin\alpha} e^{-v^2/2} dv. \end{aligned}$$

This last integral is equal to

$$\int_{-\infty}^{+\infty} e^{-v^2/2} dv - \int_{|v|>\rho\sqrt{h''(x,\rho)}\sin\alpha} e^{-v^2/2} dv = \sqrt{2\pi} + O(e^{-(\rho^2/2)h''(x,\rho)}\sin^2\alpha).$$

This gives $J = i\sqrt{2\pi/h''(x,\rho)}(1 + O(e^{-\gamma_1 d\alpha^2}) + O(\alpha^2\sqrt{d}e^{-\gamma_0 d\alpha^2}))$, with γ_0 and γ_1 strictly positive constants; in particular, they are independent of r and d. We then plug the approximation of J into the expression of the integral to get Lemma B. □

Appendix B. Proof of Lemma C. The main idea in proving Lemma C is a classical one, namely, that the modulus of an analytical function with positive coefficients on a circle of given radius $y > 0$ attains its maximum at point y and only at that point, except when the function is actually a function of y^m for some positive integer m

$$|\lambda(ye^{i\theta})| = |\sum_{n\ge 0} \lambda_n y^n e^{in\theta}| \le \sum_{n\ge 0} \lambda_n y^n = \lambda(y).$$

We want to extend it to get a uniform upper bound $|\lambda(ye^{i\theta})| \le \lambda(y)(1 - C\alpha^2)$, for $\alpha \le |\theta| \le \pi$. We note that, if $\lambda(y) = \Lambda(y^2)$, for example, $|\lambda(ye^{i\theta})|$ would attain its maximum at both points y and $ye^{i\pi}$, and it would not be possible to get an inequality $|\lambda(ye^{i\theta})| \le \lambda(y)(1 - C\alpha^2)$ on most of the circle of radius y, excluding only an arc near y. We first give the proof of Lemma C in a simple case, then the general argument.

Define $\lambda_n = [y^n]\lambda(y)$ for $n \ge 0$. From Property $\mathcal{P}$ of §3.4, we know that $\lambda_0 = 1$. We first assume that $\lambda_1 \ne 0$. We rewrite $\lambda(ye^{i\theta})$ as

$$\lambda(ye^{i\theta}) = (1 + \lambda_1 ye^{i\theta}) + (\lambda(ye^{i\theta}) - 1 - \lambda_1 ye^{i\theta}).$$

The triangular inequality gives

$$|\lambda(ye^{i\theta})| \le |1 + \lambda_1 ye^{i\theta}| + |\lambda(ye^{i\theta}) - 1 - \lambda_1 ye^{i\theta}|.$$

We also have that $\lambda(ye^{i\theta}) - 1 - \lambda_1 ye^{i\theta} = \sum_{n\ge 2} \lambda_n y^n e^{in\theta}$. As the coefficients λ_n are real and positive, we get that

$$|\lambda(ye^{i\theta}) - 1 - \lambda_1 ye^{i\theta}| \le \sum_{n\ge 2} \lambda_n y^n = \lambda(y) - (1 + \lambda_1 y). \tag{22}$$

We can also write

$$|1 + \lambda_1 ye^{i\theta}|^2 = (1 + \lambda_1 y)^2 \left(1 - 2\frac{\lambda_1 y}{(1 + \lambda_1 y)^2}(1 - \cos\theta)\right).$$

This can be simplified by using the fact that $\sqrt{1-t} \leq 1 - t/2$ for $t \in [0,1]$; we get that

$$|1 + \lambda_1 y e^{i\theta}| \leq (1 + \lambda_1 y)\left(1 - \frac{\lambda_1 y}{(1+\lambda_1 y)^2}(1 - \cos\theta)\right).$$

Now, for $|\theta|$ in the interval $[\alpha, \pi]$, we have that $\cos\theta \leq \cos\alpha$; this gives the following inequality valid on $[\alpha, \pi]$, for a strictly positive constant C_0 that can be chosen independent of y and of α (remember that y varies in a compact subset of $]0, +\infty[$):

$$|1 + \lambda_1 y e^{i\theta}| \leq 1 + \lambda_1 y - C_0\alpha^2. \tag{23}$$

Combined with bound (22), this gives, in turn, the following bound on $|\lambda(ye^{i\theta})|$:

$$|\lambda(ye^{i\theta})| \leq \lambda(y) - C_0\alpha^2 \leq \lambda(y)(1 - C\alpha^2).$$

The term C in this last inequality is positive and can again be chosen so as to be independent of y.

When the term λ_1 is equal to zero, inequality (23) does not hold, and the former proof must be adapted as follows. By Property $\mathcal{P}$, there exists a finite set of indices K such that, for all $k \in K$, $\lambda_k \neq 0$, and that $GCD(k|k \in K) = 1$. Hence there exist relative numbers a_k such that $\sum_k k\, a_k = 1$. Let us define $c = 1/(2\sum_k |a_k|)$; we can assume that α is close enough to zero for the inequality $c\alpha < \pi$ to hold. Let $\theta \in [\alpha, \pi]$ (the case where θ belongs to $[-\pi, -\alpha]$ is symmetrical). Then $\theta = \sum_k a_k k\theta$, and we can show that there is at least one indice $k = k(\theta)$ in K such that $|k\theta[2\pi]| \geq c\alpha$. Assume that it is not the case; then each of the $\eta_k = k\theta[2\pi]$ satisfies $|\eta_k| < c\alpha$; hence $|\theta| = |\sum_k a_k \eta_k| \leq \sum_k |a_k \eta_k| < (\sum_k |a_k|)c\alpha$ and $|\theta| \leq \alpha/2$, which does not hold. We now decompose $\lambda(ye^{i\theta})$ according to the indice $k = k(\theta)$ as follows:

$$\lambda(ye^{i\theta}) = (\lambda(ye^{i\theta}) - 1 - \lambda_k y^k e^{ik\theta}) + (1 + \lambda_k y^k e^{i\theta k}).$$

Hence

$$|\lambda(ye^{i\theta})| \leq \lambda(y) - 1 - \lambda_k y^k + |1 + \lambda_k y^k e^{ik\theta}|.$$

We have that

$$|1 + \lambda_k y^k e^{ik\theta}| \leq (1 + \lambda_k y^k)\left(1 - \frac{\lambda_k y^k}{(1+\lambda_k y^k)^2}(1 - \cos k\theta)\right).$$

As $k\theta[2\pi]$ is at a distance at least $c\alpha$ from 0, we can write

$$|\lambda(ye^{i\theta})| \leq \lambda(y) - \frac{\lambda_k y^k}{1 + \lambda_k y^k}(1 - \cos c\alpha).$$

To get a bound independent of θ, we must remove the dependency of the indice k on θ. This can be done by noting that, as y belongs to a compact set of the reals, the function $a(y) = \min\{\lambda_k y^k : k \in K\}$ is bounded away from zero. This shows the existence of a constant C, independent of θ, such that, for all relevant y and θ, $|\lambda(ye^{i\theta})| \leq \lambda(y)(1 - C\alpha^2)$. □

We should point out that, although this is ruled out by our assumptions, there is no difficulty in getting a bound similar to that of Lemma C when function $\lambda(t)$ is affine, of the form $\lambda_0 + \lambda_1 t$, with positive coefficients λ_0 and λ_1. The key condition of our proof is the positivity of the coefficients.

Acknowledgments. The author thanks P. Flajolet for many stimulating discussions on asymptotic distributions, R. Schott for carefully reading a preliminary version of this paper, and an anonymous referee for suggestions that led to a clarification of the original paper, most notably a simpler proof of Lemma C.

REFERENCES

[1] L. AMMANN, *Some limit theorems for clustered occupancy models*, J. Appl. Probab., 20 (1983), pp. 788–802.
[2] E. BENDER, *Central and local limit theorems applied to asymptotic enumeration*, J. Combin. Theory (A), 15 (1973), pp. 91–111.
[3] N. BLEISTEIN AND R. HANDELSMAN, *Asymptotic Expansions of Integrals*, Dover, New York, 1986.
[4] E. R. CANFIELD, *Central and local limit theorems for the coefficients of polynomials of binomial type*, J. Combin. Theory (A), 23 (1977), pp. 275–290.
[5] L. COMTET, *Analyse combinatoire*, Presses Universitaires de France, Paris, 1970.
[6] N. G. DEBRUIJN, *Asymptotic Methods in Analysis*, Dover, New York, 1981.
[7] W. FELLER, *An Introduction to Probability Theory and its Applications*, Vol. 2, John Wiley, New York, 1971.
[8] P. FLAJOLET AND M. SORIA, *Gaussian limiting distributions for the number of components in combinatorial structures*, J. Combin. Theory (A), 53 (1990), pp. 165–182.
[9] D. GARDY, *Bases de données, Allocations aléatoires: Quelques analyses de performances*, Thèse d'Etat, Université de Paris-Sud, Paris, June 1989.
[10] ———, *Join sizes, urn models and normal limiting distributions*, Tech. Report 600, Laboratoire de Recherche en Informatique, Université de Paris-Sud, Paris, October 1990.
[11] D. GARDY AND C. PUECH, *On the sizes of projections: A generating function approach*, Inform. Systems, 9 (1984), pp. 231–235.
[12] ———, *On the effect of join operations on relation sizes*, ACM Trans. Database Systems, 14 (1989), pp. 574–603.
[13] B. GNEDENKO AND A. KOLMOGOROV, *Limit Distributions for Sums of Independent Random Variables*, Addison–Wesley, Reading, MA, 1954.
[14] P. HENRICI, *Applied and Computational Analysis*, Vol. 2, John Wiley, New York, 1977.
[15] M. JARKE AND J. KOCH, *Query optimization in database systems*, ACM Comput. Surveys, 16 (1984), pp. 111–152.
[16] N. JOHNSON AND S. KOTZ, *Urn Models and Their Application*, John Wiley, New York, 1977.
[17] V. KOLCHIN, B. SEVAST'YANOV, AND V. CHISTYAKOV, *Random Allocations*, John Wiley, New York, 1978.
[18] D. MAIER, *The Theory of Relational Databases*, Computer Science Press, New York, 1983.
[19] M. V. MANNINO, P. CHU, AND T. SAGER, *Statistical profile estimation in database systems*, ACM Comput. Surveys, 20 (1988), pp. 191–221.
[20] J. ULLMAN, *Principles of Database Systems*, Computer Science Press, New York, 1980.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 249–258, May 1992
© 1992 Society for Industrial and Applied Mathematics
007

TRIANGULATING 3-COLORED GRAPHS*

SAMPATH K. KANNAN† AND TANDY J. WARNOW‡

Abstract. The problem of determining whether a vertex-colored graph can be triangulated without introducing edges between vertices of the same color is what is of interest here. This problem is known to be polynomially equivalent to a fundamental problem in numerical taxonomy called the *perfect phylogeny problem*, which is concerned with the inference of evolutionary history. This problem is also related to the problem of recognizing partial k-trees, a class of graphs that has received much attention recently. The problem in its general form is NP-complete and can be solved in $O(n^{k+1})$ time, where n is the number of vertices and k the number of colors. In this paper, a linear time algorithm for the case of 3-colored graphs is presented.

Key words. graph colorings, chordal graphs, NP-complete, polynomial algorithms, partial k-trees, numerical taxonomy

AMS(MOS) subject classifications. 05C05, 05C75, 05C15, 68C05, 92A10, 92A12

1. Introduction.

The problem we consider in this paper is the following.

TRIANGULATING COLORED GRAPH PROBLEM (TCG). *Given a graph $G = (V, W)$ with a vertex coloring $c : V \to \{1, 2, \cdots, |V|\}$, determine whether we can triangulate it without introducing edges between vertices of the same color.* (*A* triangulated *graph, also known as a* chordal *graph, is a graph that has no induced cycles of length four or greater* [9].) *If such a triangulation exists, we will refer to it as a c-triangulation and say that G is c-triangulatable.*

TCG is polynomially equivalent to a problem in evolutionary tree inference called the perfect phylogeny problem (PP) [12]. An evolutionary tree (also called a *phylogeny*) for a set S of species is a rooted tree in which the leaves represent the species in S, and internal nodes represent ancestral species. In this model, the species in S are described by their values on a set of *qualitative characters*. A *qualitative character*, or *character*, is simply an equivalence relation on the species set, describing a partition of the species set into the distinct equivalence classes, which are referred to as *character states*. These characters can arise from morphological evidence (for example, the binary character having two states *vertebrate-invertebrate*), from molecular data (such as the character specifying the base appearing in the ith position of a DNA sequence), or in some other way. Thus the species can be described by an $n \times k$ matrix M, where there are n species and and k characters, and M_{ij} is the state of the ith species for the jth character. Each species s in S is represented by an integer k-vector v_s. A proposed phylogeny for the set S is therefore a vertex-labelled tree, in which the node corresponding to the ith species is labelled by the ith row of the matrix M, and the remaining (internal) nodes are also labelled with k-vectors.

The desired phylogeny T (if it exists) will have the property that, for each state of each character, the set of nodes in T having that state forms a connected component (i.e., a subtree). Such a phylogeny is said to be *perfect*, and when a perfect phylogeny exists, the characters are said to be *perfectly compatible* on the species set S. The problem of whether a perfect phylogeny exists is called the *perfect phylogeny problem.*

* Received by the editors August 15, 1990; accepted for publication (in revised form) April 26, 1991.

† Department of Computer Science, University of Arizona, Tuscon, Arizona 85721. This author's work was supported by National Science Foundation grant CCR-9108969.

‡ Department of Mathematics, University of Southern California, Los Angeles, California, 90089. This work was supported by National Science Foundation grants DMS90-05833 and CCR87-04184.

The complexity of PP (and hence of TCG) remained open for a long time (the reduction of PP to TCG was proved by Buneman [3] in 1974, and PP has been widely discussed since the 1960s). Essentially, the only progress on the two problems was to show that many variations were NP-complete [4]–[6], [8], [10] and to solve the special cases of binary character compatibility [11] and pairwise compatibility [7]. Recently, Bodlaender, Fellows, and Warnow [2] showed TCG and PP to be NP-complete. McMorris, Warow, and Wimer [13] found an $O(n^{k+1})$ algorithm for TCG (where n is the number of vertices, and k the number of colors). For 3-colored graphs, this yields an $O(n^4)$ algorithm. A polynomial time algorithm for PP exists for the case where the number of states per character is limited to four [12]. This algorithm can be used to construct phylogenies from DNA sequences.

In the reduction from PP to TCG, the number of characters in the instance of PP translates to the number of colors in the instance of TCG. Typical inputs to PP involve many more than three characters. Hence the special case of TCG that we solve—that of determining whether a 3-colored graph can be made chordal—has no direct relevance to solving practical instances of PP. However, it provides biologists with the ability to tell if a subset of three characters is mutually compatible. Biologists currently use a procedure for pairwise compatibility of characters to eliminate characters that are incompatible with many other characters [7]. A test for 3-way compatibility should further enhance this tool.

More interesting is the connection between TCG and the recognition of partial k-trees. (For a definition of k-trees and partial k-trees, see Arnborg, Corneil, and Proskurowski [1].) The connection between these two problems comes about because of the observation that a $(k+1)$-colored graph G can be c-triangulated if and only if G can be c-triangulated into a k-tree. Thus, if G can be c-triangulated, it is a "colored version" of a partial k-tree. Using this observation, McMorris, Warnow, and Wimer [13] have produced an algorithm based upon the partial k-tree recognition algorithm of [1], which permits us to determine whether a k-colored graph can be triangulated in $O(n^{k+1})$ time. Note that this algorithm takes time $O(n^4)$ for 3-colored graphs.

In §2 we state definitions and results that we use in the remaining sections. In §3 we present our algorithm for the case of 3-colored graphs. The algorithm relies on structural properties of 3-colored graphs that are c-triangulatable. Some care must be taken in its implementation to achieve linear time. In §4 we discuss the other known results and the remaining open problems.

2. Preliminary definitions and notations. A vertex coloring of the graph $G = (V, E)$ with the property that no edge connects vertices of the same color is said to be *proper*. A properly colored graph $G = (V, E)$ with coloring $c : V \to \{1, 2, \cdots, k\}$ is said to be c-triangulatable if there exists a chordal graph $G' = (V, E')$, where $E \subset E'$ and c is proper on G'.

By a "coloring" of a graph G, we mean a proper vertex coloring. The neighbor set of a vertex v is denoted by $\Gamma(v)$.

A useful fact about chordal graphs is that every chordal graph has a *perfect elimination scheme*. A perfect elimination scheme for a graph $G = (V, E)$ is an ordering of the vertices of G, $v_1 < v_2 < \cdots < v_n$ such that, for each i, the neighbors of v_i that follow v_i in the ordering form a clique. In other words, if we removed vertices from G in the perfect elimination order, the neighbor set of each vertex at the time of its removal is a clique.

If G' is a c-triangulation of G, then G' will have a perfect elimination scheme $P = v_1 < v_2 < \cdots < v_n$.

We can use the perfect elimination scheme for G' to triangulate G by making a clique out of the neighbors of v_i that follow v_i in the ordering P. We will therefore refer to this ordering as an *elimination scheme* for G.

In this context, it is useful to define a sequence of graphs that arise when applying the elimination scheme to G. Let $G_0 = G$ and let G_i be obtained from G_{i-1} by removing vertex v_i and forming a clique out of its neighbors in G_{i-1}. If we define the graph $G'' = (V, E'')$ to be the graph on vertex set V with $E'' = \cup_{i=0}^{n} E(G_i)$, then $G \subset G'' \subset G'$, and G'' is a c-triangulated graph. We will call this graph G'' the *filled graph* defined by the perfect elimination order P and graph G. We will also refer to the sequence of graphs $G_0, G_1, \cdots, G_n$ as the *elimination sequence* of G.

3. A linear time algorithm for 3-colored graphs. We assume that $G = (V, E)$ is a graph with a proper 3-coloring. We present an algorithm that will determine whether G can be c-triangulated, and if so, will produce the triangulation.

We can assume that G is 2-connected, since G can be c-triangulated if and only if all of its biconnected components can be c-triangulated. Let us define a *simplicial* vertex to be a vertex v whose neighbor set is a clique. We can further simplify G by removing simplicial vertices from V. Since G is 3-colored, a simplicial vertex has degree at most 2. Any graph G with no simplicial vertices will be called a *reduced* graph.

The algorithm works on each biconnected component and proceeds by repeatedly finding cycles to triangulate in the graph, and then removing simplicial vertices. The choice of cycle to triangulate is based upon the following observation.

We will call a cycle γ a *feasible cycle* if (1) all but two of its vertices have degree two, and (2) the remaining two vertices a and b have neighbors outside the cycle. We will refer to the vertices with neighbors outside the cycle as *port vertices.* If the graph can be c-triangulated, then the algorithm will triangulate this cycle γ by first introducing the edge between the port vertices a and b and then completing the c-triangulation of γ by using the algorithm of Theorem 1 on each of the two new cycles created by the addition of the edge (a, b).

Note that any 3-colored graph G that has a c-triangulation is also a partial 2-tree, and thus the number of edges must be $\leq 2n - 2$, where n is the number of vertices. Thus we can count the number of edges in the graph and reject it if the number exceeds that bound.

We now describe how to determine whether a given 2-connected graph G has a c-triangulation, where we have bounded the number of edges by $2n - 2$.

Our algorithm proceeds as follows.

Algorithm.

```
begin
  while V ≠ ∅ do
    Remove simplicial vertices until no simplicial vertices remain
    If V = ∅ then return Yes
    if G is a cycle then
      if all three colors are present then use the
        algorithm of Theorem 1 to triangulate G and
        return Yes
      else return No
    else
      Find a feasible cycle γ with port vertices a and b
      if color(a) ≠ color(b) then introduce
```

```
            the edge (a, b) and complete (if possible) the triangulation
            of γ using the algorithm of Theorem 1
        else return No
    if V ≠ ∅ and no feasible cycle was found then return No
  endwhile
end
```

One consequence of the algorithm is that if G has a c-triangulation, then a perfect elimination scheme for the triangulation of G constructed by the algorithm will be given by the order in which the vertices are removed.

3.1. Proof of correctness. The main results of this section are the following two theorems.

THEOREM. *Let $G = (V, E)$ be a 2-connected reduced graph with a proper 3-coloring c, which can be c-triangulated. If G is not a cycle, then G contains a feasible cycle γ.*

THEOREM. *Let G be a 2-connected reduced graph with a proper 3-coloring c and let γ be a* feasible cycle *with port vertices a and b. Then G can be c-triangulated if and only if $G \cup (a, b)$ can be c-triangulated.*

The correctness of the algorithm follows from these two theorems, along with the observation that every simple cycle with all three colors represented can be c-triangulated.

THEOREM 1. *Let G be a simple k-cycle with a proper coloring c. Then G can be c-triangulated if and only if at least three colors are represented. In this case, we can c-triangulate G in $O(k)$ time.*

Proof. (by induction on k). It is obvious that at least three colors must be present in any cycle for the graph to be c-triangulated. We now show that every k-cycle with all three colors represented can be c-triangulated in $O(k)$ time. Beginning at any vertex v in γ, go around the cycle counting the number of vertices of each color. If there is only one vertex x of a given color in γ, insert all edges (x, y), for each vertex y in γ. Otherwise, beginning at v, again go around the cycle clockwise until a vertex w is found such that w's neighbors are not the same color and w is not the only vertex of its color. Delete w and connect its neighbors, then move counterclockwise to w's neighbor and begin again, each time updating the number of vertices of each color remaining. It is not hard to see that in one additional pass through the cycle we will have produced a c-triangulation. Thus G is c-triangulatable. □

LEMMA 1. *Let v be a simplicial vertex in V. Then G can be c-triangulated if and only if $G - \{v\}$ can also be.*

Proof. If G' is a c-triangulation of G, then the induced subgraph of G' on $V - \{v\}$ is a c-triangulation of $G - \{v\}$.

Conversely, suppose that G' is a c-triangulation of $G - \{v\}$. G' has a perfect elimination scheme. Let G'' be obtained by adding back v to G' and connecting it to all its neighbors in G. Then a perfect elimination scheme for G'' is obtained by following up v with the perfect elimination order for G'. Hence G'' is chordal and is a c-triangulation of G. □

Now let G be a reduced graph that can be c-triangulated, and assume that G is not a simple cycle. We now prove that a *feasible cycle* exists.

We now make some observations about the relationship between perfect elimination schemes and c-triangulatable graphs. We show that the effect of forming cliques out of neighbor sets of vertices is to contract cycles, until they are reduced to a single edge. Then, given a perfect elimination scheme $\mathcal{P}$ for a 3-colored chordal graph G',

there will be a unique first cycle γ in G which is contracted to an edge, (a, b). We call that first cycle the "first eliminated cycle in P, with final vertices a and b." In any cycle γ, the vertices that have neighbors outside of the cycle will be called the ports of γ.

OBSERVATION 1. *Let G be a graph that can be c-triangulated, and with a proper 3-coloring c, and let $\mathcal{P}$ be an elimination scheme for G. If γ is a chordless cycle in G, then in the successive graphs G_i that occur in the elimination sequence of G (see §2), γ is contracted one edge at a time, until only three vertices remain. These final three vertices of γ in the perfect elimination scheme will constitute a 3-clique in G'.*

Proof. Consider a chordless cycle γ in G. Let v_i be the vertex in γ to appear first in the ordering $\mathcal{P}$. When we remove v_i and connect its neighbors in G_{i-1}, these neighbors will perforce be in γ, and hence the result will be a smaller cycle, containing all of the vertices in γ except v_i. This will repeatedly occur, until γ is a 3-cycle. Thus the three vertices of γ that appear last in the perfect elimination scheme $\mathcal{P}$ must form a 3-clique in G'. □

OBSERVATION 2. *If G is a 3-colored graph that can be c-triangulated, then there exists a chordal colored supergraph G' of G such that for all vertices a,b, and c in G, if $\{a, b, c\}$ is a clique in G', then there is a simple cycle in G containing all three vertices.*

Proof. We prove this observation for reduced 2-connected 3-colored graphs and leave the details for the general case to the reader.

Our proof proceeds by induction on $n = |V|$. The observation is trivially true for $n = 1, 2$, or 3. Let G be a reduced 2-connected 3-colored graph that can be c-triangulated, and P an elimination scheme for G, and assume that the observation is true for all graphs with fewer than n vertices. Let v_1 be the first vertex in P, with neighbors a and b. We consider the chordal graph G' derived from G and P.

Let $\{x, y, z\}$ be a 3-clique in G'. If $v_1 \in \{x, y, z\}$, then, clearly, $\{x, y, z\} = \{v_1, a, b\}$. Furthermore, since G is 2-connected, there is a path from a to b not including v_1, and thus a, b, and v_1 are all in some simple cycle γ in G. On the other hand, if v_1 is not in $\{x, y, z\}$, then $\{x, y, z\}$ is a 3-clique in the graph G'' derived from $G^* = G - \{v_1\} \cup \{(a, b)\}$ and P (restricted to $V - \{v_1\}$). Thus, inductively, $\{x, y, z\}$ is in a simple cycle γ in G^*. If γ does not include the edge (a, b), or if (a, b) is an edge in G, then γ is a cycle in G as well. On the other hand, if γ includes the edge (a, b), and this edge is not in G, we can enlarge γ by removing the edge (a, b), adding v_1, and attaching v_1 to a and b. Thus, in each case, $\{x, y, z\}$ is in a simple cycle in G. □

In the next lemma, G is a 2-connected reduced graph with a proper vertex coloring $c : V \to \{1, 2, 3\}$. Assume that G can be c-triangulated and that $\mathcal{P} = v_1 < v_2 < \cdots < v_n$ is an elimination scheme for G. Using terminology defined in §2, we will let G' denote the c-triangulation of G derived from $\mathcal{P}$, $\Gamma_i(v)$ the neighbor set of vertex v in graph G_i, and $\deg_i(v)$ the cardinality of $\Gamma_i(v)$.

LEMMA 2. *There exists a unique k-cycle γ that is the first eliminated cycle for the elimination scheme $\mathcal{P}$, and for some elimination scheme $\mathcal{P}'$, the first $k-2$ vertices in $\mathcal{P}'$ are vertices in γ of degree 2.*

Proof. Since G is a 3-colored graph, $\deg_{i-1}(v_i)$ is always at most 2. If two distinct cycles γ_1 and γ_2 were both reduced to an edge by the deletion of a vertex v_i, then, in G_{i-1}, γ_1 and γ_2 would have each been reduced to a 3-cycle, each containing v. This would imply that $\deg_{i-1}(v_i) > 2$, contradicting the above. Thus the cycles are each reduced to an edge at a distinct time, and therefore there is a unique first eliminated

cycle γ in G for the elimination scheme $\mathcal{P}$.

Let the number of vertices of γ be k. We now show that we can reorder the vertices to derive a (possibly) different elimination scheme $\mathcal{P}'$, in which the first $k-2$ vertices are vertices in γ of degree 2. This proof is done by induction on $n = |V|$.

The basis case, where $n = 1$, is obviously true. Now assume that for any graph with $n-1$ or fewer vertices, the lemma holds.

Let us suppose that the elimination scheme $\mathcal{P}$ is the ordering $v_1 < v_2 < \cdots < v_n$, and that the first vertex of γ to appear in the elimination scheme is v_i. We will show that we can move v_i to the first position in $\mathcal{P}$ so that γ remains the first eliminated cycle, and the resultant ordering is still a perfect elimination scheme for G'. This will be possible, provided that v_i's neighbors in G' follow v_i in the ordering $\mathcal{P}$.

Within the cycle γ (in the graph G), v_i has two neighbors a and b. Since v_i precedes a and b, and $deg_{i-1}(v_i) \leq 2$, clearly, $\Gamma_{i-1}(v_i) = \{a, b\}$.

Now suppose that v_i's neighbors in G' do not all follow v_i in the ordering $\mathcal{P}$. Let v_r be the last neighbor of v_i in G' to occur before v_i in the ordering. Now G is 2-connected, so v_r is in some cycle γ', and hence $\deg_{r-1}(v_r) = 2$, as γ' is not contracted to an edge in G_{r-1}. Thus $\Gamma_{r-1}(v_r) = \{v_i, v_l\}$, for some $l > r$. When we remove v_r and connect its neighbors, the result is that $v_l \in \Gamma_r(v_i)$. Unless v_l is a or b, this would contradict our choice of v_r. Hence we can assume (without loss of generality) that $v_l = a$.

Now $\Gamma_{r-1}(v_r) = \{a, v_i\}$, and thus $\{a, v_i, v_r\}$ constitutes a 3-clique in G_{r-1}. Since G_{r-1} is a subgraph of G', by Observation 2, we see that a, v_i, and v_r were contained in some cycle γ'' in G. This cycle was eliminated before γ was eliminated, however, contradicting our hypothesis.

We have therefore proved that $\Gamma_{G'}(v_i) = \{a, b\}$ and we can move v_i to the front of the ordering. We have also shown that v_i has degree 2.

Now consider the graph that results when we remove v_i from G and connect its two neighbors. The cycle γ is replaced by the cycle δ, containing $k-1$ vertices. This graph can also be c-triangulated, since it has an elimination scheme (simply, take the elimination scheme $\mathcal{P}$, remove v_i from it, and use the resultant ordering $\mathcal{P}'$: $v_1 < v_2 < \cdots < v_{i-1} < v_{i+1} < \cdots < v_n$). In this ordering, the first cycle to be reduced to an edge is δ. By the inductive hypothesis, we can move all but two of the vertices of δ to the front of the ordering P'. The lemma follows. □

As a corollary to this lemma, there must be a feasible cycle! Thus, our first main result is now proved as a corollary.

COROLLARY 1. *Let G be a reduced 2-connected graph with a proper 3-coloring c, which is not a simple cycle. If G can be c-triangulated, then G contains a feasible cycle, γ.*

Proof. Recall that a k-cycle γ is a *feasible cycle* if it consists of $k-2$ vertices of degree 2 and two port vertices a and b. Now let G be a 2-connected reduced graph with a proper 3-coloring c, which is not a simple cycle and can be c-triangulated. By Lemma 2, there is an elimination scheme for G in which (for some r) the first r vertices constitute all but two of the vertices in the first eliminated cycle, γ, and these r vertices all have degree 2 in G. Let us refer to the final two vertices in the cycle by x and y. Since G is 2-connected and is not a simple cycle, at least two of the vertices in γ must be port vertices. These port vertices must be x and y, since the remaining vertices are all of degree 2. □

The second main result of the section now follows.

THEOREM 2. *Let G be a 2-connected reduced graph with a proper 3-coloring c.*

Let γ be a feasible cycle *with port vertices x and y. Then G is c-triangulatable if and only if $G \cup (x, y)$ is c-triangulatable.*

Proof. One direction is obvious. Now assume that G can be c-triangulated, that G' is one such c-triangulation, and that (x, y) are the port vertices in a feasible cycle γ. For our discussion, we write γ as the union of two paths P_1 and P_2. It is clear that since G is 2-connected and the other vertices of γ are of degree 2, x and y are connected by a path P_3 that does not intersect γ except at x and y, and which has length > 1. Thus x and y are in at least two distinct simple cycles: γ, comprised of the paths P_1 and P_2, and the cycle γ', comprised of P_1 and P_3. At least one of these cycles must be contracted to an edge before the other in the elimination sequence for G. It is easy to see that whichever cycle is eliminated first will leave x and y as the last two vertices in the cycle, and hence force x and y to be adjacent in G'. Since G' was arbitrary, this also shows that these edges are contained in the intersection over all chordal supergraphs of G. □

The last comment in our proof of Theorem 2 yields the following theorem.

THEOREM 3. *The algorithm for 3-colored graphs produces a minimum c-triangulation if the graph can be c-triangulated.*

Proof. Clearly, if the triangulation produced is minimum for each biconnected components, then it will be minimum for the entire graph, since the algorithm never introduces edges between vertices in different biconnected components. Therefore we need only consider how the algorithm treats each biconnected component. Our proof proceeds by induction on the number of vertices in G.

We will assume the graph G has some c-triangulation and that it is biconnected. If G is a k-cycle, then it has three colors represented, and the algorithm produces a c-triangulation of the k-cycle. It can easily be seen that the number of additional edges required of any c-triangulation of a k-cycle is exactly $k - 3$, and, in fact, if a k cycle has more than $k - 3$ edges added, it cannot be properly colored. Hence the algorithm produces a minimum triangulation on k-cycles.

Now consider how the algorithm responds to a biconnected graph that is not a k-cycle. It finds either a feasible cycle (in which case, it introduces an edge that must exist in *any* c-triangulation of G, as shown in the proof of Theorem 5), or it finds a simplicial vertex and removes it. Since the graph has a c-triangulation, eventually the algorithm will find a simplicial vertex and remove it. Until that point, the only edges it will have introduced will be edges that exist in all c-triangulations. After it removes a vertex, the inductive hypothesis indicates that the c-triangulation of the resultant graph is minimum, and thus the c-triangulation of G is minimum. □

This is interesting because it is known that finding a minimum triangulation of an uncolored graph is NP-complete [1].

3.2. The implementation and its running time. We now describe the details of the algorithm for determining if a 3-colored graph can be c-triangulated. We assume that our input graph is a properly colored graph G with n vertices and m edges, and with all three colors appearing. Furthermore, as we noted earlier, we can make the assumption that $m \leq 2n - 2$.

1. Using bucket sort, we sort all vertices by degree. This takes $O(n)$ time.
2. We remove all vertices of degree 0 and 1 and all simplicial vertices of degree 2. Each removal of a vertex causes an update in the degrees of its neighbors (if any). Thus new simplicial vertices may be created, which we also remove. This step takes $O(n)$ time. Since there is no fear of confusion, we will denote the new graph obtained after this step also by G.

3. We find the biconnected components of G using depth-first search. This takes time $O(m+n)$. We handle the biconnected components that are cycles with the algorithm in Theorem 1.
4. We maintain an edge-labelled multigraph, which we term the *path graph*. In each biconnected component of G, we must know which vertices of degree > 2 are connected to each other by paths consisting entirely of degree 2 vertices. To obtain this graph from G, we begin with G, and with the empty label on each edge. We repeatedly do the following: We find a degree 2 vertex v. If u and w are the neighbors of v, we remove v and add the edge between u and w. The label on the new edge will be $(\text{label}(e_1), v, \text{label}(e_2))$, where e_1 is the edge between u and v, and e_2 the edge between v and w. The resulting multigraph will be called $P(G)$. Our construction shows that $P(G)$ can be found in $O(m)$ time.

When does a cycle γ constitute a feasible cycle? It must contain exactly two port vertices u and v, with the remaining vertices having degree 2. Since there must be at least two paths consisting entirely of degree 2 vertices between u and v, the vertices u and v must be connected by at least two edges in the $P(G)$. We show that finding all feasible cycles and updating the information in $P(G)$ can be done in time $O(m+n)$.

We preprocess $P(G)$ for one more piece of information. We must know which pairs of adjacent vertices in $P(G)$ are connected to each other by at least two edges. We do this by bucket sorting each pair of adjacent vertices into two buckets: one for pairs of vertices that have exactly one edge between them, and the other for pairs of vertices with two or more edges between them.

Finally, the remaining data structure we maintain is a queue of *connectable pairs of vertices* Q. By a possible connectable pair, we mean a pair of vertices x and y that are port vertices in a feasible cycle. The labels on the edges in $P(G)$ between x and y indicate the paths of degree 2 vertices connecting x and y.

5. We pull out a possible connectable pair from the top of the queue. Let this pair be u, v. If u and v have the same color, we know the graph cannot be c-triangulated, and we exit. Otherwise, we check if they are adjacent in G. If they are not adjacent in G, we add the edge (u, v) to E, and we complete the triangulation of the *feasible cycles* containing u and v (as indicated by the labels on the edges in $P(G)$), using the algorithm of Theorem 1.
   We do the following update steps:
   - In $P(G)$ we remove all but one edge between u and v. If either u or v or both become degree 2 vertices in $P(G)$, we short-circuit them as we did before in arriving at $P(G)$. This might create at most one new possible connectable pair;
   - It is also possible that at most one pair of vertices that were not adjacent to each other in $P(G)$ become adjacent. In this case, we put this pair in the bucket of vertices connected by exactly one edge.

   Note that all of these steps can be performed in constant time and that we enqueue at most one pair, and that only in the case where we remove a vertex from $P(G)$. Thus, in the whole process, we enqueue at most n additional pairs. $P(G)$ contains at most m pairs at the start, since the number of edges in $P(G)$ is no more than the number of edges in G.
6. We terminate either when Q becomes empty or $P(G)$ becomes empty. If Q becomes empty, we say that G is not c-triangulatable. If $P(G)$ becomes empty, we say that G is c-triangulatable.

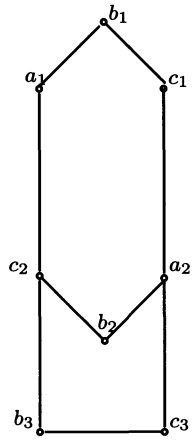


FIG. 1. *The graph.*

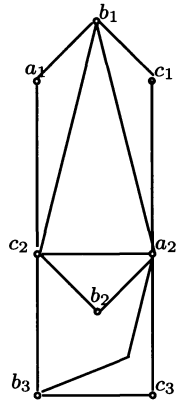


FIG. 2. *The graph triangulated.*

Note that we processed at most $m + n$ pairs and that each pair took $O(1)$ steps to process. Thus the overall running time is $O(m+n)$. However, since m is bounded by $2n - 2$, this is, in fact, $O(n)$.

3.3. A counterexample to a conjecture. In an attempt to simplify the algorithm, we might consider the following conjecture: let v be a vertex of degree 2, with neighbors x and y, such that the subgraph of G induced by the colors of x and y remains properly colored and acyclic by the addition of the edge (x, y). Then G is c-triangulatable if and only if $G \cup (x, y)$ is c-triangulatable. Figure 1 shows that this is not true: The color classes are $\{a_1, a_2\}$, $\{b_1, b_2, b_3\}$, and $\{c_1, c_2, c_3\}$. The vertex b_1 satisfies the conditions of the conjecture, and yet there is no way to triangulate G if we include the edge (a_1, c_1).

On the other hand, the graph can be triangulated, as Fig. 2 indicates.

4. Conclusions. As we have mentioned, the triangulating colored graphs problem is NP-complete, in general, and can be solved [13] in $O(n^{k+1})$ time, where n is the number of vertices and k the number of colors. Is this the best possible?

One especially interesting subcase of the corresponding molecular biology problem (the perfect phylogeny problem) is when the characters are restricted to having at most r states, where r is a fixed constant. We conjecture that the techniques in [12] may possibly be extended to that case, yielding an $O(r^{r-2}n^2k)$ algorithm, where n is the number of species and k is the number of characters.

REFERENCES

[1] S. Arnborg, D. Corneil, and A. Proskurowski, *Complexity of finding embeddings in a k-tree*, SIAM J. Algebraic Discrete Meth., 8 (1987), pp. 277–284.

[2] H. Bodlaender, M. Fellows, and T. Warnow, *Two strikes against the Perfect Phylogeny problem*, Proc. Internat. Colloq. on Automata, Languages, and Programming, 1992, to appear.

[3] P. Buneman, *A characterization of rigid circuit graphs*, Discrete Math., 9 (1974), pp. 205–212.

[4] W. H. E. Day, D. S. Johnson, and David Sankoff, *The computational complexity of inferring rooted phylogenies by parsimony*, Math. Biosci., 81 (1986), pp. 33–42.

[5] W. H. E. Day and D. Sankoff, *Computational complexity of inferring phylogenies by compatibility*, Syst. Zool., 35 (1986), pp. 224–229.

[6] ———, *Computational complexity of inferring phylogenies from chromosome inversion data*, J. Theoret. Biol., 124 (1987), pp. 213–218.

[7] G. F. Estabrook and F. R. McMorris, *When are two qualitative taxonomic characters compatible?*, J. Math. Biol., 4 (1977), pp. 195–200.

[8] L. R. Foulds and R. L. Graham, *The Steiner problem in phylogeny is* NP-*complete*, Adv. in Appl. Math., 3 (1982), pp. 43–49.

[9] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[10] D. Gusfield, *The Steiner Tree problem in phylogeny*, Tech. Report 332, Department of Computer Science, Yale University, New Haven, CT, September 1984.

[11] ———, *Efficient algorithms for inferring evolutionary trees*, Networks, 21 (1991), pp. 19–28.

[12] S. Kannan and T. Warnow, *Inferring evolutionary history from* DNA *sequences*, in Proc. Symposium on the Foundations of Computer Science, St. Louis, MO, October 1990.

[13] C. F. McMorris, T. Warnow, and T. Wimer, *Triangulating Vertex Colored Graphs*, manuscript.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 259–272, May 1992
© 1992 Society for Industrial and Applied Mathematics
008

THREE-STAGE GENERALIZED CONNECTORS*

RICHARD M. KARP†

Abstract. An acyclic directed network with n sources and m sinks is called a generalized connector if, for any request pattern in which each sink asks to be connected to some source, the required configuration of noninterfering connecting paths can be set up. This paper presents new families of two- and three-stage connection networks and gives a method of establishing that particular networks in these families are generalized connectors. This method is based on the Erdös probabilistic method and consists of two steps:

1. First it is proven that a given network is a generalized connector provided that certain events in a probability space Ω are of sufficiently low probability.
2. Then it is shown that the probabilities of these events are indeed sufficiently small.

For the family of designs presented in Section 5, the second step is accomplished by explicitly calculating the probabilities of the events in question, using dynamic programming; these calculations provide rigorous proofs that the designs are correct. However, for the more economical designs of Section 6, the probabilities of the events in question are not calculated exactly, but instead are estimated by drawing pseudo-random samples from the probability space Ω. Thus, we have only a "statistical proof" (albeit a very convincing one) of the correctness of these designs.

For certain choices of n and m the most efficient three-stage generalized connectors currently known are obtained.

Key words. connection network, generalized connector, probabilistic method

AMS(MOS) subject classifications. 05C, 68R, 94C

1. Introduction.

This paper is concerned with the design of connection networks capable of providing noninterfering connecting paths between specified pairs of input and output ports. We model connection networks in terms of directed graphs. Let G be an acyclic digraph with vertex set V. Let S (the sources) be the set of vertices of in-degree zero and let T (the sinks) be the set of vertices of out-degree zero. A *request pattern* is a partial function $F : T \rightarrow S$; $F(t)$ is the source requested by sink t. A set Ψ of (directed) source-sink paths *satisfies* request pattern F if (i) for every sink t such that $F(t)$ is defined, Ψ contains a path from $F(t)$ to t, and (ii) two paths in Ψ have a vertex in common if and only if they have a source in common.

The *depth* of G is the maximum number of edges in a path of G. Let G have n sources, m sinks, and depth k. Then G is said to be a $(n \times m)$-*generalized connector of depth* k if, for every request F, there is a set of paths Ψ satisfying F; G is said to be a $(n \times m)$-*connector of depth* k if, for every one-to-one request F, there is a set of paths Ψ satisfying F. Thus, in a generalized connector, each sink may select any source, and the required set of noninterfering source-sink paths can be set up; in a connector, the required paths can be set up provided that each source is selected by at most one sink.

Our definitions correspond to what are generally called *rearrangeable* connectors and generalized connectors. Also studied in the literature is the stronger property that the connector or generalized connector be *nonblocking*; this means that when the request pattern is altered by adding or deleting a request, the corresponding change in Ψ, the satisfying set of paths, is simply to delete or add a single path without

* Received by the editors February 5, 1990; accepted for publication (in revised form) April 30, 1991. This research was supported by DVA Limited, Alamo, California, and National Science Foundation grant CCR-8411954.

† Computer Science Division, University of California, Berkeley, California 94720.

modifying any of the other paths in Ψ. Except for §8, we will not be concerned with nonblocking networks.

This work was motivated by the efforts of DVA Ltd., a California-based research and development partnership, to develop connection networks for applications in the broadcast industry. In these applications, the inputs and outputs of the connection network are audio and video signals that are very high-powered in comparison with the weak signals usually occurring in digital computation. Consequently, the individual switches are expensive devices, and their cost is the dominant factor in the overall cost of the network. For this reason, we take the number of edges (where each edge corresponds to a switch) as our measure of cost. Another feature of the application is that power dissipation becomes a serious problem if the signals are passed through more than about three stages. For these reasons, we concentrate on networks of very low depth. Finally, the networks arising in our application have at most one hundred or two hundred inputs and outputs; accordingly, we concentrate on designs that are efficient in this restricted range, rather than asymptotically efficient designs.

This paper defines two new families of networks of depth three and gives a method of establishing that particular networks in these families are generalized connectors. This method is based on the Erdös probabilistic method, and consists of two steps:

1. First it is proven that a given network is a generalized connector provided that certain events in a probability space Ω are of sufficiently low probability.
2. Then it is shown that the probabilities of these events are indeed sufficiently small.

For the family of designs presented in §5, the second step is accomplished by explicitly calculating the probabilities of the events in question, using dynamic programming; these calculations provide rigorous proofs that the designs are correct. However, for the more economical designs of Section 6, the probabilities of the events in question are not calculated exactly, but instead are estimated by drawing pseudo-random samples from the probability space Ω. Thus, we have only a "statistical proof" (albeit a very convincing one) of the correctness of these designs.

For certain values of n and m, our networks have the fewest edges of any known $n \times m$ generalized connectors of depth three. Section 8 has a somewhat different theme than the rest of the paper. Its main result is to exhibit an infinite family of "almost nonblocking" generalized connectors of depth two.

2. Past work. In this section we give a survey of relevant past work on rearrangeable connectors and generalized connectors of small depth. We restrict ourselves to explicitly constructed families of connection networks, rather than families proven to exist by nonconstructive methods. Also, we consider only those cases in which the design is accompanied by a polynomial-time routing algorithm, i.e., a polynomial-time algorithm that, given a request pattern F, constructs a set of paths satisfying F. Finally, since our interest is in networks with moderate numbers of sources and sinks, we exclude certain constructions that are good asymptotically but useful only when the number of sources and sinks is huge.

The above exclusions eliminate most of the literature on connectors and generalized connectors. All the remaining designs involve the interconnection of crossbar switches. An $(n \times m)$-*crossbar* is an $(n \times m)$-generalized connector of depth one, in which there is an edge from each source to each sink for a total of nm edges. Because of their regular structure, crossbars are particularly convenient to build. Among networks with a given number of switches, those built up from crossbars are preferred because of their regular structure.

A construction attributed in [2] to Slepian, Duguid, and Le Corre yields a family of depth-three $(n \times n)$-connectors composed of crossbars. Let n be of the form de. Then the network has three stages. The first and third stages each contain e $(d \times d)$-crossbars. The second stage contains d $(e \times e)$-crossbars. For i and k ranging from 1 to e, and j ranging from 1 to d, let $A(i)$ denote the ith first-stage crossbar; $B(j)$, the jth second-stage crossbar; and $C(k)$, the kth third-stage crossbar. The network is obtained by identifying the jth sink of $A(i)$ with the ith source of $B(j)$, and the kth sink of $B(j)$ with the jth source of $C(k)$, for all i, j, and k. In physical terms, identifying two terminals simply means hard-wiring them together. The network contains $(2d^2e + e^2d)$ switches. When $e = 2d$, the number of switches in the network is $2\sqrt{2}\, n^{\frac{3}{2}} + O(n)$.

The construction of source-sink paths in the Slepian–Duguid–Le Corre network to satisfy a given request pattern is based on the following theorem. Let G be a bipartite multigraph in which each vertex has degree at most d; then each edge of G can be assigned a "color" from the set $1, 2, \cdots, d$ in such a way that no two edges incident with a common vertex receive the same color. To process a given request pattern, let the bipartite graph have a vertex $a(i)$ for each first-stage crossbar $A(i)$, a vertex $c(k)$ for each third-stage crossbar $C(k)$, and an edge from $a(i)$ to $c(k)$ for each sink in $C(k)$ that requests a source in $A(i)$. If, in the coloring of the edges of this bipartite graph, an edge from $a(i)$ to $c(k)$ receives the color j, then the corresponding request is routed from $A(i)$ to $C(k)$ through $B(j)$; specifically, the path includes the ith source and kth sink of $B(j)$. The required coloring can be obtained in time $O(n \log n)$ using the edge coloring algorithm of Cole and Hopcroft [3].

The paper [4] exhibits a family of generalized connectors of depth four. The construction is based on the concept of a generalizer. An $(n \times m)$-*generalizer* is an acyclic digraph with n sources and m sinks, such that if each source u is assigned a nonnegative integer $a(u)$, such that $\sum_u a(u) \leq m$, then it is possible to simultaneously construct for all sources u, a set of paths from u to $a(u)$ distinct sinks, such that no two paths have a vertex in common unless they emanate from the same source. Clearly, we can build an $(n \times m)$-generalized connector from an $(n \times m)$-generalizer A and an $(m \times m)$-connector B simply by identifying the sinks of A with the sources of B, according to a one-to-one correspondence. In electrical terms, this amounts to hard-wiring each sink of A to a source of B. The paper [4] gives the following inductive construction of a depth-k $(n \times m)$-generalizer. For $k = 1$, the generalizer is simply an $(n \times m)$-crossbar. For the inductive construction, assume that $n = d_1 d_2$ and $m = d_1 d_3$, where d_1, d_2, and d_3 are positive integers. The first stage of the depth-k generalizer has n sources $a_0, a_1, \cdots, a_{n-1}$ and n output nodes $b_0, b_1, \cdots, b_{n-1}$. From each source a_i there is an edge to each of the output nodes $b_{i \bmod n}, b_{(i+1) \bmod n}, \cdots, b_{(i+d_1-1) \bmod n}$. The remaining $k - 1$ stages consist of d_1 vertex-disjoint depth-$(k-1)$ $(d_2 \times d_3)$ generalizers. There is one of these generalizers for each residue $a \bmod d_1$; the generalizer corresponding to residue a has as its sources the set of nodes b_i such that $i \equiv a \bmod d_1$. It is an interesting combinatorial exercise to show that this inductive construction yields a depth-k $(n \times m)$-generalizer.

Let $n = m = d^2$. We can obtain a depth-five $(n \times n)$-generalized connector by composing together a depth-two $(n \times n)$-generalizer, constructed as above with $d_1 = d_2 = \sqrt{n}$, with a depth-three $n \times n$-connector, produced by the Slepian–Duguid–Le Corre construction with $d = e = \sqrt{n}$. This construction can be improved upon by noting that the second stage of the generalizer and the first stage of the connector can be compressed together into a single stage. For, since the sinks of each $(\sqrt{n} \times \sqrt{n})$-

crossbar at the second stage of the generalizer are identified one-to-one with the sources of some $(\sqrt{n} \times \sqrt{n})$ first-stage crossbar of the connector, giving two crossbars "in series," each such pair of crossbars can be replaced by a single $(\sqrt{n} \times \sqrt{n})$-crossbar. This simplification results in a depth-four $(n \times n)$-generalized connector with $4n^{3/2}$ edges, where n is a perfect square.

We turn now to the construction of depth-three $(n \times n)$-generalized connectors. The paper [6], improving upon earlier constructions [7], [8], exhibits families of such connectors with $O(n^{3/2}(\log n)^{1/2})$ edges. [5] improving on earlier bounds given in [9] and [4], shows by a nonconstructive probabilistic argument that there exists a family of depth-three $(n \times n)$-generalized connectors with $O(n^{4/3}\log^{2/3} n)$ edges; however, the construction is not explicit and, moreover, no polynomial-time or randomized polynomial-time routing algorithm is known for this family.

These results left open the question of the existence of an explicitly defined family of depth-three $(n \times n)$-generalized connectors having $O(n^{3/2})$ edges and a fast routing algorithm. This question was answered affirmatively around 1985 by Pippenger and Spencer. The Pippenger–Spencer [PS] construction will be presented in §4.

3. A general design strategy. In this section we describe a very general construction that can be specialized in various ways to yield generalized connectors. Let n, r, s, f, and g be positive integer parameters. Consider a three-stage network of the following form:

(i) The network has n sources and m sinks, where $m = fg$;
(ii) The first stage is a network (not necessarily of depth one) with n sources and rs sinks;
(iii) The second stage consists of s $(r \times f)$-crossbars;
(iv) The third stage consists of f $(s \times g)$-crossbars;
(v) The rs first-stage sinks are in one-to-one correspondence with the rs second-stage sources, and each corresponding source-sink pair is identified as a single vertex (electrically, each such source-sink pair is hard-wired together);
(vi) For $j = 1, 2, \cdots, s$ and $k = 1, 2, \cdots, f$, the kth sink of the jth second-stage crossbar is identified with (hard-wired to) the jth source of the kth third-stage crossbar.

The above description does not specify the structure of the first stage. We derive properties of the first-stage structure sufficient to ensure that the overall network is a generalized connector. Let S be the set of first-stage sources and let D be the set of first-stage sinks. Then $|S| = n$ and $|D| = rs$. A request pattern for the first-stage network is a function from D into S. Let us call such a request pattern K *realizable* if it is possible to set up paths through the first stage which are vertex-disjoint except if they originate from the same source, and which satisfy all the requests specified by K.

For each realizable K, and each first-stage source x, let A_K^x be the subset of $1, 2, \cdots, s$ defined as follows: $j \epsilon A_K^x$ if there exists a first-stage sink y such that y is hard-wired to a source of the jth second-stage crossbar and $K(y) = x$. In other words, $j \epsilon A_k^x$ if K connects first-stage source x to some source of the jth second-stage crossbar.

Let F be a request pattern, mapping each third-stage sink to a first-stage source. Let K be a realizable function. For any third-stage crossbar C, let $F(C)$ denote $\{F(z) |\ z$ is a sink of $C\}$. Let us say that a configuration of the network is *consistent with* K if, in that configuration, the setting of the first-stage switches satisfies the

first-stage request pattern K. Recall that a family of sets $S_1, S_2, \cdots, S_t$ is said to have a *system of distinct representatives* (abbreviated SDR) if there exists a sequence $x_1, x_2, \cdots, x_t$ of distinct elements such that, for $i = 1, 2, \cdots, t$, $x_i \epsilon S_i$. Element x_i is called the *representative* of set S_i. Using a reduction to bipartite matching, there is an algorithm to find an SDR for a given family of sets, or else determine that there is none, in time $O(e\sqrt{v})$, where e is the sum of the cardinalities of the sets, and v is the number of sets plus the cardinality of the union of the sets.

THEOREM 3.1. *The network has a configuration that satisfies F and is consistent with K if and only if, for each third-stage crossbar C, the family of sets $\{A_K^x |\ x \epsilon F(C)\}$ has an* SDR.

Proof. Suppose that the required SDR exists. Let w be a sink of third-stage crossbar $C(k)$. Let x be an element of $F(C(k))$ and let j be the representative of A_K^x. Since $j \epsilon A_K^x$, there is a first-stage sink y such that $K(y) = x$ and y is identified with a source of second-stage crossbar $B(j)$. Then the required configuration will include the path from x to y that is conducting in the first-stage configuration satisfying K, the edge $[y, a]$ where a is both the kth sink of second-stage crossbar $B(j)$ and the jth source of $C(k)$, and edges from a to each sink w of $C(k)$ such that $F(w) = x$. If w' is another sink, lying on third-stage crossbar $C(k')$, and $F(w') = x'$, then the path from x' to w' is defined similarly. Let it run through the first stage from x' to y', and then along the edges $[y', a']$ and $[a', w']$, where the definitions of y' and a' are analogous to those of a and y. We must show that if $F(w) \neq F(w')$, then these two paths are vertex-disjoint. Since $F(w) \neq F(w')$ the first-stage paths from $F(w)$ to y and $F(w')$ to y' are vertex-disjoint; this follows from the assumption that the first-stage configuration satisfies K. By definition, a is the kth sink of $B(j)$, and a' is the k'th sink of $B(j')$; thus a can be equal to a' only if $j = j'$ and $k = k'$; but this would imply that j is the representative of both A_K^x and $A_K^{x'}$, contradicting the definition of an SDR. Thus the paths are vertex-disjoint, and the configuration we have defined does satisfy F.

Now suppose that for some third-stage crossbar C, the required SDR does not exist. By Hall's theorem, there exists a "deficient" set of sources, i.e., a set $S \subseteq F(C)$ such that $|\bigcup_{x \epsilon S} A_K^x| < |S|$. This means that all the sources in S must be connected to sinks in C through fewer than $|S|$ second-stage crossbars; but this is impossible since each second-stage crossbar is connected to only one source of C. □

Let us define the predicate $T(F, K, C)$ as follows: $T(F, K, C)$ is true if $\{A_K^x | x \epsilon F(C)\}$ has an SDR. Then, to prove that a network of the type we have just defined is a generalized connector, we must prove that

(1) For all F, there exists K for all C $T(F, K, C)$.

To prove such a statement, we use the Erdös probabilistic method. Let Ω be a probability space whose points are realizable functions. Then it will suffice to prove that

(2) For all F Pr [for all C $T(F, K, C,)] > 0$, or equivalently,

(3) For all F Pr [there exists C $\neg T(F, K, C)] < 1$,

where the probability is with respect to a K drawn from Ω.

Since C can be chosen in only f ways, and since the probability of a union of events is less than or equal to the sum of the probabilities of the individual events, it suffices to prove that

(3) For all F for all C $(Pr\ [\neg T(F, K, C)] < 1/f)$.

Our approach in this paper will always be to prove (3). The remainder of the paper will consist of the application of this approach to various particular designs.

4. The Pippenger–Spencer design. In unpublished work done around 1985, Pippenger and Spencer devised a family of three-stage $(n \times n)$-generalized connectors with $9n^{3/2}$ switches. Their design, and the methodology used to prove its correctness, provided the motivating example for the general approach developed in §3. The Pippenger–Spencer construction resembles the three-stage connector of Slepian, Duguid, and Le Corre. Let $n = d^2$, where d is a positive integer. The first stage of the Pippenger–Spencer network contains d $(d \times 3d)$-crossbar switches; the second stage contains $3d$ $(d \times d)$-crossbar switches; and the third stage contains d $(3d \times d)$-crossbar switches. Let $A(i)$ be the ith first-stage crossbar switch; $B(j)$, the jth second-stage crossbar switch; and $C(k)$, the kth third-stage crossbar switch. The network is obtained by connecting these crossbars together as follows: the jth sink of $A(i)$ is identified with the ith source of $B(j)$, and the jth source of $C(k)$ is identified with the kth sink of $B(j)$.

In terms of the general construction of §3, we have $f = g = r = d$, $s = 3d$, and $n = m = d^2$. The first stage has the set of sources S and the set of sinks D, where $|S| = d^2$ and $|D| = 3d^2$. A function $K: D \to S$ is realizable if and only if, for each $x \epsilon D$, x and $K(x)$ lie on the same first-stage crossbar. The probability space Ω consists of those realizable functions with the additional property that each element of S is the image of exactly three elements of D. All functions in Ω are assigned the same probability. Thus, to draw a K from the probability space Ω, we partition the $3d$ sinks of each first-stage crossbar into d three-element sets, and randomly establish a one-to-one correspondence between these d three-element sets and the d sources of the same crossbar. Then $K(x) = y$ if and only if sink x is in the three-element set associated with y.

As we discussed in the last section, we can prove that the $(n \times n)$-Pippenger–Spencer network is a generalized connector by showing that, for each request pattern F and each third-stage crossbar C, the probability that $\{A_K^x |\ x \epsilon F(C)\}$ fails to have an SDR is less than $1/d$ when K is drawn from Ω. From the definition of Ω and the fact that the sinks of any first-stage crossbar are connected one-to-one to the second-stage crossbars, it follows that, for each $x \epsilon F(C)$ the set A_K^x is a three-element subset of $1, 2, \cdots, 3d$, that $A_K(x_1)$ and $A_K(x_2)$ are disjoint whenever x_1 and x_2 are sources of the same first-stage crossbar, and that all specifications of $\{A_K^x |\ x \epsilon F(C)\}$ satisfying these conditions are equally likely. Using this description of the distribution of $\{A_K^x |\ x \epsilon F(C)\}$, Pippenger and Spencer perform a probabilistic calculation based on Hall's theorem about bipartite matching to show that the probability that $\{A_K^x |\ x \epsilon F(C)\}$ fails to have an SDR is $O(d^{-4})$. It follows that when n (and hence d) is sufficiently large, the Pippenger–Spencer networks are generalized connectors. Thus Pippenger and Spencer have provided an explicit family of three-stage $(n \times n)$-generalized connectors with $9n^{3/2}$ switches.

There is a simple randomized routing algorithm for the three-stage generalized connector of Pippenger and Spencer: simply draw a realizable function K from Ω and try to construct the required SDRs. If all the SDRs exist, then use the recipe given in the proof of Theorem 3.1 to derive the paths required to satisfy F. When d is sufficiently large, the probability is high that, for a random K, the required SDRs will exist.

5. A Variant of the Pippenger–Spencer design. In this section we consider a variant of the Pippenger–Spencer design in which each first-stage crossbar has twice as many sinks as sources, rather than three times as many. Our networks will have n sources and m sinks. Assume that $n = dr$ and $m = fg$, where d, e, f, and g are positive

integers. Our network will have r $(d \times 2d)$ first-stage crossbars $A(1), A(2), \cdots, A(n)$, $2d$ $(r \times f)$ second-stage crossbars $B(1), B(2), \cdots, B(2d)$ and f $(2d \times g)$ third-stage crossbars $C(1), C(2), \cdots, C(f)$. As in the Slepian–Duguid–Le Corre and Pippenger–Spencer designs, the jth sink of $A(i)$ is identified with the ith source of $B(j)$, and the kth sink of $B(j)$ is identified with the jth source of $C(k)$. In terms of the general construction of §3, we have $n = dr$ and $s = 2d$.

We prove that, for certain choices of d, r, f, and g, this construction yields generalized connectors. Up to a point, our approach to the proof will parallel that of Pippenger and Spencer, but, at the cost of some redundancy, we will be completely explicit, rather than asking the reader to make the necessary changes in the set-up for the Pippenger–Spencer proof.

The first stage has the set of sources S and the set of sinks D, where $|S| = dr$ and $|D| = 2dr$. A function $K : D \to S$ is realizable if and only if, for each $x \epsilon D$, x and $K(x)$ lie on the same first-stage crossbar. The probability space Ω consists of those realizable functions with the additional property that each element of S is the image under F of exactly two elements of D. All functions in Ω are assigned the same probability. Thus, to draw a realizable function K from the probability space Ω, we randomly partition the $2d$ sinks of each first-stage crossbar into d two-element sets, and randomly establish a one-to-one correspondence between these two-element sets and the sources of the same crossbar. Then $K(x) = y$ if and only if sink x is in the two-element set associated with y.

We can prove that our network is a generalized connector by showing that, for each request pattern F and each third-stage crossbar C, the probability that $\{A_K^x |\ x \epsilon F(C)\}$ fails to have an SDR is less than $1/f$, when K is drawn from Ω. From the definition of Ω it follows that, for each $x \epsilon F(C)$ the set A_K^x is a two-element subset of $\{1, 2, \cdots, 2d\}$, that $A_K^{x_1}$ and $A_K^{x_2}$ are disjoint if x_1 and x_2 are distinct sources on the same first-stage crossbar, and that all specifications of $\{A_K^x |\ x \epsilon F(C)\}$ satisfying these conditions are equally likely.

By exploiting the fact that each set A_K^x is of cardinality two, we can give an alternate description of the SDR problem associated with $\{A_K^x |\ x \epsilon F(C)\}$. Let $H_{K,F}(C)$ be a multigraph (i.e., multiple edges may occur) with vertex set $\{1, 2, ..., 2d\}$, having as its multiset of edges $\{A_K^x |\ x \epsilon F(C)\}$. Then the statement that $\{A_K^x |\ x \epsilon F(C)\}$ has an SDR is equivalent to the statement that it is possible to orient the edges of $H_{K,F}(C)$ so that no two edges point toward the same vertex; this, in turn, is possible if and only if each connected component of $H_{K,F}(C)$ is either a tree or a unicyclic graph (a connected graph is said to be unicyclic if it has exactly as many vertices as it has edges; equivalently, a connected graph is unicyclic if it contains exactly one simple cycle). We omit the simple proofs of these statements.

Call a graph *good* if each of its connected components is either a tree or a unicyclic graph, and *bad* otherwise. Then we must prove that, for all F and C, the probability that $H_{K,F}(C)$ is good, when K is drawn from Ω, is at least $1/f$.

Let us define $H_F(C)$ as the random variable over the probability space Ω that, at the point K, is equal to $H_{K,F}(C)$. Then we are interested in the probability that $H_F(C)$ is a bad graph. The distribution of $H_F(C)$ depends on both the request pattern F and the crossbar C. To account for this dependence, we introduce the concept of a k-signature. A *k-signature* $\underline{a} = (a_1, a_2, \cdots, a_h)$ is simply a partition of the integer k; i.e., the a_i are positive integers summing to k, and their order does not matter. Associated with each k-signature $\underline{a}$ is a probability space $S(\underline{a})$ of k-edge multigraphs on the vertex set $1, 2, \cdots, 2d$. The experiment required to draw a multigraph from

$S(\underline{a})$ is as follows: for each integer a_i, choose a_i vertex-disjoint edges at random from the complete graph on vertex set $\{1, 2, \cdots, 2d\}$; i.e., all sets of a_i vertex-disjoint edges are equally likely to be chosen. The edge set of the multigraph is then the union of the h vertex-disjoint sets that have been chosen.

LEMMA 5.1. *A sufficient condition for the three-stage network determined by the parameters d, r, f and g to be a generalized connector is that, for every g-signature $\underline{a}$, the probability that a multigraph drawn from $S(\underline{a})$ is bad is less than $1/f$.*

Proof. We need to show that, under the stated condition, the probability that $H_F(C)$ is bad is less than $1/f$. We may restrict attention to pairs F, C such that, in request pattern F, no two sinks of C request the same source. Partition the sinks of C into equivalence classes, so that sinks x_1 and x_2 are in the same equivalence class if $F(x_1)$ and $F(x_2)$ are on the same first-stage crossbar. Associate with the pair C, F the signature $\underline{a}$ whose parts are the cardinalities of the equivalence classes into which the sinks of C are divided. Then clearly the distribution of $H_F(C)$ is the same as the distribution of multigraphs drawn from $S(\underline{a})$. □

Lampe and the author of this paper have devised a dynamic programming algorithm that computes, for each k-signature $\underline{a}$, where k ranges from 1 to g, the probability that a multigraph drawn from $S(\underline{a})$ is bad. The algorithm works as follows. Let H be a multigraph on vertex set $\{1, 2, \cdots, 2d\}$ in which each component is either a tree or unicyclic. Define the *profile* of H as the expression $(p;\ q_1, q_2, \cdots, q_t)$, where p is the number of vertices in unicyclic components, t is the number of acyclic components, and $(q_1, q_2, \cdots, q_t)$ are the numbers of vertices in the acyclic components. The algorithm computes, for each k-signature $\underline{a}$ and profile π, the probability that a multigraph drawn from $S(\underline{a})$ has the profile π. The computation is recursive, and proceeds through increasing values of k. The probability that a multigraph drawn from $S(\underline{a})$ is good is then the sum, over all profiles π, of the probability that a graph drawn from $S(\underline{a})$ has profile π. We do not describe the details of the recursive computation.

Using this dynamic programming algorithm, we have proved that the following parameter choices all yield three-stage generalized connectors:

$n = 5r$, $m = 6f$, r arbitrary, $f \leq 20$,
Number of switches: $50r + 10rf + 60f$;

$n = 6r$, $m = 7f$, r arbitrary, $f \leq 18$,
Number of switches: $72r + 12rf + 84f$;

$n = 7r$, $m = 8f$, r arbitrary, $f \leq 18$,
Number of switches: $98r + 14rf + 112f$;

$n = 8r$, $m = 9f$, r arbitrary, $f \leq 17$,
Number of switches: $128r + 16rf + 144f$;

$n = 9r$, $m = 10f$, r arbitrary, $f \leq 16$,
Number of switches: $162r + 18rf + 180f$.

Particular parameter choices yield the following designs:

n	m	Number of switches
100	102	5420
120	117	6912
136	117	7584

The dynamic programming computations also revealed an interesting pattern. Let us say that signature $\underline{a}$ is a *refinement* of signature $\underline{a}'$ if $\underline{a}'$ can be obtained by

dividing the parts of $\underline{a}$ into disjoint groups and replacing each group by the sum of its parts. For example, the partition $(2,3,2,4,1)$ is a refinement of (5,6,1); the k-partition in which each part is 1 is a refinement of every k-partition. Our computational results suggest the following conjecture: if partition $\underline{a}$ is a refinement of partition $\underline{a}'$, then the probability that a graph drawn from $S(\underline{a})$ is bad is at least as great as the probability that a graph drawn from $S(\underline{a}')$ is bad. If this conjecture were known to be true then, to prove that the network with parameters d, r, f, and g is a generalized connector, it would suffice to consider the g-signature in which every part is 1. In other words, it would suffice to show that, when g edges are drawn with replacement from the uniform distribution over the edge set of the complete graph on vertex set $\{1, 2, \cdots, 2d\}$, the probability of getting a bad graph is less than $1/f$; let us say that the triple $\langle d, g, f\rangle$ is *good* if this statement is true. By adapting standard results from the theory of random graphs, we find that, for every α less than 1, and for all d sufficiently large, the triple $\langle d, \lfloor \alpha d \rfloor, 2d\rangle$ is good. Thus, if the conjecture is true, the parameter choice $n = 2d\lfloor \alpha d \rfloor$, $r = 2\lfloor \alpha d \rfloor$, $s = 2d$, $g = \lfloor \alpha d \rfloor$, $f = 2d$ yields an infinite family of three-stage $(N \times N)$ generalized connectors with slightly more than $4\sqrt{2}\, N^{3/2}$ switches.

6. A novel interconnection pattern. In this section we modify the design just presented to obtain yet another family of three-stage networks, and use the Erdös probabilistic method together with Monte Carlo experiments to verify that certain of these networks are generalized connectors. A particular network is specified by three positive integer parameters, d, f, and g. The first stage consists of $2d+1$ $(d \times d)$-crossbars $A(0), A(1), \cdots, A(2d)$; the second stage consists of $2d+1$ $(2d \times f)$-crossbars $B(0), B(1), \cdots, B(2d)$; and the third stage consists of f $((2d+1) \times g)$ crossbars $C(1), C(2), \cdots, C(f)$. The connections between stages two and three follow the pattern of the Slepian–Duguid–LeCorre and Pippenger–Spencer designs, and of the designs we presented in the last section: the kth sink of $B(j)$ and the jth source of $C(k)$ are hard-wired together. The novelty lies in the connections between the first and second stages: each source of a second-stage crossbar is hard-wired to one sink of some first-stage crossbar, and each sink of a first-stage crossbar is hard-wired to two sources on second-stage crossbars. Specifically, let u and v be distinct integers between 0 and $2d$, and let $i = u + v \bmod (2d+1)$. Then one of the sinks of $A(i)$ is hard-wired to the ith sources of $B(u)$ and $B(v)$.

Our networks can be presented as instances of the generic construction of §3 with $n = d(2d+1)$, $r = 2d$, and $s = 2d+1$. Let us say that the ith source of $B(u)$ and the ith source of $B(v)$ are *paired* if $i = u + v \bmod (2d+1)$. Let D be the set of second-stage sources and let S be the set of first-stage sources. It follows from the definition of the first stage that a function $K: D \to S$ is realizable if and only if, whenever y_1, the ith source of $B(u)$, and y_2, the ith source of $B(v)$, are paired, $K(y_1) = K(y_2) = x$, where x is one of the sources of $A(i)$.

To prove that a particular network in the family under discussion is a generalized connector, we introduce a probability space Ω of realizable functions. For K to lie in Ω, the following special condition must hold: $K(y_1) = K(y_2)$ only if y_1 and y_2 are paired. In other words, the realizable functions in Ω correspond to first-stage switch settings that connect each first-stage source to exactly one sink of the $(d \times d)$-crossbar on which it lies. We take all elements of Ω to be equally likely.

Just as in §5, each set A_K^x contains exactly two elements, and we can represent the SDR problem associated with request pattern F, third-stage crossbar C, and realizable function K, where $K \epsilon \Omega$, by a graph $H_{K,F}(C)$ having vertex set $\{1, 2, \cdots, 2d+1\}$ and edge set $\{A_K^x |\ x \epsilon F(C)\}$. In this case the graph has no multiple edges since, for any

two first-stage sources x_1 and x_2, $A_K^{x_1} \neq A_K^{x_2}$.

Let $H_F(C)$ be the graph-valued random variable that, at point $K \epsilon \Omega$, is equal to $H_{K.F}(C)$. We wish to show that, for every C and F, the probability that $H_F(C)$ is bad is less than $1/f$. We can attack the estimation of these probabilities using a concept similar to, but more complicated than, the concept of signature used in the preceding §. Let us say that an *ordered signature* is an array $\underline{a} = (a_0, a_1, \cdots, a_{2d})$ of nonegative integers summing to g. Associate with each ordered signature a probability space $T(\underline{a})$ of g-edge graphs on the vertex set $\{0, 1, \cdots, 2d\}$. Let $E(i)$ be the set of edges $\{u, v\}$ of K_{2d+1} such that $u + v \bmod (2d+1) = i$. Then the following stochastic experiment defines the probability distribution $T(\underline{a})$: for each i, draw a_i edges randomly, without replacement, from $E(i)$; let the edge set of the graph be the union of these edge sets, as i ranges over $\{0, 1, \cdots, 2d\}$.

LEMMA 6.1. *The following is a sufficient condition for the three-stage network determined by the parameters d, f, and g to be a generalized connector: for every ordered signature $\underline{a}$, the probability that a graph drawn from $T(\underline{a})$ is bad is less than* 1/f.

Proof. We may restrict attention to pairs C, F such that no two sinks of C request the same source. Associate with C, F the ordered signature $\underline{a} = (a_0, a_1, \cdots, a_{2d})$, where a_i is the number of distinct sources on first-stage crossbar $A(i)$ requested by sinks of C. Then the distributions of $H_F(C)$ and graphs drawn from $T(\underline{a})$ are identical. □

We need to show that for each ordered signature $\underline{a}$, the probability that a graph drawn from $T(\underline{a})$ is bad is less than $1/f$. There are two major difficulties: first, the dynamic programming technique of the last section cannot be extended to the present situation and, secondly, the number of different ordered signatures is very large. The number of ordered signatures is $\binom{2d+g}{2d+1}$. For a typical choice of parameters $(d = 7, g = 9)$, $\binom{2d+g}{2d+1} = 490,314$. We shall use a Monte Carlo approach to gather convincing statistical evidence that all the required probabilities are sufficiently small. However, it is not feasible to perform a separate Monte Carlo computation for each ordered signature. Fortunately, we can avoid doing so by using two observations.

The first observation is that certain pairs of ordered signatures are equivalent, in the sense that they have exactly the same probability of producing a bad graph. Let α and β be integers between 1 and $2d$ such that α is relatively prime to $2d + 1$. Let $\underline{a} = (a_0, a_1, \cdots, a_{2d})$ and $\underline{b} = (b_0, b_1, \cdots, b_{2d})$ be ordered signatures such that, for all i, $a_i = b_{\alpha i + \beta \bmod 2d+1}$. Then there is a bijection between the probability spaces $T(\underline{a})$ and $T(\underline{b})$ that maps bad graphs onto bad graphs: to obtain the image under the bijection of a graph Γ in $T(\underline{a})$ we replace each edge $\{u, v\}$ of Γ by the edge $\{du + \beta \cdot 2^{-1}, dv + \beta \cdot 2^{-1}\}$, where arithmetic is modulo $2d + 1$ and 2^{-1} is the inverse of 2 mod $2d + 1$. For the case $d = 7$, $g = 9$, the 490,314 signatures fall into roughly 4000 equivalence classes.

By a Monte Carlo trial, we mean the following stochastic experiment associated with an ordered signature $\underline{a}$: draw a graph from $T(\underline{a})$ and test whether it is bad. The second observation is that we can perform Monte Carlo trials for many ordered signatures with a single stochastic experiment. Define the *base signature* of the ordered signature $\underline{a}$ to be the array $(a_0', a_i', \cdots, a_{2d}')$, where, for each i, a_i' is the largest even number less than or equal to a_i. The following stochastic experiment has the effect of performing a Monte Carlo trial for each of the signatures having $\langle b_0, b_1, \ldots, b_{2d} \rangle$ as its base signature:

(i) For $i = 0, 1, \cdots, 2d$ draw b_i edges without replacement from the uniform

distribution over $E(i)$;

(ii) Let B be the set of edges drawn at step (i). For $i = 0, 1, \cdots, 2d$ draw one edge from the uniform distribution over $E(i) \backslash B$;

(iii) Let C be the set of edges drawn at step (ii). Using a backtrack search method, list all the g-edge bad graphs whose edge set includes B and is included in $B \cup C$;

(iv) For each bad graph G' listed in step (iii) for $i = 0, 1, \cdots, 2d$, let a_i be the number of edges from $E(i)$ in G' (a_i is either b_i or $b_i + 1$); record the Monte Carlo trial for the signature $\underline{a} = (a_0, a_1, \cdots, a_{2d})$ as *bad*;

(v) For all signatures that do not correspond to bad graphs listed in step (iii), the Monte Carlo trial is recorded as good (this is done implicitly, by failing to record the trial as bad).

The Monte Carlo trials for the various signatures having $\langle b_0, b_1, \cdots, b_{2d} \rangle$ as base signature are dependent, but the successive Monte Carlo trials for any given signature are independent.

Combining our two observations, we can consider two base signatures $\langle b_0, b_1, \cdots, b_{2d} \rangle$ and $\langle b'_0, b'_1, \cdots, b'_{2d} \rangle$ equivalent if, for some integers α and β between 1 and $2d$ such that α is relatively prime to $2d+1$, and for all i, $b_i = b'_{\alpha i + \beta \bmod 2d+1}$. To perform a Monte Carlo trial for at least one ordered signature in each equivalence class of ordered signatures, it suffices to perform a Monte Carlo trial for one base signature in each equivalence classes of base signatures.

The number of equivalence classes of base signatures is rather small. For example, in the case where $d = 7$, $g = 9$, there are 490,314 ordered signatures, about 4000 equivalence classes of ordered signatures and 56 equivalence classes of base signatures. By working with equivalence classes of base signatures we can perform one Monte Carlo trial for each of the 490,314 ordered signatures by performing only 56 simple stochastic experiments.

The approach we have outlined makes it feasible to obtain convincing "statistical proofs" that certain networks in our family are generalized connectors. A typical example is $d = 7$, $f = 11$, $g = 9$, corresponding to a 105×99 generalized connector. It took 70 minutes on a SUN/350 (or, alternatively, a bit over 4 minutes on a network of 15 SUN/350s) to conduct 1000 Monte Carlo trials for each of the 56 equivalence classes of base signatures (and thus, implicitly, for each of the 490,314 ordered signatures). Out of the 1000 trials, the largest number of bad trials for any ordered signature was 63. There were 12 equivalence classes of ordered signatures for which 50 or more bad trials were recorded. For each of these, 1000 further trials were carried out, and in no case were more than 50 bad trials recorded. This provides convincing evidence that, for each ordered signature, the probability of a bad graph is less than 1/11 and, therefore, that our network is a generalized connector. The number of switches in this case is 4530. Another parameter choice that has similarly been "proven" to yield a generalized connector is $d = 8$, $f = 12$, $g = 10$, corresponding to a 136×120 generalized connector with 6392 switches.

Finally, we turn to the question of devising a "routing algorithm" that, for any given request pattern F, constructs a network configuration satisfying F. An obvious approach is to keep drawing realizable functions from Ω until one is found with the property that the SDR problems associated with third-stage crossbars are all solvable. There is a simple recipe for constructing the required network configuration once such a realizable function is at hand.

However, it is expensive to apply such a procedure each time a request is added to

the current request pattern. Moreover, it is inherent in this algorithm that changing a single request is likely to cause a radical rearrangement of the network. Even though such rearrangements are allowed since we are not demanding that the network be nonblocking, it is certainly preferable in practice to minimize the number of switch settings that change when a single request is made.

Extensive simulations conducted by the staff of DVA Ltd. indicate that a much simpler procedure works extremely well in practice. Consider a point in time when the request pattern is F, the network is in a configuration satisfying F, and the first-stage configuration corresponds to the realizable function K. Now suppose one sink on third-stage crossbar, say $C(i)$, changes its request. Let the new request pattern be F'. For every third-stage crossbar C different from $C(i)$, the graphs $H_{K,F'}(C)$ and $H_{K,F}(C)$ are the same, and, by the assumption that the present configuration satisfies F, these graphs are good. The graph $H_{K,F'}(C(i))$, which differs from $H_{K,F}(C(i))$ by the deletion of one edge and the insertion of another, may be bad. If $H_{K,F'}(C(i))$ is good, then the settings of the second-stage and third-stage crossbars can easily be changed to satisfy F', without altering the first-stage settings; moreover, the required change will not alter the paths used to satisfy the requests on the third-stage crossbars other than $C(i)$. If $H_{K,F'}(C(i))$ is bad, then it is necessary to change the first-stage switch settings, and this is done using the concept of an interchange, which we now define. Let $A(i)$ be a first-stage crossbar. Let x_1 and x_2 be two sources on $A(i)$, y_1 and y_2 be two sinks on $A(i)$, and consider a configuration in which x_1 is connected to y_1 and x_2 to y_2. Define an *interchange* as the process of changing the configuration so that x_1 is connected to y_2, and x_2 to y_1. Let K and K', respectively, be the functions realized by the first-stage configurations before and after the interchange. Then $A^{x_1}_{K'} = A^{x_2}_K$, $A^{x_2}_{K'} = A^{x_1}_K$, and, for all x except x_1 and x_2, $A^x_K = A^x_{K'}$. Thus, the effect of the interchange is to exchange the roles of the edges $A^{x_1}_K$ and $A^{x_2}_K$. If graph $H_{K,F'}(C)$ contains neither or both of these edges, then it is unaffected by the interchange; i.e., $H_{K,F'}(C) = H_{K,F}(C)$. On the other hand, if $H_{K,F'}(C)$ contains one of these two edges, then that edge is replaced by the other.

If the new request F' makes $H_{K,F'}(C(i))$ bad, then the algorithm looks for an interchange that will make $H_{K',F'}(C(i))$ good. Among such interchanges, it gives preference to one that does not create any bad graphs, i.e., one such that, for all C, $H_{K',F'}(C)$ is good, where K' is the function realized by the first stage after the interchange. Usually, this will be possible. If not, the algorithm chooses an interchange that makes the graph associated with $C(i)$ good and minimizes the number of third-stage crossbars whose graphs become bad. It then seeks one or more interchanges to correct these newly formed bad graphs one at a time, always choosing an interchange that corrects the bad graph under consideration and creates as few bad graphs as possible. The process continues until no bad graphs remain. A few interchanges have always sufficed to eliminate all bad graphs.

7. An "almost strictly nonblocking" two-stage network. Throughout this paper we have been concerned exclusively with rearrangeable networks. Such a network is capable of satisfying any request pattern F; however, a change in the source requested by one sink may necessitate a global change in the configuration of the network. In some applications a more stringent requirement is imposed. A connector or generalized connector is said to be *strictly nonblocking* if, whenever it is in a configuration satisfying a request pattern F and one output z changes its request, it is possible to satisfy the new request pattern in such a way that, for every output y different from z, the path connecting $F(y)$ to y is not changed. There is also the

weaker concept of a *wide-sense nonblocking* network; this means that the above requirement holds, not for every possible configuration, but for every configuration that can occur when a particular routing algorithm is used. In [1] and, independently, in [5], it is proved that a strictly nonblocking $(n \times n)$-generalized connector must have $\Omega(n^2)$ switches, and a strictly nonblocking $(n \times n)$-connector with depth k must have $\Omega(n^{1+\frac{1}{k-1}})$ switches. This result suggests the following questions: for which c does there exist a family of strictly non-blocking $n \times n$ generalized connectors (respectively, connectors) of depth at most 2 with $Cn^2 + o(n^2)$ switches? Of course, for all k, $C = 1$ is achievable using an $(n \times n)$-crossbar. In this section, we exhibit a family of depth-two $(n \times n)$-generalized connectors that are almost strictly nonblocking in the sense that, when a single output z changes its request, at most one input-output path, other than the one terminating at z, must be changed. The number of switches required by the networks in this family is $\frac{2}{5}n^2 + 2n^{3/2}$.

A particular network in the family is specified by positive integer parameters d and f; the number of inputs is equal to d^2, and the number of outputs is $5h$. To enable comparison with the generic construction of §3, we describe our design as a three-stage network; however, the first stage contains no switches, so the network is indeed of depth two. The second stage consists of $2d$ $(d \times f)$-crossbars $B(1), B(2), \cdots, B(2d)$, and the third stage consists of f $(2d \times 5)$-crossbars $C(1), C(2), \cdots, C(f)$; as usual, the kth sink of $B(j)$ is hard-wired to the jth source of $C(k)$. With each source of the overall network is associated a pair (u, v), where $u \epsilon \{1, 2, \cdots, d\}$ and $v \epsilon \{d + 1, \cdots, 2d\}$.The source is hard-wired to a source of $B(u)$ and a source of $B(v)$; each pair $(u, v) \epsilon \{1, 2, \ldots, d\} \times \{d + 1, \cdots, 2d\}$ is associated with exactly one source of the network. Thus the first stage realizes a single function K; if a source x is connected to $B(u)$ and $B(v)$, then $A_K^x = \{u, v\}$. For every third-stage crossbar C and every request pattern F, the graph $H_{K,F}(C)$ is a bipartite graph with five distinct edges, each of which connects a vertex in $\{1, 2, \cdots, d\}$ with a vertex in $\{d+1, d+2, \cdots, 2d\}$. Since no five-edge bipartite graph can be bad, the network is a generalized connector. We leave it to the reader to verify that, when an output on third-stage crossbar C changes its request, at most one of the other outputs on C, and no output on any of the other third-stage crossbars, needs to be served by a different input-output path.

In the case where $n = m = d^2 = 5f$, the network is a depth-two $(n \times n)$-generalized connector with $\frac{2}{5}n^2 + 2n^{3/2}$ switches. This is uninteresting as an asymptotic result, but is competitive in switch count with the known three-stage designs when n is less than 50.

8. Conclusion. Table 1 compares the various generalized connectors presented in this paper by giving the number of switches required when n and m are close to 100.

TABLE 1

Design	n	m	Number of switches	Number of stages
Dolev et al.	100	100	4000	4
Pippenger–Spencer	100	100	9000	3
Karp (§5)	100	102	5420	3
Karp (§6)	105	99	4530	3
Karp (§7)	100	100	6000	2

A peculiarity of this paper is that, except in §7, we establish the correctness of

particular networks, with fixed numbers of inputs and outputs, rather than proving the correctness of entire families, as is usually done. Moreover, for the networks considered in §6, we depend on stochastic experiments that are only valid if we trust our pseudorandom number generator. We do not know of any fully satisfactory way around this difficulty, which is endemic to the fields of simulation and Monte Carlo computation.

Acknowledgments. Paul Wible of DVA Ltd. stimulated the author to work on generalized connectors. Nick Pippenger was the guide through the lore of connection networks, and he and Joel Spencer provided access to their unpublished work. Jordan Lampe was a full partner in designing the dynamic programming algorithm of §5 and the efficient search algorithm of §6 based on equivalence classes of base signatures. He also very ably programmed all the Monte Carlo experiments. The author thanks these people for their substantial help.

REFERENCES

[1] L. Bassalygo and M. Pinsker, *Asymptotically optimal networks for generalized rearrangeable switching and generalized switching without rearrangement*, Problemy Pederachi Informatsii, 16 (1980), pp. 94–98.

[2] V. Benes, *On rearrangeable three-stage connecting networks*, Bell Syst. Tech. J., 41 (1962), pp. 1481–1492.

[3] R. Cole and J. Hopcroft, SIAM J. Comput, 11 (1982), pp. 540–546.

[4] D. Dolev, C. Dwork, N. Pippenger, and A. Wigderson, *Superconcentrators, generalizers and generalized connectors with limited depth*, in Proc. of the 15th Annual ACM Symp. on the Theory of Computing, 1983, pp. 42–51.

[5] P. Feldman, J. Friedman, and N. Pippenger, *Non-blocking networks*, in Proc. of the 18th Annual ACM Symposium on the Theory of Computing, 1986, pp. 247–254.

[6] M. Kirkpatrick, M. Klawe, and N. Pippenger, *Some graph coloring theorems with applications to generalized connection networks*, SIAM J. Algebraic Discrete Meth., 6 (1985), pp. 576–582.

[7] G. Masson and B. Jordan, *Generalized multi-stage connection networks*, Networks, 2 (1972), pp. 191–209.

[8] D. Nassimi and S. Sahni, *Parallel permutation and sorting algorithms and a new generalized connection network*, J. Assoc. Comput. Mach., 29 (1982), pp. 642–667.

[9] N. Pippenger and A.-C. Yao, *Rearrangeable networks with limited depth*, SIAM J. Algebraic Discrete Meth., 3 (1982), pp. 411–417.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 273–284, May 1992
© 1992 Society for Industrial and Applied Mathematics
009

PRIVACY AND COMMUNICATION COMPLEXITY*

EYAL KUSHILEVITZ†

Abstract. Each of two parties $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ holds an n-bit input, x and y, respectively. They wish to *privately* compute the value of $f(x, y)$. That is, $P_{\mathcal{X}}$ should not learn any additional information about y (in the information-theoretic sense) other than what follows from its input x and the function value $f(x, y)$, and similarly, $P_{\mathcal{Y}}$ should not learn any additional information about x.

In this paper, the two following basic questions in the theory of private computations are considered:

1. Which functions can be privately computed?
2. What is the communication complexity of protocols that privately compute a function f (in the case that such protocols exist)?

A complete combinatorial characterization of privately computable functions is given. This characterization is used to derive tight bounds on the rounds complexity of any privately computable function and to design optimal private protocols that compute these functions.

It is shown that for every $1 \leq g(n) \leq 2 \cdot (2^n - 1)$ there are functions that can be privately computed with $g(n)$-rounds of communication, but not with $(g(n) - 1)$-rounds of communication. This implies that the communication costs of private protocols can be exponentially higher than the communication costs of nonprivate protocols.

Interestingly, randomization helps neither to increase the set of privately computable functions, nor to improve the rounds complexity of these functions.

Key words. private distributed computations, communication complexity

AMS(MOS) subject classifications. 94A15, 94A60, 68R05

1. Introduction.

The topic of this paper is private computations and their communication complexity. To exemplify the issue, we consider the "two millionaires" problem presented by Yao [17]. Two millionaires wish to know who is the richer. However, they want to do this in a way such that neither of them will receive any additional information about the other's wealth. Can the two millionaires solve their problem?

The general question that we address is which functions of two arguments can be computed in such a way that no party learns any additional knowledge, other than what follows from the value of the function and its input. The result is that the answer to this question relies heavily on the assumptions that are made regarding the computational power of the parties. One possible approach is to assume that the parties are limited to efficient computations (i.e., in polynomial time). The issue of privacy, using this approach, was resolved in [19], [11], under (unproved) intractability assumptions.

Another possible approach is the information theoretic approach. The computational power of the parties is not restricted, and no intractability assumptions are made. Thus the notion of privacy is much stronger. It is not only that the parties cannot obtain additional information using polynomial-time computations, but that such information cannot be obtained at all. This approach was studied in [5], [6]. It was shown that any function f of N arguments can be computed $\lfloor (N-1)/2 \rfloor$-privately. No coalition of $t \leq \lfloor (N-1)/2 \rfloor$ parties learns anything, other than the value of the function from the execution of the protocol. (Note that for the case

* Received by the editors April 16, 1990; accepted for publication (in revised form) March 14, 1991. An early version of this paper appeared in [*Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, Research Triangle Park, North Carolina, 1989, pp. 416–421]. This research was supported by United States–Israel Binational Science Foundation grant 88-00282.

† Department of Computer Science, Technion–Israel Institute of Technology, Haifa 32000, Israel.

where $N = 2$ this claim is not meaningful.) On the other hand, for $t > \lfloor (N-1)/2 \rfloor$, Ben-Or, Goldwasser, and Wigderson [5] showed that some functions cannot be computed t-privately. In [8] a complete characterization of the t-private Boolean functions, for $\lfloor (N-1)/2 \rfloor < t \leq N$ is given. The basis for this characterization was a reduction from the multiparty case to the two-party case. In the two-party case, it is shown that a *Boolean* function f is 1-private if and only if it can be expressed as $f(x,y) = h(x) \oplus g(y)$, where h and g are also Boolean functions, and $\oplus$ denotes exclusive-or [8].

In this paper we solve the question of privacy with respect to *arbitrary* two-argument functions: A *complete combinatorial characterization* of the privately computable functions is given. In particular, this provides a necessary condition for privacy in the general *multiparty* case. This characterization was independently found by Beaver [3].

A new facet of privacy, considered in this paper, is the communication complexity of computing functions privately. Two measures of complexity are considered in this paper: communication complexity (number of bits) and rounds complexity. The communication complexity of computing functions in a two-party system was extensively studied in previous works. Yao, for example, investigated the communication complexity of computing arbitrary Boolean functions in such a system (both in a deterministic model and a probabilistic model) [16], [18]. Tight $\Theta(n)$ bounds on the communication complexity of explicit functions and of random functions were given both for the case of deterministic protocols [16] and for randomized protocols [1], [7], [12]. Rounds complexity was studied by Papadimitriou and Sipser [15] and by Duris, Galil, and Schnitger [9]. They showed that for certain functions there is an exponential gap between the number of bits that must be exchanged using k-round protocols and $k+1$ -round protocols.

We use the characterization of privately computable functions to derive tight bounds on the communication complexity and rounds complexity of these functions. We show that the privately computable functions form a very dense rounds-complexity hierarchy. For every $1 \leq g(n) \leq 2 \cdot (2^n - 1)$, there exists a function that is privately computable by a $g(n)$-round protocol but cannot be privately computed by any $(g(n)-1)$ -round protocol. In particular, certain functions require $2 \cdot (2^n - 1)$ communication rounds and $\Theta(2^n)$ bits to be privately computed. This should be contrasted with the fact that in a regular computation (that is, without privacy constraints), any function f can be computed using $O(n)$ bits and two communication rounds ($P_{\mathcal{X}}$ sends x; $P_{\mathcal{Y}}$ computes and sends $f(x,y)$). Comparing these two bounds we conclude that even when a function is privately computable, the "cost" of privacy may be exponentially larger.

In distributed computing, randomization often increases the computation power [13], [2], or significantly decreases computational costs [4], [10], [14]. Interestingly, this is not the case for private two-party computation. For every privately computable function f, we present a *deterministic* protocol, which privately computes f in an *optimal* number of rounds (and deterministically optimal number of bits). We conclude that randomization helps neither to increase the set of privately computable functions nor to improve the rounds complexity of privately computable functions.

The rest of this paper is organized as follows. In §2 we present the model and the definitions of privacy. In §3 we give the characterization of the privately computable functions, and in §4 we deal with the communication complexity and rounds complexity of these functions. Finally, in §5 we discuss some connections of the results

with multiparty private protocols.

2. Preliminaries.

2.1. Two-party protocols and communication complexity (without privacy). In this section we define the model of two-party computation and the notions of complexity that are used in this paper. These definitions are originally due to Yao [16].

The model consists of two parties, $P_{\mathcal{X}}$—holding n-bit input x—and $P_{\mathcal{Y}}$—holding n-bit input y. $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ wish to compute the value of $f(x,y)$ ($f : \{0,1\}^n \times \{0,1\}^n \to \{0,1,\cdots,m-1\}$) using a probabilistic protocol. A *probabilistic protocol* [16] for computing $f(x,y)$ is a predetermined probabilistic program. That is, the parties are allowed to toss coins during their local computations. The parties are alternately sending messages to each other. The message q_i, sent by a party in the ith round, is a function of its input, its coin tosses, and the messages it has received so far $(q_1,\cdots,q_{i-1})$. We assume that the parties are honest; that is, they follow their predetermined programs. The last message sent in the protocol $\mathcal{A}$ is assumed to contain the value of the function and is denoted $\mathcal{A}(x,y)$. We say that the protocol $\mathcal{A}$ computes the function f if

$$\forall x,y \in \{0,1\}^n : \quad \Pr(\mathcal{A}(x,y) = f(x,y)) > \tfrac{1}{2},$$

where the probability is taken over the random coin tosses of the two parties.

The *communication string* passed in the protocol is the concatenation of all the messages $q_1 \cdot q_2 \cdot \cdots \cdot q_k$ sent in the course of the protocol. We assume that in every round, the set of all possible messages forms a prefix-free code. Thus the communication string can be uniquely decomposed to its messages. It also enables a party that receives a message to recognize its end. The influence of this assumption on the communication complexity of the computation is bounded by a constant.

The *communication complexity* of a protocol $\mathcal{A}$ is the maximal number of bits transmitted during the execution of the protocol $\mathcal{A}$ (where the maximum is taken over all the possible inputs (x,y), and all the possible coin tosses). Similarly, the *rounds complexity* of a protocol $\mathcal{A}$ is the maximal number of rounds in any run of the protocol $\mathcal{A}$.

The *communication complexity* (*rounds complexity*) of a function f is the minimum communication complexity (rounds complexity) over all protocols that compute f.

Remark 1. Since the last message of the protocol should contain the value of $f(x,y)$, it was assumed that if the image of f is of size m, then the image is the set $\{0,1,2,\cdots,m-1\}$. If this is not the case, we can use, in the last message, some encoding of $f(x,y)$ into the set $\{0,1,\cdots,m-1\}$. Therefore, the length of the last message is at most $\log_2 m$ bits (note that $m \le 2^{2n}$ and therefore $\log_2 m \le 2n$).

Remark 2. The assumption that $|x| = |y|$ is not essential for any of our results and can be easily relaxed.

It is convenient to visualize any function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1,\cdots,m-1\}$ as a $2^n \times 2^n$ matrix with entries in $\{0,1,\cdots,m-1\}$. We denote this matrix by M_f. Each row of M_f represents an input x held by $P_{\mathcal{X}}$, and each column of M_f represents an input y held by $P_{\mathcal{Y}}$. The entry (x,y) of the matrix M_f contains the value $f(x,y)$. A submatrix of M_f is called *monochromatic* if f is constant over it.

2.2. The privacy constraints. In this section we formally define the notions of weak and strong privacy in a two-party distributed system. The definitions are

taken from [8], and the definition of strong privacy is equivalent to the definition of [5].

Informally, a protocol is private if each party does not learn anything from the execution of the protocol other than the value of the function. This means that the parties do not gain even a probabilistic advantage over what they know by themselves.

Formally, we call a protocol *weakly private with respect to* $P_{\mathcal{X}}$ (a similar definition holds for $P_{\mathcal{Y}}$) if for every two inputs (x, y_1) and (x, y_2) satisfying $f(x, y_1) = f(x, y_2)$, and for every communication string s

$$\Pr(s|(x, y_1)) = \Pr(s|(x, y_2)),$$

where the probability is taken over all the possible coin tosses of both parties.

Now, we make the notion of privacy stronger in two ways. First, we require that the protocols will not make errors, and, second, we require that the parties not learn anything from the execution of the protocol even when taking into account their random coin tosses. Formally, we say that a protocol $\mathcal{A}$ is *strongly private with respect to* $P_{\mathcal{X}}$ (a similar definition holds for $P_{\mathcal{Y}}$) if $\mathcal{A}$ always computes the correct value of the function (that is, $\mathcal{A}(x, y) = f(x, y)$), and if for every two inputs (x, y_1) and (x, y_2) satisfying $f(x, y_1) = f(x, y_2)$, every string of coin tosses $r_{\mathcal{X}}$ held by $P_{\mathcal{X}}$, and for every communication string s,

$$\Pr(s|r_{\mathcal{X}}, (x, y_1)) = \Pr(s|r_{\mathcal{X}}, (x, y_2)),$$

where the probability is taken over all the possible coin tosses of $P_{\mathcal{Y}}$.

A protocol $\mathcal{A}$ is called *weakly/strongly private* if it is weakly/strongly private with respect to both parties. A function f is called *weakly/strongly private* if there exists a weakly/strongly private protocol that computes it.

Finally, we remark that when dealing with the communication/rounds complexity of private functions, we consider only private protocols.

3. Characterization of private functions. In this section we give a complete combinatorial characterization of the privately computable functions. This characterization is used to derive a deterministic private protocol for any privately computable function. We start with some definitions.

DEFINITION 1. Let $M = C \times D$ be a matrix (C is the set of rows and D is the set of columns). The relation $\sim$ on the rows of the matrix M is defined as follows: $x_1, x_2 \in C$ satisfies $x_1 \sim x_2$ if there exists $y \in D$ such that $M_{x_1,y} = M_{x_2,y}$. The equivalence relation $\equiv$ on the rows of the matrix M is defined as the transitive closure of the relation $\sim$. That is, $x_1, x_2 \in C$ satisfies $x_1 \equiv x_2$ if there exist $z_1, z_2, \cdots, z_\ell \in C$ such that $x_1 \sim z_1 \sim z_2 \sim \cdots \sim z_\ell \sim x_2$. Similarly, the relations $\sim$ and $\equiv$ are defined on the columns of the matrix.

DEFINITION 2. A matrix M is called *forbidden* if it is not monochromatic, all its rows are equivalent, and all its columns are equivalent. That is, every $x_1, x_2 \in C$ satisfies $x_1 \equiv x_2$, and every $y_1, y_2 \in D$ satisfies $y_1 \equiv y_2$. (For examples of forbidden matrices, see Fig. 1).

The first theorem claims that if a function f is represented by a matrix M_f that contains a forbidden submatrix, then f is not privately computable.

THEOREM 3.1. *Let f be a function. If M_f contains a forbidden submatrix $M = C \times D$ then f is not (weakly) private.*

Proof. The idea of the proof is to show that any protocol $\mathcal{A}$, trying to compute f privately, fails (with "high" probability) in computing the correct value of the function

	y_1	y_2
x_1	0	0
x_2	0	1

	y_1	y_2	y_3
x_1	0	0	1
x_2	2	4	1
x_3	2	3	3

FIG. 1. *Forbidden matrices.*

for some inputs. Intuitively, this is done by showing that for any $x_1, x_2 \in C$, and any message q, the probability that $P_{\mathcal{X}}$ sends q on input x_1 is equal to the probability that $P_{\mathcal{X}}$ sends q on input x_2. An analogous argument holds for $P_{\mathcal{Y}}$ with respect to any $y_1, y_2 \in D$. Thus, the probability distribution on communication strings is the same for any $(x, y) \in M$. Since M is not monochromatic, and since the protocol is required to compute the correct value of f for every input with probability greater than $\frac{1}{2}$, then for some of the inputs the protocol fails.

Let $C = \{x_1, \cdots, x_\ell\}$ and $D = \{y_1, \cdots, y_p\}$. Let $s = q_1 \cdot q_2 \cdot \cdots \cdot q_k$ be any communication string. We prove, by induction on the round number t, that for every $(x_i, y_j) \in M$,

$$\Pr(q_1, \cdots, q_t | (x_i, y_j)) = \Pr(q_1, \cdots, q_t | (x_1, y_1)). \tag{1}$$

This implies that for every $(x, y) \in M$ the probability $Pr(s|(x, y))$ is the same. The induction assumption implies that for every $(x, y) \in M$ the probability $\Pr(q_1, \cdots, q_{t-1}|(x, y))$ is the same. If this probability is zero then for every $(x, y) \in M$ we have $\Pr(s|(x, y)) = 0$. Otherwise, we consider the tth round $(1 \leq t \leq k)$ and assume, without loss of generality, that t is odd. Therefore the message q_t is sent by $P_{\mathcal{X}}$ (a similar argument holds for the messages of $P_{\mathcal{Y}}$). The submatrix $M = C \times D$ is forbidden; hence, all the rows of M are equivalent. Therefore we can also assume that the set C is ordered in a way such that for every i $(i \geq 2)$ there exists $j < i$ such that $x_i \sim x_j$. We now prove by induction on i that for every $1 \leq i \leq \ell$,

$$\Pr(q_t | x_i, q_1, \cdots, q_{t-1}) = \Pr(q_t | x_1, q_1, \cdots, q_{t-1}) \ . \tag{2}$$

Given $i > 1$, there is some $j < i$ with $x_i \sim x_j$. By the definition of $\sim$, there exists $y \in D$ such that $f(x_i, y) = f(x_j, y)$. According to the definition of (weak) privacy, this implies that for every communication string s, Pr $(s|(x_i, y)) = \Pr(s|(x_j, y))$. In particular, $\Pr(q_1, \cdots, q_t|(x_i, y)) = \Pr(q_1, \cdots, q_t|(x_j, y))$. However,

$$\Pr(q_1, \cdots, q_t | (x, y)) = \Pr(q_1, \cdots, q_{t-1} | (x, y)) \cdot \Pr(q_t | x, q_1, \ldots, q_{t-1}). \tag{3}$$

By the assumptions, $\Pr(q_1, \cdots, q_{t-1}|(x_i, y)) = \Pr(q_1, \cdots, q_{t-1}|(x_j, y)) \neq 0$. Therefore, by (3) and the privacy condition, $\Pr(q_t|x_i, q_1, \cdots, q_{t-1}) = \Pr(q_t|x_j, q_1, \cdots, q_{t-1})$, which implies by the induction hypothesis (2) that Pr $(q_t|x_i, q_1, \cdots, q_{t-1})$ $= \Pr(q_t|x_1, q_1, \cdots, q_{t-1})$. This completes the proof of claim (2).

By the induction hypothesis (1) and claim (2) we get that for every $(x_i, y_j) \in M$,

$$\Pr(q_1, \cdots, q_t | (x_i, y_j)) = \Pr(q_1, \cdots, q_t | (x_1, y_1)) \ .$$

This completes the proof of (1). Therefore, for every input $(x_i, y_j) \in M$, the probability distributions of communication strings sent on these inputs are all equal. In particular, for every $(x_i, y_j) \in M$ the last message of the communication string, which is $\mathcal{A}(x_i, y_j)$, is distributed in the same way. On the other hand, the correctness of $\mathcal{A}$ implies that for every (x_i, y_j),

$$\Pr(\mathcal{A}(x_i, y_j) = f(x_i, y_j)) > \tfrac{1}{2}.$$

Thus the same function value is computed for all the inputs in M, in contradiction to the assumption that M is not monochromatic. □

The special case of Theorem 3.1, where M is a 2×2 submatrix, is useful for proving that particular functions are not private. It says that if there exist $x_1, x_2, y_1, y_2 \in \{0,1\}^n$ such that $f(x_1, y_1) = f(x_1, y_2) = f(x_2, y_1) = a$, and $f(x_2, y_2) \neq a$, then f is not private. We can now prove, for example, that the "greater equal" function ($f(x, y) = 1 \iff x \geq y$) is not private. It is enough to observe that for every integer a we have $f(a-1, a) = f(a-1, a+1) = f(a, a+1) = 0$, but $f(a, a) = 1$. Note that finding a private protocol for the "greater than" function is actually solving the "two millionaires" problem. We remark that in the computational model this problem can be solved [17].

We will now prove that the condition of Theorem 3.1 is not only necessary but also sufficient. Namely, if the matrix M_f does not contain a forbidden submatrix then f is privately computable. Before proving this claim, we introduce some new definitions.

DEFINITION 3. A matrix $C \times D$ is called *rows decomposable* if there exist nonempty sets $C_1, C_2, \cdots, C_t$ ($t \geq 2$) such that

1. $C_1, C_2, \cdots, C_t$ are a partition of C. That is, $\bigcup_{i=1}^{t} C_i = C$ and for all $i \neq j$: $C_i \cap C_j = \emptyset$;
2. For every $x_1, x_2 \in C$, if $x_1 \sim x_2$ then x_1 and x_2 are in the same C_i.

Similarly, we can define when a matrix $C \times D$ is *columns decomposable* . When we say *the optimal rows* (*columns*) *decomposition* of a matrix we mean the rows (columns) decomposition that maximizes t (the number of sets). Note that the optimal decomposition is unique. This is because the sets of the optimal decomposition are exactly the equivalence classes of the relation $\equiv$ (in general, the sets of any decomposition are unions of equivalence classes of $\equiv$).

DEFINITION 4. A matrix $C \times D$ is called *decomposable* if one of the following conditions holds:

1. $C \times D$ is monochromatic.
2. $C \times D$ is rows decomposable to submatrices $C_1 \times D, C_2 \times D, \cdots, C_t \times D$, which are all decomposable.
3. $C \times D$ is columns decomposable to submatrices $C \times D_1, C \times D_2, \cdots, C \times D_t$, which are all decomposable.

An example of a decomposable matrix is given in Fig. 2 (§4).

Given a matrix M, it is easy to check whether it is decomposable, or rows (columns) decomposable, and to find the optimal decomposition. Such an algorithm will be needed for the design of efficient private protocols. For example, the following algorithm checks if M ($n \times m$ matrix) is rows decomposable and finds the optimal rows decomposition:

1. For every row $1 \leq i \leq n$, create a set $C_i = \{i\}$.
2. If there exist $x_1 \in C_i, x_2 \in C_j$ ($i \neq j$) and y such that $f(x_1, y) = f(x_2, y)$ (i.e., $x_1 \sim x_2$) then $C_i = C_i \cup C_j$ (that is, unite C_i and C_j to a single set) and return to step (2).

3. If the number of sets is 1−, there is no rows decomposition. Else, the current sets are the optimal rows decomposition of M.

To prove the sufficiency of our condition, we will use the following lemma, which claims that a matrix is decomposable if and only if it does not contain a forbidden submatrix.

LEMMA 3.2. *Let $M = C \times D$. M is decomposable if and only if for every submatrix M' of M, M' is not forbidden.*

Proof. If M contains a forbidden submatrix M', then by induction we can show that the rows and columns of this submatrix always remain in the same submatrix of the decomposition. As a decomposition must end with monochromatic submatrices, M is not decomposable. Conversely, if M is not decomposable, then we have a submatrix M', which is not monochromatic, not rows decomposable, and not columns decomposable. This means that M' is a forbidden submatrix of M. □

THEOREM 3.3. *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1,\cdots,m-1\}$ be an arbitrary function. If M_f does not contain a forbidden submatrix then f is (strongly) private.*

Proof. If M_f does not contain a forbidden submatrix, then according to Lemma 3.2, M_f is decomposable. We use the optimal decomposition of M_f for presenting a deterministic private protocol $\mathcal{A}_f$ which computes f. We assume that the two parties have a common way of numbering the sets in the decomposition. The protocol is as follows.

Protocol $\mathcal{A}_f$.

1. $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ both set $C = D = \{0,1,\cdots,2^n-1\}$.
2. While $C \times D$ is not monochromatic,
   (a) $P_{\mathcal{X}}$ sends the j such that $x \in C_j$ in the optimal rows decomposition of $C \times D$. Both $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ set $C = C_j$.
   (b) $P_{\mathcal{Y}}$ sends the j such that $y \in D_j$ in the optimal columns decomposition of $C \times D$. Both $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ set $D = D_j$.
3. $P_{\mathcal{X}}$ sends the constant value in $C \times D$ as the value of $f(x,y)$.

From the definition of a decomposable matrix, it is clear that during this algorithm $C \times D$ is always decomposable and that the protocol $\mathcal{A}_f$ terminates. Since the input (x,y) always belongs to $C \times D$, then it follows that $\mathcal{A}_f$ terminates with the correct value of $f(x,y)$. (Note that the first time step 2(a) is performed, it is possible that $C \times D$ is not rows decomposable. In such a case, $C \times D$ must be columns decomposable, and we can start from step 2(b). From this point on, since we use the optimal decomposition, $C \times D$ is rows decomposable and columns decomposable alternately.) Since $\mathcal{A}_f$ is deterministic, then for every input (x,y), the communication string is unique. Therefore, for showing that $\mathcal{A}_f$ is (strongly) private, we should verify that for every two inputs (x,y_1) and (x,y_2) that satisfy $f(x,y_1) = f(x,y_2)$ the same communication string is exchanged. This is true since the messages sent in step 2(a) depend only on x, while, according to the definition of columns decomposition, the messages sent in step 2(b) are uniquely determined by x and the function value (note that $y_1 \sim y_2$ in every submatrix $C \times D$ that contains (x,y_1) and (x,y_2), and thus they will never be decomposed). Similarly, we can verify that for every two inputs (x_1,y) and (x_2,y) that satisfy $f(x_1,y) = f(x_2,y)$, the same communication string is exchanged. □

From Theorems 3.1 and 3.3, we obtain the following corollaries.

COROLLARY 3.4. *If f is weakly private, then f is also strongly private.*

If f is weakly private then, according to Theorem 3.1, M_f does not contain a forbidden submatrix. Therefore, according to Theorem 3.3, f is strongly private. In

particular, this means that the ability of the protocols to make errors does not help to achieve privacy.

COROLLARY 3.5. *If f is private, then it can be privately computed using a deterministic protocol.*

According to the proof of Theorem 3.3, every private function is privately computable using a deterministic protocol. That is, randomization does not help to achieve privacy in the two-party case. We emphasize that this is not the case for multiparty protocols (see [5], [8]).

4. The communication complexity of private functions. In this section we deal with the communication complexity of private functions. We show a correspondence between *every* protocol $\mathcal{A}$, which computes a function f privately, and a decomposition of the matrix M_f. This correspondence is used for proving lower bounds on the number of rounds and the number of communication strings needed for computing f privately.

It may be convenient to look at the decomposition of a matrix M, as a tree containing submatrices in its nodes. The root (level 0) of the tree contains the given matrix M. The children of a node v in an even level of the tree contain a rows decomposition of the submatrix in v, while the children of a node in an odd level contain a columns decomposition of the submatrix. All the leaves of the tree are monochromatic submatrices. Such a tree is called a *decomposition tree* of the matrix M. The tree corresponding to the optimal decomposition is called the *optimal decomposition tree.* (We assume without loss of generality that M itself is rows decomposable. Otherwise, in the even levels we should use columns decomposition and in the odd levels rows decomposition.)

Following Yao [16], we visualize every protocol that computes f as a decision tree. We show that the decision tree of any private protocol is also a decomposition tree. Given $\mathcal{A}$, a (weakly) private protocol for computing f, we describe how to create a decomposition tree that corresponds to $\mathcal{A}$. The root (level 0) contains the matrix M_f. Let v be a node in level ℓ of the tree containing a submatrix $M = C \times D$. We define its children as follows. Assume that the message in round $\ell+1$ of the protocol should be sent by $P_{\mathcal{X}}$ (a similar definition holds for the case where the next message should be sent by $P_{\mathcal{Y}}$). Every $x_1, x_2 \in C$, for which $P_{\mathcal{X}}$ behaves "similarly" will be in the same C_i, where "similarly" means that for every message $q_{\ell+1}$ and for every sequence of messages $q_1, q_2, \cdots, q_\ell$ passed in previous rounds (with a positive probability),

$$\Pr(q_{\ell+1}|x_1, q_1, q_2, \cdots, q_\ell) = \Pr(q_{\ell+1}|x_2, q_1, q_2, \cdots, q_\ell).$$

Since it is given that $\mathcal{A}$ is (weakly) private, then $f(x_1, y) = f(x_2, y)$ implies that for every communication string s we have $\Pr(s|(x_1, y)) = \Pr(s|(x_2, y))$. Thus, the probability $\Pr(q_{\ell+1}|x_1, q_1, q_2, \cdots, q_\ell)$ is equal to $\Pr(q_{\ell+1}|x_2, q_1, q_2, \cdots, q_\ell)$. Therefore, if $x_1 \sim x_2$, then x_1 and x_2 are in the same C_i, and the conditions for rows decomposition hold. Finally, we must show that the leaves contain monochromatic submatrices. This follows from the fact that the same probability distribution of messages is sent for every element of the submatrix, and that for every input the protocol should compute the correct value of the function with probability greater then $\frac{1}{2}$. (Note that if the protocol $\mathcal{A}$ is not efficient, it is possible that there are rounds that give decomposition with $t = 1$. In such a case, we can omit these rounds.)

Continuing with this correspondence, it is easy to see that the number of communication rounds in the protocol is equal to the depth of the corresponding decom-

position tree. To derive our lower bounds we use this equality together with the next lemma, which claims that the optimal decomposition tree has the minimal depth.

LEMMA 4.1. *Let M be a decomposable matrix. Let T_{opt} be the optimal decomposition tree for M. For every T, a decomposition tree of M, the depth of T_{opt} is less than or equal to the depth of T.*

Proof. Given a nonoptimal decomposition tree T, we can use the following bottom-up process to get the optimal tree T_{opt} without increasing the depth: In each step, find an internal node v for which the decomposition is not optimal, but the decomposition in any node of its subtree is optimal. Changing the decomposition in the node v to the optimal one, does not increase the depth of the tree (and may even decrease it).

More formally, let v be an internal node of T as above and denote by $C_v \times D_v$ the matrix corresponding to v. Assume without loss of generality that in v we have rows decomposition. That is, C_v is decomposed to $C_1, C_2, \cdots, C_t$. Since the decomposition is not optimal, there exists a set C_i in the decomposition that can be replaced by C_i^1 and C_i^2 (i.e., $C_1, \cdots, C_{i-1}, C_i^1, C_i^2, C_{i+1}, \cdots, C_t$ are the sets of a row decomposition of $C_v \times D_v$). Let $C \times D$ be the matrix in any node of the subtree corresponding to $C_i \times D_v$; then in the subtree corresponding to $C_i^j \times D_v$ $(j = 1, 2)$ this matrix will be replaced by $(C \cap C_i^j) \times D$. In the case that this matrix is monochromatic, then its subtree will be omitted. It is easy to verify that the modified tree is indeed a decomposition tree, and clearly the depth is not increased. □

COROLLARY 4.2. *For every private function f, the protocol $\mathcal{A}_f$ (described in the proof of Theorem 3.3) achieves the optimal number of rounds.*

This is because every private protocol that computes f corresponds to a decomposition tree. According to Lemma 4.1, the optimal decomposition tree, which is exactly the tree corresponding to $\mathcal{A}_f$, has the minimal depth. Note that Lemma 4.1 implies that $\mathcal{A}_f$ is optimal in a very strong sense: the depth of *every* input (x, y) in the corresponding tree is minimal, and not only of the worst-case input.

COROLLARY 4.3. *For every $1 \leq g(n) \leq 2 \cdot (2^n - 1)$ there exists a function f that is privately computable using a $g(n)$-round protocol but is not privately computable using any $(g(n) - 1)$-round protocol.*

This corollary implies the existence of a rounds-complexity hierarchy for privately computable functions. To prove it, we present below a function that is privately computable using a $(2 \cdot (2^n - 1))$-round protocol, but is not privately computable using any $(2 \cdot (2^n - 1) - 1)$-round protocol. This example can be easily generalized to any $1 \leq g(n) \leq 2 \cdot (2^n - 1)$. We consider the following function f (see M_f in Fig. 2):

$$f(x, y) = \begin{cases} 2\text{x} & \text{if } x \leq y, \\ 2\text{y+1} & \text{if } x > y. \end{cases}$$

The optimal decomposition of M_f gives us the following protocol $\mathcal{A}_f$:

For $i = 0, 1, 2, \cdots$,

1. $P_{\mathcal{X}}$ compares its input x with i. If $x = i$ it sends $f(x, y) = 2 \cdot i$ and the protocol is terminated. Otherwise it sends a GO-ON message (one bit).
2. $P_{\mathcal{Y}}$ compares its input y with i. If $y = i$ it sends $f(x, y) = 2 \cdot i + 1$, and the protocol is terminated. Otherwise, it sends a GO-ON message (one bit).

It is easy to see that this protocol requires $2 \cdot (2^n - 1)$ rounds (and $\Theta(2^n)$ bits) for the input $(2^n - 1, 2^n - 1)$. Moreover, the average number of rounds (and bits) is also $\Theta(2^n)$. Note that in a regular computation every function f can be computed using at most $n + |f(x, y)|$ bits and two communication rounds (simply, $P_{\mathcal{X}}$ sends x

f	0	1	2	3	$\cdots$	2^n-2	2^n-1
0	0	0	0	0	$\cdots$	0	0
1	1	2	2	2	$\cdots$	2	2
2	1	3	4	4	$\cdots$	4	4
3	1	3	5	6	$\cdots$	6	6
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
2^n-2	1	3	5	7	$\cdots$	$2^{n+1}-4$	$2^{n+1}-4$
2^n-1	1	3	5	7	$\cdots$	$2^{n+1}-3$	$2^{n+1}-2$

FIG. 2.

and P_y computes and sends $f(x,y)$). Since in every round of the optimal protocol we decompose our matrix into at least two submatrices, then the function f defined above is the worst function. In other words, every private function f can be privately computable using at most $2\cdot(2^n-1)$ rounds.

COROLLARY 4.4. *Let m be the size of f's range (i.e., $f:\{0,1\}^n\times\{0,1\}^n\to\{0,1,\cdots,m-1\}$); then the rounds complexity of f is bounded above by $2\cdot(m-1)$.*

Here again we use the optimality of $\mathcal{A}_f$ to claim that in every round the current submatrix is decomposed into at least two submatrices. In addition, in a rows decomposition, for every row x all the y's such that $M_{x,y}$ has the same value belong to the same D_i in the decomposition. (A similar property holds for columns decomposition.) Thus, if the original matrix contains m different values, then (by a simple induction) we can show that after $2\cdot i$ rounds of the protocol, every row and every column of the submatrix $C\times D$ contains at most $m-i$ different values. Therefore the number of rounds is bounded by $2\cdot(m-1)$.

To give lower bounds on the communication complexity of privately computable functions, we present the following claim, which relates rounds complexity to communication complexity in private computations.

CLAIM 1. *Let f be a privately computable function. Denote its communication complexity by $C(f)$ and its rounds complexity by $R(f)$. Then the following holds:*

$$R(f)\le C(f)\le n\cdot R(f).$$

Proof. $R(f)\le C(f)$ since in any protocol the parties send at least one bit per round. On the other hand, for every privately computable function f, the bound $R(f)$ is achieved by the protocol $\mathcal{A}_f$. In any round of this protocol, the parties send a message with size at most n. (Note that in steps 2(a) and 2(b) of $\mathcal{A}_f$ we may save one bit per round by omitting the most significant bit which is always 1.) The claim follows. □

COROLLARY 4.5. *Randomization does not help to reduce the number of rounds needed in private computations.*

This is true since the protocol $\mathcal{A}_f$ is a deterministic protocol. We remark that $\mathcal{A}_f$, besides being optimal in its rounds complexity, is also optimal in its communication complexity among the deterministic protocols. To prove this, we observe that if $\mathcal{A}$ is a deterministic protocol then the number of different communication strings is equal to the number of leaves in the decomposition tree that corresponds to $\mathcal{A}$. We conclude not only that randomization does not help to compute more functions privately, but also that it does not help to reduce the number of rounds needed in private computations.

5. Discussion. As is pointed out in [8], the characterization of two-party private functions can be used to give necessary conditions for privacy in a multiparty scenario. That is, $f(x_1, x_2, \cdots, x_N)$ is t-private, for $t \geq \lfloor (N-1)/2 \rfloor + 1$, if for *any* partition of the N parties into two sets of participants T and $\bar{T}$, where T is of size t, the two-argument function $\hat{f}$ defined as

$$\hat{f}(\{x_i\}_{i \in T}, \{x_i\}_{i \in \bar{T}}) \stackrel{\text{def}}{=} f(x_1, x_2, \cdots, x_N)$$

is private.

Using the above observation and Theorem 3.1, we get from any such partition a necessary condition for f to be t-private ($t \geq \lfloor (N-1)/2 \rfloor + 1$). In addition, from any such partition and using Corollary 4.2, we get lower bounds on the rounds complexity and the communication complexity of computing f t-privately.

The problem of giving an exact characterization of the t-private (general) functions for $\lfloor (N-1)/2 \rfloor < t \leq N$ is still open.

Acknowledgments. The author thanks Benny Chor and Oded Goldreich for helpful discussions on the topics of this paper, and Avi Wigderson for asking the question of Corollary 4.4. The author also thanks Shai Ben-David, Hugo Krawczyk, Alon Lavie, Lior Moscovici, and Dan Simon for their useful comments on this paper.

REFERENCES

[1] N. Alon, P. Frankl, and V. Rodl, *Geometric realization of set systems and probabilistic communication complexity*, Proc. of 26th FOCS, Portland, OR, 1985, pp. 277–280.
[2] M. Ben-or, *Another advantage of free choice: complete asynchronous agreement protocols*, in Proc. of 2nd PODC, 1982, pp. 27–30.
[3] D. Beaver, *Perfect privacy for two-party protocols*, Tech. Report TR-11-89, Harvard University, Cambridge, MA, 1989.
[4] G. Bracha, *An $O(\log n)$ expected rounds randomized Byzantine generals protocol*, in Proc. of 17th STOC, Providence, RI, 1985, pp. 316–326.
[5] M. Ben-or, S. Goldwasser, and A. Wigderson, *Completeness theorems for non-cryptographic fault-tolerant distributed computation*, in Proc. of 20th STOC, Chicago, IL, 1988, pp. 1–10.
[6] D. Chaum, C. Crepeau, and I. Damgard, *Multiparty unconditionally secure protocols*, in Proc. of 20th STOC, Chicago, IL, 1988, pp. 11–19.
[7] B. Chor and O. Goldreich, *Unbiased bits from sources of weak randomness and probabilistic communication complexity*, SIAM J. Comput., 17 (1988), pp. 230–261.
[8] B. Chor and E. Kushilevitz, *A zero-one law for Boolean privacy*, SIAM J. Discrete Math., 4 (1991), pp. 36–47.
[9] P. Duris, Z. Galil, and G. Schnitger, *Lower bounds on communication complexity*, in Proc. of 16th STOC, Washington, DC, 1984, pp. 81–91.
[10] P. Feldman and S. Micali, *Optimal algorithms for Byzantine agreement*, in Proc. of 20th STOC, Chicago, IL, 1988, pp. 148–161.
[11] O. Goldreich, S. Micali, and A. Wigderson, *How to play any mental game*, in Proc. of 19th STOC, New York, NY, 1987, pp. 218–229.
[12] A. El-Gamal and A. Orlitsky, *Randomized communication complexity*, preprint, 1985.
[13] D. Lehman and M. O. Rabin, *On the advantage of free choice: A symmetric and fully distributed solution to the dining philosophers problem*, in Proc. of 8th POPL, 1981, pp. 133–138.
[14] K. Mehlhorn and E. Schmidt, *Las Vegas is better than determinism in* VLSI *and distributed computing*, in Proc. of 14th STOC, San Francisco, CA, 1982, pp. 330–337.
[15] C. Papadimitriou and M. Sipser, *Communication complexity*, J. Comput. System. Sci., 28 (1984), pp. 260–269.
[16] A. C. Yao, *Some complexity questions related to distributed computing*, in Proc. of 11th STOC, Atlanta, GA, 1979, pp. 209–213.
[17] ———, *Protocols for secure computations*, in Proc. of 23th FOCS, 1982, pp. 160–164.
[18] ———, *Lower bounds by probabilistic arguments*, in Proc. of 24th FOCS, 1983, pp. 420–428.

[19] A. C. YAO, *How to generate and exchange secrets*, in Proc. of 27th FOCS, Toronto, Ontario, 1986, pp. 162–167.

SIAM J. DISC. MATH.
Vol. 5, No. 2, pp. 285–304, May 1992
© 1992 Society for Industrial and Applied Mathematics
010

AN OPTIMAL ALGORITHM FOR THE MAXIMUM TWO-CHAIN PROBLEM*

R. D. LOU†, M. SARRAFZADEH†, AND D. T. LEE†

Abstract. Given a point set ρ, a *chain* is a subset $\mathcal{C} \subseteq \rho$ of points in which, for any two points, one is dominated by the other. A *two-chain* is a subset of ρ that can be partitioned into two chains. A two-chain with maximum cardinality among all possible two-chains is called a *maximum two-chain.* This paper presents a $\Theta(n \log n)$ time and $\Theta(n)$ space algorithm for finding a maximum two-chain in a point set ρ, where $n = |\rho|$. Maximum two-chain has applications in, for example, graph-theoretic problems, VLSI layout, and sequence manipulation.

Key words. point dominance, dominance hull, max two-chain, optimal algorithm, lower bound

AMS(MOS) subject classifications. 68Q25, 68U05, 68P05, 05C10

1. Introduction.

Consider a point set $\rho = \{P_1, P_2, \cdots, P_n\}$ on the x-y plane, where $P_i = (x_i, y_i)$ with $x_i > 0$ and $y_i > 0$; by convention, $x_i > x_j$ for $i > j$ and all the y's are distinct. Following [PS], we say P_i *dominates* P_j if $x_i > x_j$ and $y_i > y_j$. Points P_i and P_j are *crossing* if $x_i > x_j$ and $y_j > y_i$, or vice versa; otherwise, they are *noncrossing.* A subset of ρ is called a *chain* if its points are pairwise noncrossing. A subset is called a *two-chain* if it contains no three points that are pairwise crossing. A *maximum chain* is a chain with the maximum number of points among all chains. A *maximum two-chain* is a two-chain with the maximum number of points among all two-chains. The algorithm for finding a maximum two-chain can be used to solve the following problems.

Graph-theoretic problem. Given a permutation graph model, find a maximum bipartite subgraph. The problem of finding a maximum bipartite subgraph in an overlap graph (i.e., circle graph) is generally NP-hard [SL1]. However, for problems with a fixed "partition number," a polynomial-time algorithm is proposed in [SL2].

VLSI *layout.* This is the two-layer topological via minimization problem in a restricted two-sided routing region (called the *two-sided-channel* TVM *problem*) [SL1], [SL2].

Sequence manipulation. Given a sequence of numbers, find a maximum two-increasing subsequence.

The maximum one-chain problem can be solved by an algorithm proposed in [AK] in $\Theta(n \log n)$ time. For the maximum two-chain problem, it is easy to find an example for which the "greedy method" (e.g., applying the algorithm for the maximum one-chain problem twice, where the chain obtained in the first application is removed from the original set before the second application of the algorithm) will not produce a maximum two-chain [SL1]. Previously, an $O(n^2 \log n)$ time algorithm to solve the maximum two-chain problem was proposed [SL1]. Recently, an $O(kn^2)$ time algorithm to find a maximum-weighted k-chain was proposed [SLo]. In this paper, we present a $\Theta(n \log n)$ time algorithm for finding a maximum two-chain of a point set ρ, where $n = |\rho|$ (an $\Omega(n \log n)$ lower bound is also established).

* Received by the editors June 28, 1989; accepted for publication (in revised form) April 30, 1991. This work was supported in part by National Science Foundation grants DCR-8420814, CCR-8901815, MIP-8709074, and MIP-8921540. A preliminary version of this work appeared in Proceedings of the First ACM-SIAM Conference on Discrete Algorithms, San Francisco, January 1990.

† Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois 60208.

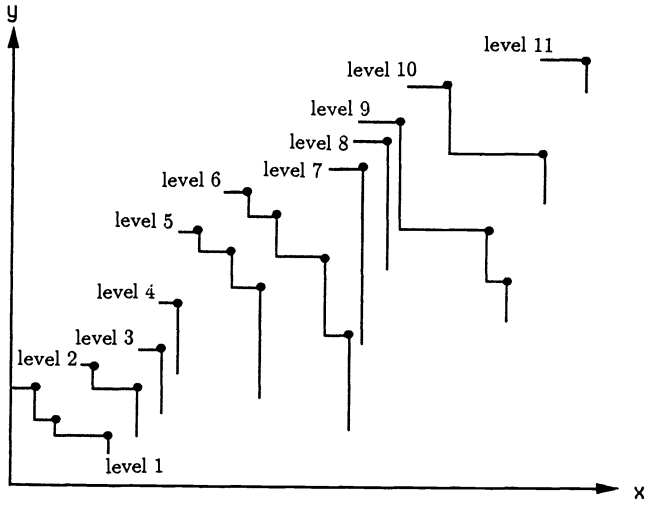


FIG. 1. *To assign each point a level.*

This paper is organized as follows. In §2 the basic idea of the algorithm is outlined. In §3 details of the algorithm are given, and its time complexity is analyzed.

2. Basic idea of the algorithm. In this section, we first define some terminologies that simplify the description of the algorithm.

Consider a chain $C = \{P_{\sigma_1}, P_{\sigma_2}, \cdots, P_{\sigma_{|C|}}\}$, where, by convention, $x_{\sigma_i} < x_{\sigma_j}$ for $i < j$. Point P_{σ_i} is called the ith *point* of the chain. The first point and the last point are also called the *bottom* and the *top* of the chain, respectively. Given a set of points ρ, the points that are not dominated by any other points are called the *maxima* of ρ. We can find the maxima of ρ : all maxima are at the same "*level*," known as the *dominance hull* [PS] of ρ. Then ignore the points at this level and recursively find other points at the same level for the remaining points. This process is repeated until all points in ρ have been assigned a level (Fig. 1). We call the levels, from bottom to top, *levels* $1, 2, \cdots, s$, where level s contains points on the dominance hull of ρ. We denote the set of points at levels i by ρ_i. We also assign each point P_a a value $L(a)$, called the *L-value* of point P_a, to indicate that P_a is at level $L(a)$. The set of points in $\cup_{i=u}^{v}\rho_i$, where $u \leq v$, is denoted by $\rho_{u,v}$. By convention, $\rho_{u,v}=\phi$, for $u > v$.

The levels have a property stated in Lemma 2.1.

LEMMA 2.1. *Each point at level u, where $1 \leq u < s$, is dominated by at least one point at level $u+1$.*

Proof. Assume that point P_a is at level u. If P_a is not dominated by any point at level $u+1$, then point P_a is a maximum of $\rho_{1,u+1}$. According to the definition of a level, P_a should be at level $u+1$. This is a contradiction. □

The point with the smallest (or largest) x-coordinate at a level is called the *left-* (or *right-*) *point* of the level. The point at level $u+1$ with the smallest (or largest) x-coordinate among all points dominating P_a at level u is called the *left-dominating* (or *right-dominating*) *point* of P_a. Let the left-point of level 1 be the first point of a chain, and let the left-dominating point of the ith point of the chain be the $(i+1)$th point, for $1 \leq i < s$. Symmetrically, let the right-point of level 1 be the

first point of another chain and the right-dominating point of the ith point of the chain be the $(i+1)$th point, for $1 \leq i < s$. A chain with one point at each level is called a *solid chain.* Obviously, each of the above two chains is a solid chain. If these two chains have no common point(s), then they form a maximum two-chain containing $2s$ points. If these two chains have some common point(s), then the union of these two chains may not be a maximum two-chain. For example, consider points $P_1 = (1,3)$, $P_2 = (2,1)$, $P_3 = (3,4)$, $P_4 = (4,5)$, and $P_5 = (5,2)$. The just-described technique selects $\{P_1, P_2, P_3, P_4\}$ as a solution. However, the entire point set is a two-chain. (One "wrong" point, in this case, P_3, can guide us to select a suboptimal two-chain.) Thus we need a more involved algorithm, described below, to find a maximum two-chain. However, the above two chains will be used as a "skeleton" of the final maximum two-chain.

We assume that there is a fictitious level 0 containing one fictitious point $P_0 = (0,0)$. Clearly, P_0 is dominated by all points in ρ. Then we process the points from left to right, starting from level 0, level by level. For each level u, we perform the following four phases:

(1) *local chain assignment phase,*
(2) *N-value assignment phase,*
(3) *candidate chain assignment phase,* and
(4) *adjustment phase.*

At the end, we construct chains $\mathcal{T}(0), \mathcal{B}_{\mathcal{L}}(1), \mathcal{B}_{\mathcal{R}}(1), \mathcal{T}(1), \cdots, \mathcal{B}_{\mathcal{L}}(k), \mathcal{B}_{\mathcal{R}}(k)$, and $\mathcal{T}(k)$, where $k \leq s$, each of which is a subset of ρ: we refer to these chains as *local chains.* Chains $\mathcal{B}_{\mathcal{L}}(i)$, $\mathcal{B}_{\mathcal{R}}(i)$, and $\mathcal{T}(i)$ are called *left branch, right branch,* and *tie* of the ith local chains, respectively. (Changes in the index i indicate that a new branch pair has been reached.) A point in a local chain is called a *marked point*; otherwise, a point is *unmarked.* Initially, all local chains are empty. At local chain assignment phase of level 0, we assign P_0 to $\mathcal{T}(0)$. At local chain assignment phase of level u, where $u > 0$, we assign points at level u to local chains depending on the assignment of points to local chains at level $u-1$ (see Fig. 2). There are two cases, as follows:

(LC1) Point P_a at level $u-1$ has been assigned to local chain $\mathcal{T}(i)$: If the left-dominating point and the right-dominating point of P_a are distinct, we assign them to $\mathcal{B}_{\mathcal{L}}(i+1)$ and $\mathcal{B}_{\mathcal{R}}(i+1)$, respectively. Otherwise, we assign the point dominating P_a to $\mathcal{T}(i)$.

(LC2) Points P_a and P_b at level $u-1$ have been assigned to local chains $\mathcal{B}_{\mathcal{L}}(i)$ and $\mathcal{B}_{\mathcal{R}}(i)$, respectively: If the left-dominating point of P_a and the right-dominating point of P_b are distinct, then we assign them to $\mathcal{B}_{\mathcal{L}}(i)$ and $\mathcal{B}_{\mathcal{R}}(i)$, respectively; otherwise, we assign the point dominating both P_a and P_b to $\mathcal{T}(i)$.

At N-value assignment phase of level u, we assign a value $N(a)$ to each point P_a at level u, called *N-value* of P_a. Intuitively, $N(a)$ is the size of a maximum two-chain in $\rho_{1,L(a)}$ *containing* P_a (obviously, fictitious point P_0 does not contribute to any two-chains). Such a maximum two-chain is denoted by P_a-max-2-*chain.* We assign $N(a)$ to P_a according to the following rules (see Fig. 3):

(R1) $N(0) = 0$ for P_0;
(R2) If point P_a is in a tie, then $N(a)$ equals $N(b)+1$, where P_b is any marked point at level $L(a)-1$;
(R3) If point P_a is in a branch, then $N(a)$ equals $N(b)+2$, where P_b is any marked point at level $L(a)-1$;
(R4) If point P_a is unmarked, then $N(a)$ equals the maximal value of $[N(b)+$

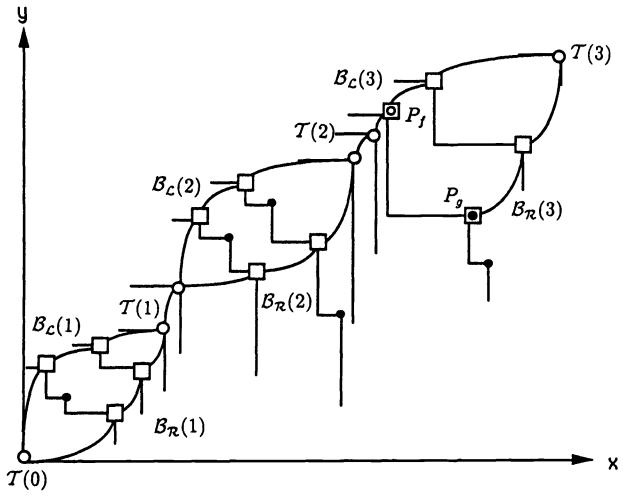


FIG. 2. *Local chains.* •: *unmarked point*; ○: *marked point assigned to a tie at local chain assignment phase*; □ : *marked point assigned to a branch at local chain assignment phase*; ◙: *marked point assigned to a branch at adjustment phase*; ◎: *marked point assigned to a tie at local chain assignment phase and then removed to a branch at adjustment phase*; / : *local chains.*

$L(a) - L(b) + 1]$ over all points P_b dominated by P_a. One of the points, chosen arbitrarily, whose N- and L-values make $[N(b) + L(a) - L(b) + 1]$ maximum, is called the *preceding point* of P_a.

Note that marked points at the same level have the same N-value.

At candidate chain assignment phase of level u, for each point P_a at level u, we construct a chain $C(a)$ with top P_a, called *candidate chain* of P_a, in the following manner (see Fig. 3). If P_a is marked, then $C(a)=\{P_a\}$. Otherwise, $C(a) = C(b) \cup \{P_a\}$, where P_b is the preceding point of P_a. Note that the bottom of a candidate chain must necessarily be a marked point.

At adjustment phase of level u, we examine all points at that level to adjust local chains and N-values of marked points. If there exists a point P_f at level u belonging to a tie $\mathcal{T}(i)$, and there also exists some unmarked point at level u, say P_g, whose N-value is the largest among all unmarked points at level u and is no less than $N(f)$, then we delete P_f from $\mathcal{T}(i)$ and set $N(f)$ equal to $N(g)$ (e.g., P_f and P_g in Fig. 3). If $x_g > x_f$, then we assign P_g to $\mathcal{B}_{\mathcal{R}}(i+1)$ and P_f to $\mathcal{B}_{\mathcal{L}}(i+1)$ (e.g., P_f and P_g in Fig. 2). Otherwise, we assign P_g to $\mathcal{B}_{\mathcal{L}}(i+1)$ and P_f to $\mathcal{B}_{\mathcal{R}}(i+1)$. Note that P_g is now a marked point but may have a candidate chain containing more than one point.

Local chains and candidate chains have properties stated in the following lemmas. Lemma 2.2 follows directly from local chain assignment rules (LC1), (LC2) and adjustment phase procedure.

LEMMA 2.2. *At each level, there are exactly two points in branches, one in a left branch and the other in a right branch, or one point in a tie.*

LEMMA 2.3. *There exists a solid chain from each marked point at level u to a marked point at level v, $u < v$.*

Proof. According to Lemma 2.1, at local chain assignment phase and adjustment phase we can find for any point P_a in $\mathcal{B}_{\mathcal{L}}(i)$, $\mathcal{B}_{\mathcal{R}}(i)$, or $\mathcal{T}(i)$ at level u, at least one

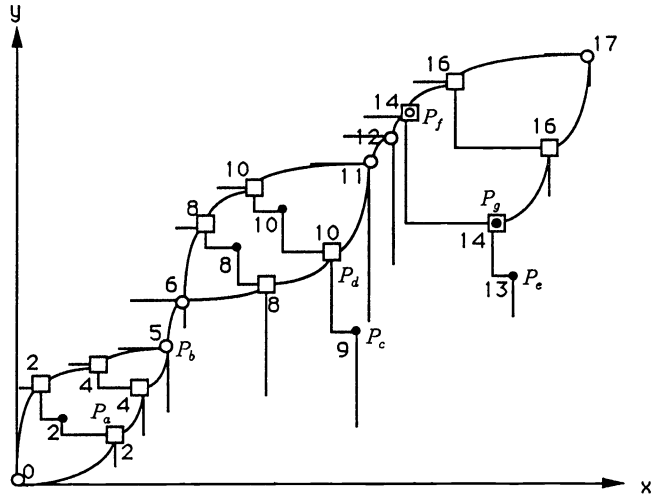


FIG. 3. *N-value of each point and candidate chains of some points in Fig.* 1: $C(a) = \{P_a\}$, $C(b) = \{P_b\}$, $C(c) = \{P_b, P_c\}$, $C(d) = \{P_d\}$, $C(e) = \{P_b, P_c, P_e\}$, $C(g) = \{P_d, P_g\}$.

marked point at level $u + 1$ dominating P_a. Thus, inductively, we can find a solid chain, in which every point is marked (i.e., belonging to a branch or a tie), from P_a to a marked point at level v, where $u < v$. Note that, if there exist two marked points at level v, the solid chain may reach only one of them. In particular, there exists a solid chain from any marked point P_a at level $u < s$ to a marked point at level s. The claim follows. □

LEMMA 2.4. *Each marked point has the largest N-value among all points at the same level.*

Proof. Let us consider the following cases first.

(1) If P_p is the first point of a branch obtained at local chain assignment phase (from local chain assignment rule (LC1)), and P_q is the marked point (in a tie) at level $L(p) - 1$, then $N(p) = N(q) + 2 = N(q) + L(p) - L(q) + 1$.

(2) If two points P_p and P_q are in the same branch and $L(p) > L(q)$ (and hence P_p is assigned to the branch at local chain assignment phase), then, from N-value assignment rule (repeated applications of rule (R3)), $N(p) = N(q) + 2(L(p) - L(q)) \geq N(q) + L(p) - L(q) + 1$.

In other words, if P_p is a point assigned to a branch at local chain assignment phase, if P_q is a marked point, and if $L(p) > L(q)$, then $N(p) \geq N(q) + L(p) - L(q) + 1$.

We now prove the lemma by induction on the *level* number. It is clear that the lemma is true for level 0. For level u, we assume that the lemma is true for any level v, where $v < u$. Let P_a and P_b be a marked and an unmarked point at level u, respectively. We distinguish two cases.

Case 1. P_a is assigned to a branch at local chain assignment phase (see Fig. 4): Let P_f be the preceding point of point P_b, and P_g be a marked point at level $L(f)$ (if P_f is marked, then let P_g be P_f). P_g may or may not be dominated by P_a. By inductive hypothesis, $N(g) \geq N(f)$. From the N-value assignment rule (R4), $N(b) = N(f) + L(a) - L(g) + 1$ and, from the previous discussion, $N(a) \geq N(g) + L(a) - L(g) + 1$.

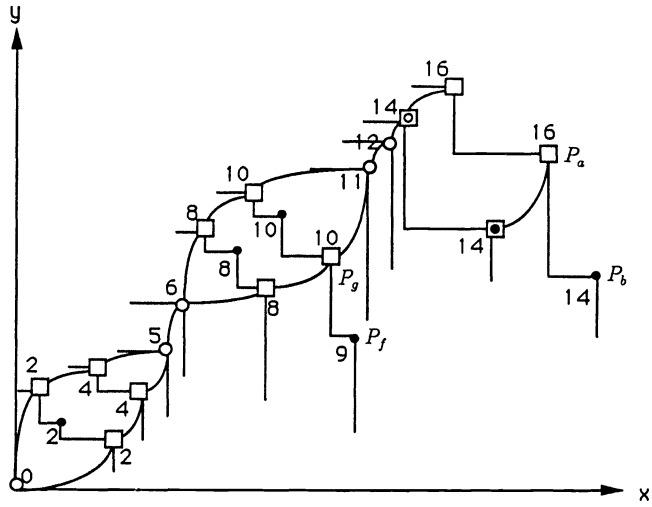


FIG. 4. *Proof of Lemma* 2.4.

Therefore $N(a) \geq N(b)$.

Case 2. P_a is a point in a tie or is assigned to a branch at adjustment phase: Because of the comparison of N-values at adjustment phase, it is clear that the lemma is true in this case.

Hence the lemma is established. □

In the following, we show how to construct a two-chain $\mathcal{M}(a)$ associated with a point P_a ($\mathcal{M}'(a)=\mathcal{M}(a) - \{P_0\}$ being a P_a-max-2-chain; see Lemma 2.5 below).

(M1) For P_0, $\mathcal{M}(0) = \{P_0\}$.

(M2) To construct $\mathcal{M}(a)$ of P_a, where $L(a) > 0$, we assume that $\mathcal{M}(g)$ of each point P_g at level $L(g)$, where $L(g) < L(a)$, has been constructed. We have the following cases.

(M2.1) P_a is an unmarked point (see Fig. 5(a)): Let the marked point(s) at level $L(a)$ be P_b (and $P_{b'}$), P_d the *preceding point* of P_a at level $L(d) < L(a)$, and P_e a marked point at level $L(e) = L(d)$. According to Lemma 2.3, there exists a solid chain from P_e to P_b (or from P_e to $P_{b'}$). We assign to $\mathcal{M}(a)$ the set $\mathcal{M}(d) \cup P_a \cup \mathcal{C}$, where $\mathcal{C}$ denotes a solid chain from P_e to P_b (or $P_{b'}$).

(M2.2) P_a is a marked point in a tie or a branch in the following cases:

(1) If $P_a \in \mathcal{B}_{\mathcal{R}}(i)$ and $P_b \in \mathcal{B}_{\mathcal{L}}(i)$ are the marked points at level $u = L(a) = L(b)$, and one of them, say P_a, has a preceding point P_d at level $L(d) < u$ (see Fig. 5(b)), then we assign to $\mathcal{M}(a)$ the set $\mathcal{M}(d) \cup P_a\} \cup \mathcal{C}$, where $\mathcal{C}$ denotes a solid chain from a marked point P_e at level $L(e) = L(d)$ to point P_b.

(2) If $P_a \in \mathcal{B}_{\mathcal{R}}(i)$ and $P_b \in \mathcal{B}_{\mathcal{L}}(i)$ are the marked points at level $u = L(a) = L(b)$, and both of them have no preceding point (see Fig. 5(c)), we assign to $\mathcal{M}(a)$ the set $\mathcal{M}(d) \cup \{P_a, P_b\}$, where P_d denotes a marked point at level $L(d) = L(a) - 1$.

(3) If P_a is in a tie, let P_d be a marked point at level $L(d) = L(a) - 1$

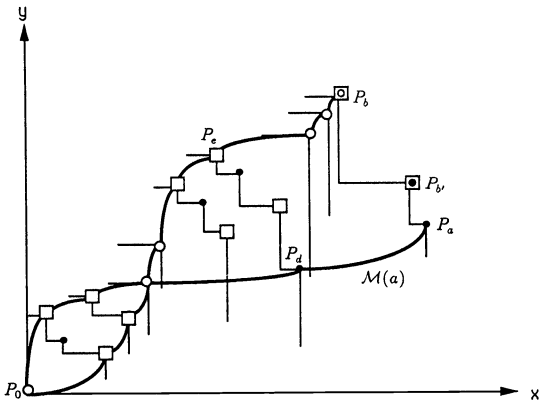


FIG. 5(a). $\mathcal{M}(a)$ *of an unmarked point* P_a.

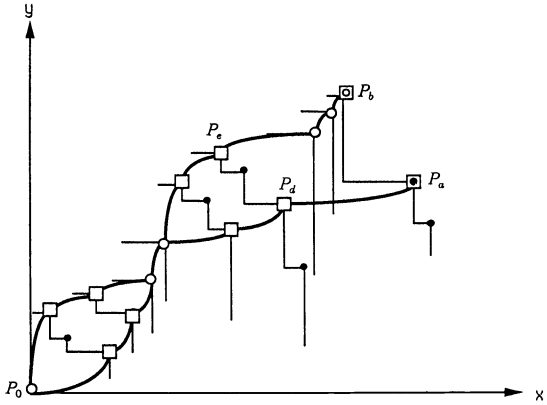


FIG. 5(b). $\mathcal{M}(a)$ *of a point* P_a *in a branch.*

(see Fig. 5(d)). We assign to $\mathcal{M}(a)$ the set $\mathcal{M}(d) \cup \{P_a\}$.

LEMMA 2.5. *For each point* P_a, $\mathcal{M}'(a)$ *is a* P_a-max-2-*chain containing* $N(a)$ *points.*

Proof. Because P_0 is a fictitious point, clearly P_0-max-2-chain is an empty set.

To establish the result for P_a, $a > 0$, we assume that the lemma is true for all points P_g with $L(g) < L(a)$.

Case 1. P_a is unmarked: We first show that $\mathcal{M}'(a)$ is a two-chain containing $N(a)$ points. It is obvious that $\mathcal{M}'(a)$ is a two-chain (since it was obtained by adding a chain to the top point of each chain of a two-chain). The total number M of points contained in $\mathcal{M}(a)$ is the sum of $N(d) + 1$, 1, and $L(a) - L(d)$, which correspond to the sizes of sets $\mathcal{M}(d)$, $\{P_a\}$, and solid chain from P_e to P_b (or $P_{b'}$), respectively (cf. (M2.1)). Since P_d is the preceding point of P_a, $N(a) = N(d) + L(a) - L(d) + 1$. Therefore we have that $M = N(a) + 1$.

Second, we show that no two-chain containing point P_a in $\rho_{1,L(a)}$ has size larger than $N(a)$. Consider any two-chain in $\rho_{1,L(a)}$ containing point P_a with size $N'(a)$ (see Fig. 6(a)). Assume that this two-chain is constructed from chain $\mathcal{A}$ and chain $\mathcal{B}$, where the top of chain $\mathcal{A}$ is point P_a. Let the top of chain $\mathcal{A} - \{P_a\}$ be P_f. From inductive hypothesis, $N(f)$ is no less than the total size of the portions of chains $\mathcal{A}$ and $\mathcal{B}$ in $\rho_{1,L(f)}$. Because $\mathcal{A} \cap \rho_{L(f)+1,L(a)} = \{P_a\}$ and the size of $B \cap \rho_{L(f)+1,L(a)}$ is

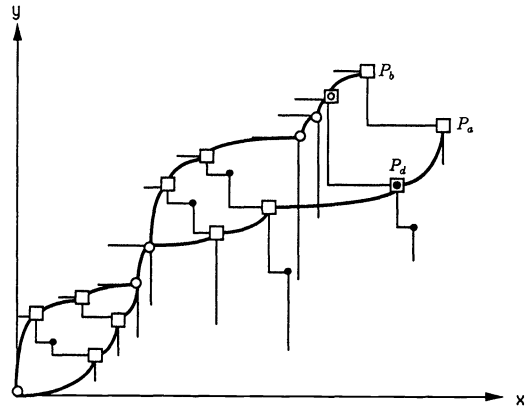


FIG. 5(c). $\mathcal{M}(a)$ *of a point* P_a *in a branch.*

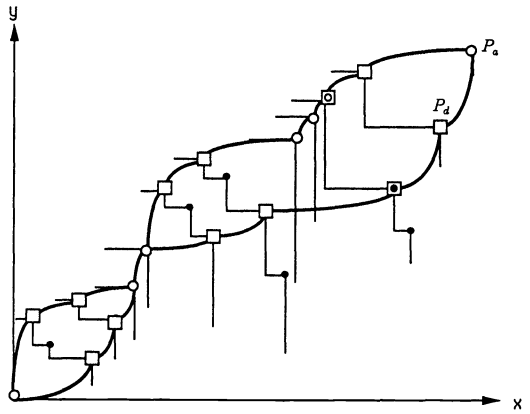


FIG. 5(d). $\mathcal{M}(a)$ *of a point* P_a *in a tie.*

no larger than the size of a solid chain in $\rho_{L(f)+1,L(a)}$, we have that

$$\begin{aligned}
&|(\mathcal{A}\cup\mathcal{B})\cap\rho_{1,L(f)}| \\
&\quad = N'(a) - |(\mathcal{A}\cup\mathcal{B})\cap\rho_{L(f)+1,L(a)}| \\
&\quad \geq N'(a) - |\mathcal{A}\cap\rho_{L(f)+1,L(a)}| - |\mathcal{B}\cap\rho_{L(f)+1,L(a)}| \\
&\quad = N'(a) - |\mathcal{B}\cap\rho_{L(f)+1,L(a)}| - 1 \\
&\quad \geq N'(a) - (L(a) - L(f)) - 1.
\end{aligned}$$

Hence $N(f) \geq N'(a) - (L(a) - L(f)) - 1$ and $N(f) + L(a) - L(f) + 1 \geq N'(a)$.

Because P_a is an unmarked point, if P_f is the preceding point of P_a, it holds that

$$N(a) = N(f) + L(a) - L(f) + 1 \geq N'(a);$$

otherwise,

$$N(a) \geq N(f) + L(a) - L(f) + 1 \geq N'(a).$$

Thus $N(a) \geq N'(a)$.

Case 2. P_a is marked: It is easy to show that $\mathcal{M}'(a)$ is a two-chain and contains $N(a)$ points. We show below that no two-chain containing point P_a in $\rho_{1,L(a)}$ has size larger than $N(a)$. Consider any two-chain in $\rho_{1,L(a)}$ containing P_a with size $N'(a)$

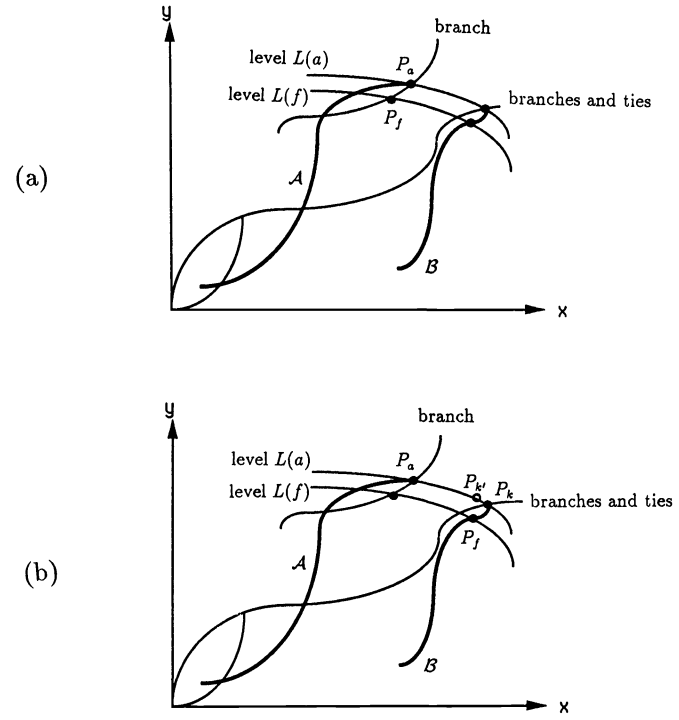


FIG. 6. *Proof of Lemma* 2.5.

(see Fig. 6(b)). Assume that this two-chain is constructed from chain $\mathcal{A}$ and chain $\mathcal{B}$, where the top of chain $\mathcal{A}$ is point P_a. Let the top of chain $\mathcal{B}$ be point P_k and let the top of chain $\mathcal{B} - \{P_k\}$ be P_f. As in Case 1,

$$N(f) + L(a) - L(f) + 1 \geq N'(a).$$

We have the following two cases.

Case 2.1. P_k is an unmarked point : If P_f is the preceding point of P_k, then

$$N(k) = N(f) + L(a) - L(f) + 1 \geq N'(a);$$

otherwise,

$$N(k) \geq N(f) + L(a) - L(f) + 1 \geq N'(a).$$

Thus $N(a) \geq N(k) \geq N'(a)$, where the first inequality follows from Lemma 2.4.

Case 2.2. P_k is a marked point (see Fig. 6(b)): We temporarily introduce a fictitious unmarked point $P_{k'} = (x_{k'}, y_{k'})$ at level $L(a)$ with $x_{k'} = x_k - \epsilon$ and $y_{k'} = y_k + \epsilon$ (or $x_{k'} = x_k + \epsilon$ and $y_{k'} = y_k - \epsilon$), where ϵ is a real number small enough to make $P_{k'}$ dominate P_f. Note that the fictitious point has no effect on N-values of the points having been processed. Then $N(a) = N(k) \geq N(k') \geq N(f) + L(a) - L(f) + 1 \geq N'(a)$.

Hence the size of P_a-max-2-chain is $N(a)$. □

Finally, we can obtain a maximum two-chain in $\rho_{1,s}$. The following theorem is readily established from Lemmas 2.4 and 2.5.

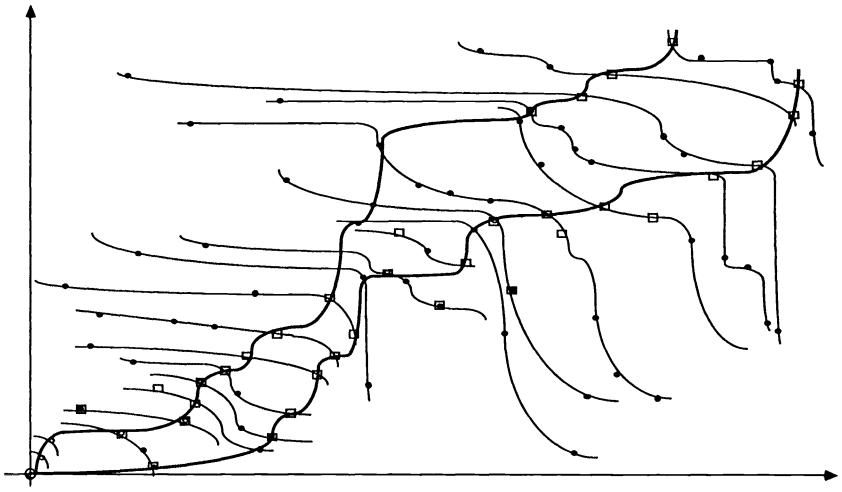


FIG. 7. *An example.* _: *levels*; ▬: *a maximum two-chain*; ◯: P_0; •: *unmarked point*; □ : *marked point assigned to a branch at local chain assignment phase*; ○: *marked point assigned to a tie at local chain assignment phase*; ▣: *marked point assigned to a branch at adjustment phase*; ⧈: *marked point assigned to a tie at local chain assignment phase and then remove to a branch at adjustment phase.*

THEOREM 2.1. *If P_a is a marked point at level s, then P_a-*max-2*-chain is a maximum two-chain in ρ.*

$\mathcal{M}(a) - \{P_0\}$ in Fig. 5(d) is a maximum two-chain of the point set in Fig. 1. Another example is shown in Fig. 7.

3. Details of implementation. In this section, we discuss detailed implementation of the algorithm in a bottom-up manner. We first discuss implementations of several algorithms and then construct the algorithm of finding a maximum two-chain from these algorithms.

3.1. Algorithm LEVEL. First, we introduce algorithm LEVEL to assign each point in ρ to a level. Recall that points in ρ are assumed to be sorted in ascending order of x-coordinates such that $x_1 < x_2 < \cdots < x_n$. We use a plane sweep technique [PS] to scan the points from right to left and perform the following operations. Suppose that we have processed points $P_n, P_{n-1}, \cdots, P_{i+1}$ and obtained l lists of points that belong to the same levels. For each list, we keep the y-coordinate of the last scanned point. These y-values are maintained as a height-balanced tree T. In processing P_i, we use T to first identify the list $\mathcal{L}(k)$ whose associated y-value is immediately below y_i. We then insert P_i to list $\mathcal{L}(k)$ and replace the associated y-value by y_i. If y_i is smaller than all y-values in T, the $(l+1)$th list is created, and its associated y-value is set to be y_i. At the end of the process, the number s of the lists created is the total number of levels. We then reindex the lists so that the innermost level (created last) has index 1 and the outermost one has index s. A formal description of algorithm LEVEL is as follows.

Procedure LEVEL(ρ)
begin
(* Assume that points are sorted so that $x_i < x_{i+1}, i = 0, 1, \cdots, n-1$, and $x_0 = 0$ *)
(* T := the height-balanced tree containing y-coordinates of some scanned points in ρ *)
(* $\mathcal{L}(j)$:= a list of the points that belong to the same level and maintained as a stack *)
(* $find(y_i, T)$ returns the list $\mathcal{L}(k)$ whose top element contains a point P_j such that y_j is the largest in T among all y-values smaller than y_i *)
(* $push(P_i, \mathcal{L}(k))$ inserts point P_i on the top of the list $\mathcal{L}(k)$ *)
(* $replace(y_i, \mathcal{L}(k), T)$ replaces the y-value associated with $\mathcal{L}(k)$ in tree T by y_i *)
(* $insert(y_i, \mathcal{L}(k), T)$ inserts y_i, which is associated with $\mathcal{L}(k)$ to tree T *)
$l := 1$; $\mathcal{L}(1) := \{P_n\}$; $\mathcal{L}(0) := \{P_0\}$; $T = \{P_0, P_n\}$; (* initialization *)
for $i = n - 1$ **downto** 1 **do**
begin
$\mathcal{L}(k) := find(y_i, T)$;
if $(k = 0)$
then begin
$l := l + 1; k := l$;
$push(P_i, \mathcal{L}(k))$;
$insert(y_i, \mathcal{L}(k), T)$;
end
else begin
$push(P_i, \mathcal{L}(k))$;
$replace(y_i, \mathcal{L}(k), T)$;
end
for i= 1 **to** l **do**
$\rho_i := \mathcal{L}(l + 1 - i)$;
end.

3.2. Algorithm LOCAL-CHAIN. We now describe the local chain assignment phase. Let level i, $i \geq 1$, be the first level with $|\rho_i| > 1$. All the points at level $j < i$ are assigned to $\mathcal{T}(0)$. Let the left-point in ρ_i be assigned to $\mathcal{B}_{\mathcal{L}}(1)$ and the right-point in ρ_i be assigned to $\mathcal{B}_{\mathcal{R}}(1)$. Assume now that all points at level 1 through $k - 1$ have been processed. We begin to process the points at level k by invoking the following procedure.

Procedure **LOCAL-CHAIN(k)**
begin
(* Assume that $k > 1$ and let $P_{\pi_m}, P_{\pi_{m+1}}, \cdots, P_{\pi_l}$ be the points in ρ_k *)
(* Suppose that the marked points at level $k - 1$ are P_a and P_b *)
(* If *chain-type*$(k-1)$= branch, i.e., $P_a \neq P_b$, P_a is assigned to $\mathcal{B}_{\mathcal{L}}(j)$ and P_b is assigned to $\mathcal{B}_{\mathcal{R}}(j)$ *)
(* If *chain-type*$(k - 1)$= tie, i.e., $P_a = P_b$, P_a is assigned to $\mathcal{T}(j)$ *)

$i := m$;
while $i \leq l$ and P_{π_i} does not dominate P_a **do** $i := i + 1$;
(* There is at least one point dominating P_a *)
$u := i$; (* Record the index of the left-dominating point of P_a *)
$i := i + 1$;
second := false;
while (not *second*) and $i \leq l$ **do**
  if P_{π_i} dominates P_b
  then *second* := true;
  else $i := i+1$;
if (not *second*)
then begin
    Assign P_{π_u} to $\mathcal{T}(j)$;
    chain-type(k) := tie;
  end
else begin
    while $i \leq l$ and P_{π_i} dominates P_b **do** $i := i + 1$;
    if *chain-type*($k-1$)= tie **then** $j := j + 1$;
    Assign P_{π_u} to $\mathcal{B}_{\mathcal{L}}(j)$ and assign $P_{\pi_{i-1}}$ to $\mathcal{B}_{\mathcal{R}}(j)$;
    chain-type(k) := branch;
  end
end.

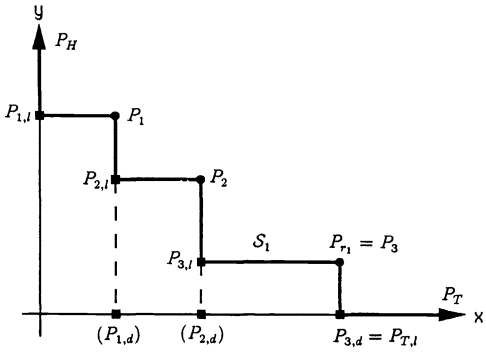
3.3. Algorithm N-ASSIGNMENT. Algorithm N-ASSIGNMENT(k) assigns a value $N(a)$ to each point P_a at a specified level k. If P_a is a marked point, it is trivial to assign $N(a)$ according to assignment rules (R1)–(R3). If P_a is unmarked, then, according to (R4), it is necessary to find a point P_b dominated by P_a and with maximum $[N(b) - L(b) + 1]$. Rule (R4) can be performed in an obvious way: for each point P_a we exhaustively examine all points dominated by P_a and find a point P_b with maximum $[N(b) - L(b) + 1]$. This would take $O(n)$ time, however, resulting in an $O(n^2)$ time algorithm. We show below that with a careful manipulation of the value $W(b) = N(b) - L(b)$, referred to as the *weight* of point P_b, we can find for each point P_a the desired point in $O(\log n)$ time. Toward this end, we introduce the notion of *staircases* associated with the points in ρ.

3.3.1. Staircases. To simplify the discussion, we *assume* that $|\rho_k| = r_k$, $k = 1, 2, \cdots, s$, and the points in ρ are reindexed as

$$\rho_1 = \{P_1, P_2, \cdots, P_{r_1}\}, \quad \rho_2 = \{P_{r_1+1}, \cdots, P_{r_1+r_2}\}, \cdots,$$
$$\rho_s = \{P_{r_1+\cdots+r_{s-1}+1}, \cdots, P_{r_1+\cdots+r_s}\}$$

so that $x_1 < x_2 < \cdots < x_{r_1}$, $x_{r_1+1} < x_{r_1+2} < \cdots < x_{r_1+r_2}$, and so forth. Consider the points in ρ_1. Let $P_{i,d} = (x_i, 0)$ be the *downward vertical intersection* (d-intersection for short) of P_i, $i = 1, 2, \cdots, r_1$. Let $P_{1,l} = (0, y_1)$ and $P_{i,l} = (x_{i-1}, y_i)$ be the *leftward horizontal intersection* (l-intersection for short) of the left-point P_1 and other points P_i in ρ, respectively, on the y-axis and the vertical line segment $\overline{P_{i-1}, P_{i-1,d}}$, $i = 2, 3, \cdots, r_1$ (see Fig. 8(a)). The rectilinear path $(P_{1,l}P_1P_{2,l}P_2 \cdots P_{i,l}P_iP_{i,d})$, abbreviated as $(P_{1,l} \cdots P_{i,d})$, is referred to as the *staircase* $\mathcal{S}(P_i)$, $i = 1, 2, \cdots, r_1$. For convenience, we augment the staircase by two points $P_H = (0, \infty)$ and $P_T = (\infty, 0)$ and denote the staircase as $(P_HP_{1,l}P_1 \cdots P_iP_{T,l}P_T)$, where $P_{T,l} := P_{i,d}$. Further-

(a)



(b)

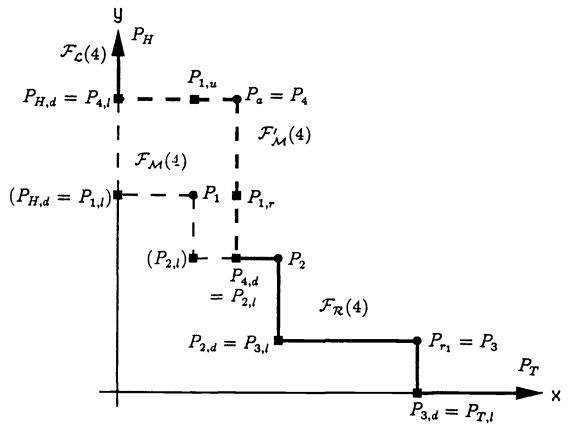


FIG. 8. (a) *Staircase* $\mathcal{S}_1$ (*where* $r_1 = 3$); (b) *staircase* $\mathcal{S}(P_4)$ (*where* $r_1 = 3$).

more, we adopt the convention that each point P_j in $\rho \cup \{P_H, P_T\}$ on a staircase has both d- and l-intersections on the staircase except P_H (which has only d-intersection $P_{H,d} := P_{1,l}$) and P_T (which has only l-intersection $P_{T,l} := P_{i,d}$). That is, we shift each $P_{j,d}$ up to the staircase and let new $P_{j,d} := P_{j+1,l}$, where $j = 1, 2, \cdots, i-1$. Staircase $\mathcal{S}(P_{r_1})$ is called the staircase of level 1, denoted $\mathcal{S}_1$, and is shown in Fig. 8(a).

Consider now the points in ρ_2, i.e., $P_{r_1+1}, P_{r_1+2}, \cdots, P_{r_1+r_2}$. Imagine dropping downward a vertical line V_i from each point P_i, $r_1 < i \leq r_1 + r_2$. The point in $\mathcal{S}_1$ that is hit by V_i is the d-intersection, denoted by $P_{i,d}$, of P_i on $\mathcal{S}_1$ (see Fig. 8(b)). For $P_i \in \rho_2$, the l-intersection on $\mathcal{S}_1$, $P_{i,l}$ is defined in the same way as for the points in ρ_1. Let P_a be the left-point in ρ_2. The l- and d-intersections, $P_{a,l}$ and $P_{a,d}$, respectively, partition $\mathcal{S}_1$ into three *substaircases*: $\mathcal{F}_{\mathcal{L}}(P_a), \mathcal{F}_{\mathcal{M}}(P_a)$, and $\mathcal{F}_{\mathcal{R}}(P_a)$, where $\mathcal{F}_{\mathcal{L}}(P_a) = (P_H P_{a,l})$ is the *leading portion* of $\mathcal{S}_1$, $\mathcal{F}_{\mathcal{M}}(P_a) = (P_{a,l} \cdots P_{a,d})$ is the *middle portion* of $\mathcal{S}_1$, and $\mathcal{F}_{\mathcal{R}}(P_a) = (P_{a,d} \cdots P_{T,l} P_T)$ is the *trailing portion* of $\mathcal{S}_1$. Let $P_{k,l}$ and P_k be two consecutive points in $\mathcal{S}_1$ such that $x_{k,l} < x_a < x_k$. The *upward vertical projection* (u-projection for short) of all the points, excluding the endpoints $P_{a,l}$ and $P_{a,d}$ in $\mathcal{F}_{\mathcal{M}}(P_a)$ on the horizontal line segment $\overline{P_{a,l}, P_a}$, gives rise to the u-projections

$P_{1,u}, P_{2,u}, \cdots, P_{k-1,u}$. Similarly, the *rightward horizontal projection* (r-projection for short) of all the points, excluding the two endpoints in $\mathcal{F}_{\mathcal{M}}(P_a)$ on the vertical line segment $\overline{P_a, P_{a,d}}$, yields r-projections $P_{1,r}, P_{2,r}, \cdots, P_{k-1,r}$. The staircase $\mathcal{S}(P_a)$ associated with P_a is obtained from $\mathcal{S}_1$ as the concatenation of $\mathcal{F}_{\mathcal{L}}(P_a)$, $\mathcal{F}'_{\mathcal{M}}(P_a)$, and $\mathcal{F}_{\mathcal{R}}(P_a)$, where $\mathcal{F}'_{\mathcal{M}}(P_a)$ is the "substaircase" $(P_{a,l}P_{1,u} \cdots P_{k-1,u}P_aP_{1,r} \cdots P_{k-1,r}P_{a,d})$ derived from projecting $\mathcal{F}_{\mathcal{M}}(P_a)$ onto $\overline{P_{a,l}, P_a}$ and $\overline{P_a, P_{a,d}}$. Thus $\mathcal{S}(P_a)$ contains not only point P_a and its l- and d-intersections $P_{a,l}$ and $P_{a,d}$, but also the leading and trailing portions of $\mathcal{S}_1$ and projections of the points in the middle portion of $\mathcal{S}_1$.

Let P_a be the left-point in $\rho_j, j > 1$. In general, the staircase $\mathcal{S}(P_i)$ of any point $P_i \in \rho_j$, is a rectilinear path of the form $(P_H\ P_{a,l}\ \mathcal{U}_a\ P_a\ \mathcal{R}_a\ P_{a+1,l}\ \mathcal{U}_{a+1}\ P_{a+1}\ \mathcal{R}_{a+1} \cdots P_{i,l}\ \mathcal{U}_i\ P_i\ \mathcal{R}_i\ P_{i,d}\ \mathcal{F}_{\mathcal{R}}(P_i))$, where the $\mathcal{U}$'s, $\mathcal{R}$'s, and $\mathcal{F}_{\mathcal{R}}(P_i)$ are, respectively, u-projections, r-projections, and the trailing portion of the previous staircase $\mathcal{S}(P_{i-1})$. Note that $\mathcal{F}_{\mathcal{L}}(P_i) = (P_H\ P_{a,l}\ \mathcal{U}_a\ P_a\ \mathcal{R}_a\ P_{a+1,l}\ \mathcal{U}_{a+1}\ P_{a+1}\ \mathcal{R}_{a+1} \cdots P_{i,l})$ and $\mathcal{F}_{\mathcal{M}}(P_i) = (P_{i,l}\ \mathcal{U}_i\ P_i\ \mathcal{R}_i\ P_{i,d})$. Now let us consider how to obtain $\mathcal{S}(P_{i+1})$ from $\mathcal{S}(P_i)$.

We first find the l-intersection $P_{i+1,l}$ on VL$= \overline{P_i, P_{i,d}}$ and the d-intersection $P_{i+1,d}$ on $\mathcal{S}(P_i)$. Let R' and R'' be two consecutive points on VL containing $P_{i+1,l}$, and let Q' and Q'' be two consecutive points on $\mathcal{S}(P_i)$ containing $P_{i+1,d}$. Thus $\mathcal{F}_{\mathcal{M}}(P_{i+1})$ is the substaircase $(R'' \cdots Q')$ of $\mathcal{S}(P_i)$, and the new $\mathcal{F}'_{\mathcal{M}}(P_{i+1})$ is obtained by projecting all points (including projections) in $\mathcal{F}_{\mathcal{M}}(P_{i+1})$ vertically upward and horizontally rightward onto $\overline{P_{i+1,l}, P_{i+1}}$, $\overline{P_{i+1}, P_{i+1,d}}$, respectively. Note that points on the same vertical line segment have the same u-projection and that points on the same horizontal line segment have the same r-projection. The staircase $\mathcal{S}(P_{i+1})$ is the concatenation of the leading portion $(P_H \cdots R')$ of $\mathcal{S}(P_i)$, $\mathcal{F}'_{\mathcal{M}}(P_{i+1})$, and the trailing portion $(Q'' \cdots P_T)$ of $\mathcal{S}(P_i)$. According to the convention adopted, we shift certain points each time a new staircase is obtained. Specifically, let $P_f \in \rho \cup \{P_H, P_T\}$ be the rightmost point on $\mathcal{S}(P_i)$, whose y-coordinate is greater than y_{i+1}, and let $P_g \in \rho \cup \{P_H, P_T\}$ be the leftmost point on $\mathcal{S}(P_i)$, whose x-coordinate is greater than x_{i+1}. We let $P_{i+1,l}$ and $P_{i+1,d}$ be the new $P_{f,d}$ and $P_{g,l}$ respectively; that is, we shift $P_{f,d}$ and $P_{g,l}$ from their original positions to $P_{i+1,l}$ and $P_{i+1,d}$, respectively. For example, in Fig. 8(b), when $\mathcal{S}(P_4)$ is obtained, $P_{2,l}$ is shifted to $P_{4,d}$, and $P_{H,d}$ is shifted to $P_{4,l}$.

3.3.2. Implementation of algorithm N-ASSIGNMENT. Clearly, for every point at level 1, its N-value equals 1 if $|\rho_1| = 1$, or 2 if $|\rho_1| > 1$. Recall that the staircase $\mathcal{S}(P_i)$ of any point $P_i \in \rho_j$, $j > 1$, is a rectilinear path of the form $(P_H\ P_{a,l}\ \mathcal{U}_a\ P_a\ \mathcal{R}_a\ P_{a+1,l}\ \mathcal{U}_{a+1}\ P_{a+1}\ \mathcal{R}_{a+1} \cdots P_{i,l}\ \mathcal{U}_i\ P_i\ \mathcal{R}_i \cdots \mathcal{U}_q\ P_q\ \mathcal{R}_q\ P_{T,l}\ P_T)$, where P_a is the left-point in ρ_j and P_q is the right-point in ρ_{j-1}. We now describe how the N-value of each point P_i is computed, and hence the weight $W(i) = N(i) - L(i)$.

We maintain a 2-3 tree for all the points P_H, $P_{i,l}$, P_i in ρ, $P_{T,l}$, and P_T of $\mathcal{S}(P_i)$ such that each leaf corresponds to a point. The leaf v corresponding to P_i in ρ is the root of a binary tree, so that the left and right subtrees, denoted by $\mathcal{U}(P_i)$ and $\mathcal{R}(P_i)$, are height-balanced and contain u-projections $\mathcal{U}_i$ and r-projections $\mathcal{R}_i$, respectively. The leaf corresponding to $P_{i,l}$, where $P_i \in \rho$, stores the maximum weight of the points dominated by $P_{i,l}$. The leaves corresponding to P_H, $P_{T,l}$, and P_T store weight 0. Each leaf w in the height-balanced subtrees corresponds to an r- or u-projection of a point P_w or its projection and stores the weight of the point P_w. The height-balanced subtrees are maintained so that at each internal node w a weight $W(w)$, which is equal to the maximum weight among those stored at the leaves rooted at w, is stored. This is referred to as the *max heap* property. The points in these trees are all maintained according to the order in which they are traversed along the staircase. Initially, the two height-balanced subtrees $\mathcal{U}(P_i)$ and $\mathcal{R}(P_i)$ at node v_i corresponding

to point P_i in ρ_1 are empty, reflecting the fact that the $\mathcal{U}$'s and $\mathcal{R}$'s are empty. All leaves in the 2-3 tree, except the leaves corresponding to points $P_i \in \rho_1$, which contain the weight $N(i) - 1$, are assigned weight equal to 0.

We now describe how to maintain the 2-3 tree and the height-balanced trees as we process the points in a level-by-level, left-to-right order. Consider the staircase $\mathcal{S}(P_i)$, as described above, and let the next point to be processed be P_{i+1}. We first perform a search in the 2-3 tree to find the two points P_a and P_b in $\rho \cup \{P_H, P_T\}$ that satisfy the following conditions:

(i) P_a is the rightmost point whose y-coordinate is greater than y_{i+1};

(ii) P_b is the leftmost point whose x-coordinate is greater than x_{i+1}.

Points in $\mathcal{R}(P_a)$ and $\mathcal{U}(P_b)$, $P_{q,l}$'s, P_q's, and all points in $\mathcal{U}(P_q)$ and $\mathcal{R}(P_q)$, where $P_q \in \rho_{L(i+1)-1}$, that are dominated by P_{i+1} are in $\mathcal{F}_{\mathcal{M}}(P_{i+1})$. We find the point P_c in $\mathcal{F}_{\mathcal{M}}(P_{i+1})$ with the maximum weight $W(c)$ and assign to P_{i+1} the appropriate N-value. More specifically, we perform the following steps on the trees while maintaining the *max heap* property of the height-balanced subtrees:

1. The leaves of the 2-3 tree that lie between P_a and P_b are deleted and replaced by three leaves corresponding to $P_{i+1,l}$, P_{i+1}, and $P_{i+1,d}$;
2. We split the tree $\mathcal{R}(P_a)$ into two subtrees $\mathcal{R}_l(P_a)$ and $\mathcal{R}_r(P_a)$ such that the (projection) points in $\mathcal{R}_l(P_a)$ have y-coordinates greater than y_{i+1} and those in $\mathcal{R}_r(P_a)$ have y-coordinates less than y_{i+1}. We replace $\mathcal{R}(P_a)$ with $\mathcal{R}_l(P_a)$;
3. We split the tree $\mathcal{U}(P_b)$ into two subtrees $\mathcal{U}_l(P_b)$ and $\mathcal{U}_r(P_b)$ such that the (projection) points in $\mathcal{U}_l(P_b)$ have x-coordinates less than x_{i+1} and those in $\mathcal{U}_r(P_b)$ have x-coordinates greater than x_{i+1}. We replace $\mathcal{U}(P_b)$ with $\mathcal{U}_r(P_b)$;
4. We find that the weight of the node corresponding to $P_{i+1,l}$ (respectively, $P_{i+1,d}$) is the maximum of those in $\mathcal{R}_r(P_a)$ and that in $P_{a,d}$ (respectively, $\mathcal{U}_l(P_b)$ and $P_{b,l}$);
5. We find that the left subtree $\mathcal{U}(P_{i+1})$ is the concatenation of the trees $\mathcal{U}_l(P_b)$ and all $\mathcal{U}(P_q) \cup \{P_q\}$, where P_q's are as defined above;
6. We find that the right subtree $\mathcal{R}(P_{i+1})$ is the concatenation of the trees $\mathcal{R}_r(P_a)$ and all $\mathcal{R}(P_q) \cup \{P_q\}$, where P_q's are as defined above;
7. If P_{i+1} is marked, we assign N-value to P_{i+1} according to assignment rules (R2) and (R3). Otherwise, we find the point P_c in $\{P_{i+1,l}\} \cup \mathcal{U}(P_{i+1}) \cup \mathcal{R}(P_{i+1}) \cup \{P_{i+1,d}\}$ with the maximum weight $W(c)$ and assign $W(c) + L(i+1) + 1$ to $N(i+1)$. The weight $W(i+1)$ is $N(i+1) - L(i+1)$.

The algorithm for N-value assignment is summarized below. Initially, we build a 2-3 tree, denoted by T_{2-3}, for points in $\mathcal{S}_1$, as discussed above. These leaves of T_{2-3} are doubly-linked from left to right by the pointers PRED and SUCC. The two height-balanced subtrees rooted at leaf v corresponding to P_j are initially empty, i.e., $\mathcal{U}(P_j) = \mathcal{R}(P_j) := nil$, for $P_j \in \rho_1$. The leaf node v corresponding to P_j is denoted by $v(P_j)$. It holds that $N(v(P_j))$ (or $N(j)$ to be consistent) $:= \min(2, |\rho_1|)$, $W(v(P_j))$ (or $W(j)$ to be consistent) $:= N(v(P_j)) - 1$, $MAXID(v(P_j)) := P_j$ for all leaves $v(P_j)$ such that $P_j \in \rho_1$ and $W(v(P_q)) := 0$, $MAXID(v(P_q)) := nil$ for all leaves $v(P_q)$ such that $P_q \in \mathcal{S}_1$ and $P_q \notin \rho_1$. Assume now that we are processing points at level $k, k > 1$ by invoking the following procedure.

Procedure N-ASSIGNMENT(k)

begin

($*$ FIND(T_{2-3}, P_i) returns points (P_a, P_b) so that P_a, P_b are in $\rho \cup \{P_H, P_T\}$ and T_{2-3}, P_a is the rightmost point whose y-coordinate is greater than y_i, and P_b is the leftmost point whose x-coordinate is greater than x_i. If P_i is the left-point of level $L(i)$, then $P_a = P_H$. If P_i is the right-point of level $L(i)$, then $P_b = P_T$. $*$)

($*$ SPLIT($\mathcal{R}(v), y$) splits tree $\mathcal{R}(v)$ by value y into two subtrees $\mathcal{R}_l(v)$ and $\mathcal{R}_r(v)$ that contain values strictly less and greater than y, respectively, and returns two pointers (l, r) to these two subtrees, respectively. $*$)

($*$ SPLICE($\mathcal{R}_1(u), \mathcal{R}_2(v)$) concatenates two trees $\mathcal{R}_1$ and $\mathcal{R}_2$ to form a new tree $\mathcal{R}$, and returns pointer l to tree $\mathcal{R}$. $*$)

($*$ We assume that in operations SPLIT and SPLICE the *max heap* property of each node w is still maintained. $*$)

($*$ $\mathcal{MAX}(u, v, \cdots, w)$ returns (m, M), where m is the maximum of the values $W(u)$, $W(v), \cdots$, and $W(w)$, and M is $MAXID(u)$, $MAXID(v), \cdots$, or $MAXID(w)$, depending on which W value is the maximum. $*$)

($*$ Let $R_k = r_1 + \cdots + r_{k-1} + r_k$ $*$)

(1) **for** $i = R_{k-1} + 1$ **to** R_k **do**

begin

(2) $(P_a, P_b) := \text{FIND}(T_{2-3}, P_i)$;

(3) **if** $\mathcal{R}(P_a)$ is not *nil*

then begin

$(l, r) := \text{SPLIT}(\mathcal{R}(P_a), y_i)$;

$\mathcal{R}(P_a) := l$;

$(maxw1, maxid1) := \mathcal{MAX}(r, v(P_{a,d}))$;

end

else begin

$r := nil$;

$maxw1 := W(v(P_{a,d}))$;

$maxid1 := MAXID(v(P_{a,d}))$;

end

(4) **if** $\mathcal{U}(P_b)$ is not *nil*

then begin

$(l', r') := \text{SPLIT}(\mathcal{U}(P_a), x_i)$;

$\mathcal{U}(P_b) := r'$;

$(maxw2, maxid2) := \mathcal{MAX}(l', v(P_{b,l}))$;

end

else begin

$l' := nil$;

$maxw2 := W(v(P_{b,l}))$;

$maxid2 := MAXID(v(P_{b,l}))$;

end

(5) $u := v(P_a)$; $\mathcal{R}(P_i) := r$;

while SUCC(SUCC(u)) $\neq v(P_b)$ **do**

begin

$w := v(P_q) := \text{SUCC}(\text{SUCC}(u))$;

$\mathcal{R}(P_i) :=$SPLICE($\mathcal{R}(P_i), \{w\} \cup \mathcal{R}(P_q)$);

```
                    u := w;
                end
(6)         u := v(P_b); U(P_i) := l';
            while PRED(PRED(u)) ≠ v(P_a) do
                begin
                    w := v(P_g) := PRED(PRED(u));
                    U(P_i) :=SPLICE(U(P_g) ∪ {w},U(P_i));
                    u := w;
                end
(7)         (maxw,maxid) := MAX(R(P_i),U(P_i),v(P_{a,d}),v(P_{b,l}));
(8)         v(P_{i,l}) := new-node( ); v(P_{a,d}) := v(P_{i,l});
            (* Create a new leaf in T_{2-3} for P_{i,l} = P_{a,d} = (x_a, y_i) pointed to
            by v(P_{i,l}) and v(P_{a,d}). *)
            W(v(P_{i,l})) := maxw1; MAXID(v(P_{i,l})) := maxid1;
            PRED(v(P_{i,l})) := v(P_a); SUCC(v(P_a)) := v(P_{i,l});
(9)         v(P_i) := new-node( );
            (* Create a new leaf in T_{2-3} for point P_i pointed to by v(P_i). *)
            W(v(P_i)) := maxw; MAXID(v(P_i)) := maxid;
            PRED(v(P_i)) := v(P_{i,l}); SUCC(v(P_{i,l})) := v(P_i);
(10)        v(P_{i,d}) := new-node( ); v(P_{b,l}) := v(P_{i,d});
            W(v(P_{i,d})) := maxw2; MAXID(v(P_{i,d})) := maxid2;
            PRED(v(P_{i,d})) := v(P_i); SUCC(v(P_i)) := v(P_{i,d});
            PRED(v(P_b)) := v(P_{i,d}); SUCC(v(P_{i,d})) := v(P_b);
(11)        precede(P_i) := nil;
            (* precede(P_i) is the preceding point of P_i; here we first assume P_i
            is a marked point. *)
            if P_i is marked
                then call N-ASSIGNMENT-(R2)-(R3);
                    (* Assign N(i) according to (R2) and (R3). *)
                else begin
                    N(i) := maxw + k + 1;
                    precede(P_i) := maxid;
                end
(12)        W(v(P_i)) := N(v(P_i)) - k;
        end (* end of the for loop *)
    end. (* end of the Procedure *)
```

There is an important property of the staircase $\mathcal{S}(P_a)$, referred to as the *projection containment property* (Lemma 3.1). Note that the u-projection of a u-projection of point P_i is considered a u-projection of P_i (i.e., $P_{i,u}$); the r-projection of point P_i is similarly defined. The u- (or r-) projection of an r- (or u-) projection of a point is simply called a projection of the point. For simplicity, the u-projection and the r-projection of point P_i are also called the projections of the point.

A point P_i dominated by P_j is said to satisfy the *projection containment property* if the following are true: (1) If the x-coordinate of $P_{j,l}$ is greater than x_i, then a projection of P_i appears at $P_{j,l}$; otherwise, at least a projection $P_{i,u}$ appears on $\overline{P_{j,l}P_j}$; (2) If the y-coordinate of $P_{j,d}$ is greater than y_i, then a projection of P_i appears at $P_{j,d}$; otherwise, at least a projection $P_{i,r}$ appears on $\overline{P_jP_{j,d}}$.

LEMMA 3.1. *Each point P_i in ρ satisfies the projection containment property with*

respect to each point P_j dominating P_i in ρ.

Proof. Assume that the points dominating P_i in ρ are $P_{\alpha_1}, P_{\alpha_2}, \cdots, P_{\alpha_z}$, where z is the number of points in ρ dominating P_i and $\alpha_1 < \alpha_2 < \cdots < \alpha_z$. We prove the lemma inductively. For inductive basis, consider P_{α_1} first. P_{α_1} must be the left-dominating point of P_i. Obviously, $P_{i,u}$ and $P_{i,r}$ are in $\mathcal{F}'_{\mathcal{M}}(P_{\alpha_1})$, and the lemma is true for P_i with respect to P_{α_1}.

Now consider point P_{α_h} and suppose inductively that the lemma is true for P_i with respect to each point P_{α_k}, where $k < h$. There are two cases.

Case 1. If $P_{\alpha_{(h-1)}}$ and P_{α_h} are at the same level then, by definition, either P_i has a projection at $P_{\alpha_{(h-1)},d}$ or $P_{i,r}$ appears at a position on $\overline{P_{\alpha_{(h-1)}}P_{\alpha_{(h-1)},d}}$, dependent upon if y_i being less than or greater than the y-coordinate of $P_{\alpha_{(h-1)},d}$, respectively. Thus P_i will have a projection at $P_{\alpha_h,l}$.

Case 2. If $P_{\alpha_{(h-1)}}$ and P_{α_h} are not at the same level, then let P_b be the leftmost point of $\mathcal{S}(P_{\alpha_{h-1}})$ in ρ dominating P_i. By induction hypothesis, there is a u-projection of P_i, i.e., $P_{i,u}$, that lies on $\overline{P_{b,l}P_b}$. The x-coordinate of $P_{i,u}$ is less than x_{α_h}, and thus $P_{i,u}$ will be projected on $\overline{P_{\alpha_h,l}P_{\alpha_h}}$ in $\mathcal{S}(P_{\alpha_h})$, for there are no points between $P_{\alpha_{(h-1)}}$ and P_{α_h} dominating P_i.

Similarly, we can show that either P_i has a projection at $P_{\alpha_h,d}$ or that $P_{i,r}$ appears on $\overline{P_{\alpha_h}P_{\alpha_h,d}}$ in $\mathcal{S}(P_{\alpha_h})$. Thus the lemma is true. □

3.4. Algorithm MAX-TWO-CHAIN. It is straightforward to implement an algorithm C-CHAIN for the candidate chain assignment phase and an algorithm ADJUSTMENT for the adjustment phase. Algorithm READ-MAX-TWO-CHAIN scans all points at level s to find a marked point P_a and assigns points to $\mathcal{M}(a)$ iteratively, according to Lemma 2.5. From Theorem 2.1, it follows that $\mathcal{M}'(a)$ is a maximum two-chain in ρ. We now summarize the entire algorithm for finding a maximum two-chain, below.

```
Procedure MAX-TWO-CHAIN (ρ)
  begin
    call LEVEL(ρ) ;
    call INITIALIZATION(ρ1) ;
    (* Assign points at level 0 and level 1 to local chains, and assign each
    point an N-value. Also, find points in S1 and build T2-3 for them. *)
    for k = 1 to s do
      begin
        call LOCAL-CHAIN(k) ;
        call N-ASSIGNMENT(k) ;
        call C-CHAIN(k) ;
        call ADJUSTMENT(k) ;
      end
    call READ-MAX-TWO-CHAIN(ρ) ;
  end.
```

We now analyze the complexity of the algorithm. Algorithm LEVEL and INITIALIZATION are easily seen to take $O(n \log n)$ time. Since algorithms LOCAL-CHAIN, C-CHAIN, ADJUSTMENT, and READ-MAX-TWO-CHAIN can be done in linear time, we concentrate on the analysis of algorithm N-ASSIGNMENT, which is the most complicated. We now analyze the complexity of algorithm N-ASSIGNMENT.

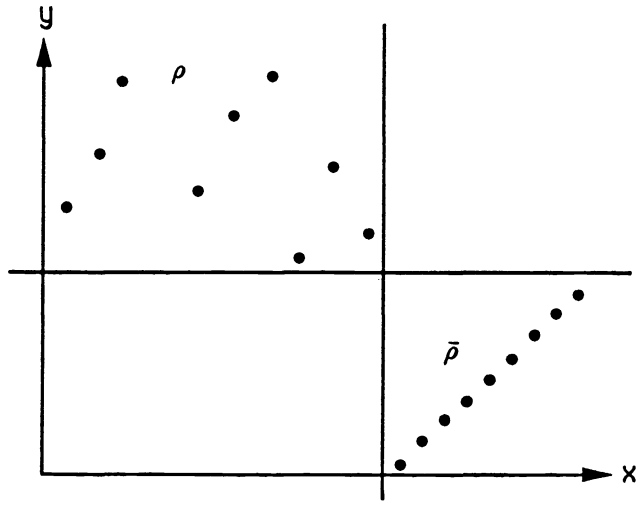


FIG. 9. ρ *and* $\bar{\rho}$.

Step (1) (in §3.3.2) can be done in $O(\log n)$ time per point. Steps (2)–(7) require SPLIT and SPLICE operations, each taking $O(\log n)$ time [AHU]. Let $P_i \in \rho_k$ be the left-dominating point of some points P_w's $\in \rho_{k-1}$. The number of times the operation SPLICE is executed is equal to twice the number of points P_w. Since a point P_w can appear at most once as a leaf in the 2-3 tree when it is inserted and is "left-dominated" by only one point at the next higher level, the total amount of time contributed by each point is at most $O(\log n)$. Thus algorithm N-ASSIGNMENT takes $O(n \log n)$ time.

It is obvious that the space requirement is only linear, and we thus have the following main result.

THEOREM 3.1. *Algorithm* MAX-TWO-CHAIN *finds a maximum two-chain in a point set* ρ *in* $\Theta(n)$ *space and* $O(n \log n)$ *time, where* $n = |\rho|$.

In the following, we show that $\Omega(n \log n)$ is a time lower bound for the maximum two-chain problem. In [KR] it is shown that the problem of finding the set of points on the dominance hull of a given set ρ of n points requires $\Omega(n \log m)$ time under the algebraic decision (computation) tree model (see [AHU], [BO]), where m is the number of maximal points in ρ. We can apply the same argument to the problem of finding a maximum chain and show that $\Omega(n \log m)$ time is required to find a maximum chain of m points. In the worst case, since the number of points on the dominance hull can be n, $\Omega(n \log n)$ is a lower bound for the maximum one-chain problem.

We now show that the maximum two-chain problem requires $\Omega(n \log n)$ time as well. Given a point set ρ, we can construct in linear time a chain $\bar{\rho}$, in which each point is crossing with all points in ρ (see Fig. 9). A maximum two-chain of $\rho \cup \bar{\rho}$ should include $\bar{\rho}$ and a maximum chain of ρ. So we can find a maximum chain of ρ by applying any maximum two-chain algorithm to $\rho \cup \bar{\rho}$. The following theorem is established.

THEOREM 3.2. *In the algebraic computation tree model, any algorithm that finds a maximum two-chain in a point set* ρ *requires* $\Omega(n \log n)$ *time, where* $n = |\rho|$.

4. Conclusion. We have presented an optimal algorithm that runs in $\Theta(n \log n)$ time and $\Theta(n)$ space for solving the maximum two-chain problem : given a set ρ of n points, find a maximum subset $\mathcal{C} \subseteq \rho$ of points such that $\mathcal{C}$ can be divided into two chains, each of which contains points that dominate each other. Whether the algorithm can be extended to handle weighted-point set version and to handle k-chains (i.e., to improve the $O(kn^2)$ time algorithm of [SLo]) or restricted versions of the problem (e.g., when some points are to be assigned entirely to chain i, $i = 1, 2$) that was solved in $O(n^2 \log n)$ time [SL2] remains to be seen.

REFERENCES

[AHU] A. V. AHO, J. E. HOPCROFT, AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison–Wesley, Reading, MA, 1974.

[AK] M. J. ATALLAH AND S. R. KOSARAJU, *An efficient algorithm for maxdominance, with applications*, Algorithmica, 4 (1989), pp. 221–236.

[BO] M. BEN-OR, *Lower bounds for algebraic computation trees*, in Proc. of 15th ACM Sympos. on Theory of Computing, Boston, MA, 1983, pp. 80–86.

[KR] S. KAPOOR AND P. RAMANAN, *Lower bounds for maximal and convex layers problems*, Algorithmica, 4 (1989), pp. 447–459.

[PS] F. P. PREPARATA AND M. I. SHAMOS, *Computational Geometry*, Springer-Verlag, New York, 1985.

[SL1] M. SARRAFZADEH AND D. T. LEE, *A new approach to topological via minimization*, IEEE Trans. Computer-Aided Design, 8 (1989), pp. 890–900.

[SL2] ———, *Topological via minimization revisited*, IEEE Trans. Comput., 40 (1991), pp. 1307–1312.

[SLo] M. SARRAFZADEH AND R. D. LOU, *Maximum k-coverings in transitive graphs*, in Proc. IEEE International Symposium on Circuits and System, May 1990, pp. 332–335; Algorithmica, 1992, to appear.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 305–320, August 1992
© 1992 Society for Industrial and Applied Mathematics

ON THE POSITION VALUE FOR COMMUNICATION SITUATIONS*

PETER BORM†, GUILLERMO OWEN‡, AND STEF TIJS†

Abstract. A new solution concept for communication situations is considered: the position value. This concept is based on an evaluation of the importance of the various communication links between the players. An axiomatic characterization of the position value is provided for the class of communication situations where the communication graphs contain no cycles. Furthermore, relations with the Myerson value are discussed, and, for special classes of communication situations, elegant calculation methods for their position values are described.

Key words. game theory, graphs, communication, Myerson value, position value

AMS(MOS) subject classification. 90D

1. Introduction. A *communication situation* corresponds to a triple (N, v, A), where $N := \{1, \cdots, n\}$ is the set of agents (players), (N, v) is a *coalitional game* having player set N and characteristic function $v : 2^N \to \mathbb{R}$ with $v(\varnothing) = 0$, and where (N, A) is an undirected *communication graph*.

The game (N, v) describes the economic possibilities of each coalition (= subgroup of players) that decides to cooperate. However, cooperation is restricted because communication is. The possibilities for communication are described by the graph (N, A), where the arc set A consists of (unordered) pairs of players. If $\{i, j\} \in A$, then the interpretation is that players i and j can communicate directly. Indirect communication between i and j is possible if there is a path in (N, A) from i to j. Because of this interpretation and since we will implicitly assume that each player is able to communicate with himself, we may restrict our attention to communication graphs without parallel arcs and loops. For convenience, we assume throughout this paper that the underlying game (N, v) is zero-normalized; i.e., $v(\{i\}) = 0$ for all $i \in N$.

Communication situations were first studied in Myerson (1977), who introduced corresponding *graph-restricted games* or *communication games* and who characterized the Shapley value (Shapley (1953)) of these games in terms of efficiency and fairness. An alternative proof of this characterization is provided in Aumann and Myerson (1988), who also address the following question. Given a coalitional game, what communication links may be expected to form between the players? Here the Shapley value of the corresponding communication games is used as a criterion. In the present paper, Myerson's communication games will be called *point games*, and the Shapley value of these games will be called the *Myerson value* (cf. Aumann and Myerson (1988)) for the corresponding communication situation. Point games and the Myerson value were also investigated in Owen (1986), who concentrated on situations where the communication graph is a tree. Other types of graph-restricted games were introduced in Rosenthal (1988a), (1988b) by putting weights on the communication arcs representing costs of communication and measures of trust or friendship, respectively. Finally, we note that Myerson (1980) generalized the idea of direct communication between two players toward direct communication between the players of certain subgroups of N (*conferences*).

The present paper offers an alternative approach to evaluate a communication situation. While Myerson's point game focuses on the role of a node of the communication

* Received by the editors February 10, 1990; accepted for publication (in revised form) May 2, 1991.

† Department of Mathematics/Nijmegen Institute for Cognition Research and Information Technology (NICI), University of Nijmegen, Toernooiveld, 6525ED, Nijmegen, the Netherlands.

‡ Department of Mathematics, Naval Postgraduate School, Monterey, California 93943.

graph (a player) in establishing communication within the various possible coalitions, this paper proposes a dual point of view and concentrates on the role of an arc (communication link). The communicative strength of an arc is measured by means of the Shapley value of a kind of "dual" game on the arcs of the communication graph: a so-called arc game. Then, assuming each player has veto power of the use of any arc that he is an endpoint of, it seems reasonable to divide the worth of an arc equally between the two players who are at its endpoints. The total amount that a player obtains in this way is called the *position value* for the player in the corresponding communication situation. This value was first introduced in Meessen (1988). Formal definitions are provided in § 2.

It may be noted that, in general, the Myerson value and the position value differ. In § 3 it is shown that for the class of those communication situations in which the communication graph contains no cycles, the position value can be characterized by four properties: additivity, component efficiency, the superfluous arc property, and the degree property. Furthermore, for the same class of communication situations, a new axiomatic characterization of the Myerson value is provided in terms of the first three properties mentioned above and the so-called communication ability property.

In § 4 we derive a relation between the *dividends* (cf. Harsanyi (1959)) of an arc game and the dividends of the coalitional game underlying the communication situation. For the position value, this leads to computational results in the manner of Owen (1986) for special subclasses of communication situations in which the underlying coalitional game is a pure overhead game or a quadratic measure game. These results are described in § 5. The paper concludes with some remarks for the case when the communication graph does contain cycles.

Preliminaries. Let $N := \{1, \cdots, n\}$ and $2^N := \{S | S \subset N\}$. By G^N we denote the class of all coalitional games (N, v) and by G_0^N, the subclass of all zero-normalized games. A game (N, v) often will be identified with its characteristic function v.

Let $v \in G^N$. Then the *Shapley value* $\Phi(v)$ of v (cf. Shapley (1953)) is defined by

$$\Phi_i(v) = \frac{1}{n!} \sum_{\sigma \in P(N)} (v(PR_\sigma(i) \cup \{i\}) - v(PR_\sigma(i)))$$

for all $i \in N$, where $P(N)$ is the set of all permutations of N and

$$PR_\sigma(i) := \{j \in N | \sigma(j) < \sigma(i)\}$$

denotes the set of *predecessors* of player i according to σ. Furthermore, having the *unanimity game* $u_S \in G^N$ on S defined by

$$u_S(T) = \begin{cases} 1 & \text{if } T \supset S, \\ 0 & \text{else,} \end{cases}$$

for all $S \in 2^N \setminus \{\varnothing\}$, we find that $v = \sum_{S \in 2^N \setminus \{\varnothing\}} \Delta_v(S) u_S$, where the *dividends* $\Delta_v(S)$ (cf. Harsanyi (1959)) are given by

$$\Delta_v(S) = \sum_{T \subset S} (-1)^{|S| - |T|} v(T).$$

For the Shapley value, it readily follows that

$$\Phi_i(u_S) = \begin{cases} \dfrac{1}{|S|} & \text{if } i \in S, \\ 0 & \text{else} \end{cases}$$

for all $S \in 2^N \setminus \{\varnothing\}$ and, consequently,

$$\Phi_i(v) = \sum_{S \subset N : i \in S} \frac{\Delta_v(S)}{|S|}$$

for all $i \in N$. Finally, we define the empty sum to be zero.

2. Myerson value and position value. Let CS^N denote the class of all communication situations (N, v, A) with fixed player set N as described in § 1. So, especially, (N, v) is zero-normalized, and (N, A) is an undirected graph without parallel arcs and loops.

Let $(N, v, A) \in CS^N$. It is clear that the communication possibilities within N, given by the graph (N, A), determine a partition N/A of N into (communication) *components*. So a coalition T is a component within N if and only if all players in T can communicate and if there is no coalition $\bar{T}$ with $T \subset \bar{T}$ and $T \neq \bar{T}$ in which all players can communicate. Similarly, we can define components within each given coalition S by only allowing the communication possibilities given by the subgraph $(S, A(S))$, where

$$A(S) := \{\{i,j\} \in A \mid i \in S, j \in S\}. \tag{1}$$

Then a partition of S results, which will be denoted by S/A.

The following notation will be used frequently. Let the player set N and the game $v \in G_0^N$ be fixed. Then, for each $S \subset N$ and each $L \subset \{\{i,j\} \mid i \in N, j \in N\}$,

$$r^v(S,L) := \sum_{T \in S/L} v(T) \tag{2}$$

will denote the reward for the coalition S having the communication arcs in $L(S) \subset L$ available (cf. (1)). Note that $r^v(\varnothing, L) = r^v(S, \varnothing) = 0$ for all S and L.

Now we can formulate the following definition.

DEFINITION (cf. Myerson (1977)). Let $(N, v, A) \in CS^N$. Then the *point game* (N, r_A^v) corresponding to (N, v, A) is given by

$$r_A^v(S) := r^v(S,A) \quad \text{for all } S \in 2^N. \tag{3}$$

Furthermore, the *Myerson value* $\mu(N, v, A) \in \mathbb{R}^N$ corresponds to the Shapley value of (N, r_A^v), so

$$\mu(N,v,A) := \Phi(N, r_A^v). \tag{4}$$

Another type of game corresponding to a communication situation is introduced in the definition below.

DEFINITION. Let $(N, v, A) \in CS^N$. Then the *arc game* (A, r_N^v) corresponding to (N, v, A) is given by

$$r_N^v(L) = r^v(N,L) \quad \text{for all } L \in 2^A. \tag{5}$$

Furthermore, the *position value* $\pi(N, v, A) \in \mathbb{R}^N$ is given by

$$\pi_i(N,v,A) := \sum_{a \in A_i} \frac{1}{2} \Phi_a(A, r_N^v) \quad \text{for all } i \in N, \tag{6}$$

where

$$A_i := \{\{i,j\} \in A \mid j \in N\} \tag{7}$$

denotes the set of all arcs of which player i is an endpoint.

If there can be no misunderstanding, the upper index v will be omitted from the notation above. The following example illustrates the various concepts introduced above.

Example 2.1. For the communication situation (N, v, A), let $N = \{1, 2, 3\}$, $A = \{\{1, 3\}, \{2, 3\}\}$, and let v equal the unanimity game $u_{\{1,2\}}$. This situation is schematically represented in Fig. 1. Then (N, r_A) is given by

$$r_A(S) = \begin{cases} 1 & \text{if } S = N, \\ 0 & \text{else.} \end{cases}$$

So $\mu(N, v, A) = \Phi(N, r_A) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

Furthermore, (A, r_N) is given by

$$r_N(L) = \begin{cases} 1 & \text{if } L = A, \\ 0 & \text{else.} \end{cases}$$

So, with $a := \{1, 3\}$ and $b := \{2, 3\}$,

$$\Phi_a(A, r_N) = \Phi_b(A, r_N) = \tfrac{1}{2}$$

and

$$\pi_1(N, v, A) = \tfrac{1}{2}\Phi_a(A, r_N) = \tfrac{1}{4}, \qquad \pi_2(N, v, A) = \tfrac{1}{2}\Phi_b(A, r_N) = \tfrac{1}{4},$$
$$\pi_3(N, v, A) = \tfrac{1}{2}\Phi_a(A, r_N) + \tfrac{1}{2}\Phi_b(A, r_N) = \tfrac{1}{2}.$$

Note that, for the game in Example 2.1, the position value for each player $i \in N$ is the same multiple of the *degree* $d_i(N, A) := |A_i|$ of the corresponding point in the graph (N, A), which in some sense is a "natural" measure for the importance of a point (player) in the (communication) graph. This relation between the position value and degree will play an important role in the axiomatic characterization of the position value given in § 3.

This section is concluded with an example showing that even if there are no restrictions on communication at all, i.e., if $A = \{\{i, j\} \in 2^N \mid i \neq j\}$, the Myerson value (which then just is the Shapley value of the underlying game (N, v)) may differ from the position value.

Example 2.2. Let (N, v, A) be given by

$$N = \{1, 2, 3\}, v = u_{\{1,2\}} \quad \text{and} \quad A = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\},$$

as represented in Fig. 2. Then $r_A = v$ and, with $a := \{1, 2\}$, $b := \{1, 3\}$, and $c := \{2, 3\}$,

$$r_N(L) = \begin{cases} 1 & \text{if } a \in L \text{ or } \{b, c\} \subset L, \\ 0 & \text{else.} \end{cases}$$

Hence $\mu(N, v, a) = \Phi(N, u_{\{1,2\}}) = (\frac{1}{2}, \frac{1}{2}, 0)$, $\Phi_a(r_N) = \frac{2}{3}$, $\Phi_b(r_N) = \Phi_c(r_N) = \frac{1}{6}$, and $\pi(N, v, A) = (\frac{5}{12}, \frac{5}{12}, \frac{2}{12})$.

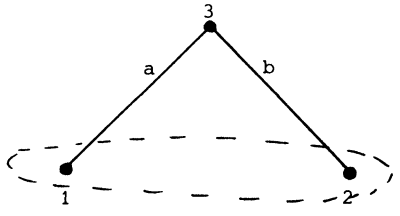


FIG. 1

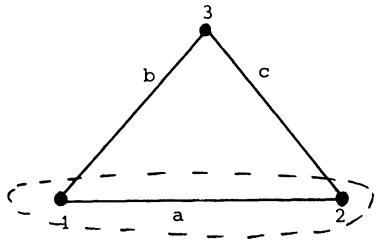


FIG. 2

3. Axiomatic characterizations. In this section we consider properties of allocation rules $\gamma : CS^N \to \mathbb{R}^N$ and, in particular, of the rules μ and π introduced in the previous section. Myerson (1977) showed the following result.

THEOREM 3.1. *The Myerson value $\mu : CS^N \to \mathbb{R}^N$ is the unique allocation rule that satisfies component efficiency and fairness.*

Here, a rule $\gamma : CS^N \to \mathbb{R}^N$ is called *component efficient* if

$$\sum_{i \in C} \gamma_i(N, v, A) = v(C) \tag{8}$$

for all $(N, v, A) \in CS^N$ and all components $C \in N/A$. Note that, for all $C \in N/A$, we have that $v(C) = \sum_{T \in C/A} v(T) = r(C, A)$. Furthermore, $\gamma : CS^N \to \mathbb{R}^N$ is called *fair* (cf. Myerson (1977)) if

$$\gamma_i(N, v, A) - \gamma_i(N, v, A \setminus \{\{i,j\}\}) = \gamma_j(N, v, A) - \gamma_j(N, v, A \setminus \{\{i,j\}\}) \tag{9}$$

for all $(N, v, A) \in CS^N$ and $\{i, j\} \in A$. So, if we use a fair allocation criterion in the manner of Myerson and an arc is removed from the communication graph, then the two players connected by this arc lose (or gain) the same amount.

The following example shows that the position value $\pi : CS^N \to \mathbb{R}^N$ does not satisfy this fairness criterion.

Example 3.1. Consider the three-person communication situation (N, v, A) of Example 2.1. Then

$$\pi_1(N, v, A) - \pi_1(N, v, A \setminus \{\{1,3\}\}) = \tfrac{1}{4} - 0 = \tfrac{1}{4},$$

while

$$\pi_3(N, v, A) - \pi_3(N, v, A \setminus \{\{1,3\}\}) = \tfrac{1}{2} - 0 = \tfrac{1}{2}.$$

A rule $\gamma : CS^N \to \mathbb{R}^N$ is called *additive* if

$$\gamma(N, v + w, A) = \gamma(N, v, A) + \gamma(N, w, A) \tag{10}$$

for all $v, w \in G_0^N$ and communication graphs (N, A).

An arc $a \in A$ is called *superfluous* for the communication situation (N, v, A) if

$$r(N, L) = r(N, L \cup \{a\}) \quad \text{for all } L \subset A. \tag{11}$$

This means that in each communication subsystem the presence of a superfluous arc does not affect the gains of the grand coalition. A rule $\gamma : CS^N \to \mathbb{R}^N$ is said to have the *superfluous arc property* if

$$\gamma(N, v, A) = \gamma(N, v, A \setminus \{a\}) \tag{12}$$

for all $(N, v, A) \in CS^N$ and superfluous arcs $a \in A$.

Now we can formulate the following lemma.

LEMMA 3.1. *Both the Myerson value* $\mu : CS^N \to \mathbb{R}^N$ *and the position value* $\pi : CS^N \to \mathbb{R}^N$ *satisfy component efficiency, additivity, and the superfluous arc property.*

Proof. For both μ and π, additivity follows trivially from the additivity of the Shapley value, since $r_A^{v+w} = r_A^v + r_A^w$ and $r_N^{v+w} = r_N^v + r_N^w$ for all $v, w \in G_0^N$ and all communication graphs (N, A).

For μ, component efficiency follows from Theorem 3.1. To prove the component efficiency for π, let $(N, v, A) \in CS^N$ and $C \in N/A$. Then

$$\sum_{i \in C} \pi_i(N,v,A) = \sum_{i \in C} \sum_{a \in A_i} \frac{1}{2} \Phi_a(A, r_N) = \sum_{a \in A(C)} \Phi_a(A, r_N)$$

$$\overset{(*)}{=} \sum_{a \in A(C)} \Phi_a(A(C), r_N) = r_N(A(C)) = v(C),$$

where equality $(*)$ follows from the definition of the Shapley value and the fact that

$$r(N, L \cup \{a\}) - r(N, L) = r(N, (L \cap A(C)) \cup \{a\}) - r(N, L \cap A(C))$$

for all $L \subset A$ and $a \in A(C)$.

Now let $(N, v, A) \in CS^N$ have a superfluous arc $a \in A$. To prove the superfluous arc property for μ, it suffices to show that $r_A = r_{A \setminus \{a\}}$.

Let $S \subset N$. Then

$$r_A(S) = \sum_{T \in S/A} v(T) = \sum_{T \in S/A(S)} v(T) = \sum_{T \in N/A(S)} v(T) = r(N, A(S)),$$

where the third equality follows from the fact that v is zero-normalized and, similarly,

$$r_{A \setminus \{a\}}(S) = r(N, A(S) \setminus \{a\}).$$

So (11) immediately implies that $r_A(S) = r_{A \setminus \{a\}}(S)$ for all S.

Proving the superfluous arc property for π, expression (11) directly implies that the arc a is a zero-player in the game (A, r_N) and $\Phi_a(A, r_N) = 0$. Furthermore, it follows that $\Phi_b(A, r_N) = \Phi_b(A \setminus \{a\}, r_N)$ for all $b \in A \setminus \{a\}$. Hence, $\pi(N, v, A) = \pi(N, v, A \setminus \{a\})$. □

A communication situation $(N, v, A) \in CS^N$ is called *arc anonymous* if there exists a function $f: \{0, 1, \cdots, |A|\} \to \mathbb{R}$ such that

$$r(N, L) = f(|L|) \quad \text{for all } L \subset A. \tag{13}$$

Since in an arc anonymous communication situation all arcs are equally important (cf. Lemma 3.2), the communicative strength of a node (player) can be measured by its degree, and therefore it seems reasonable to allocate the gains of the grand coalition proportional to this degree. Formally, an allocation rule $\gamma : CS^N \to \mathbb{R}^N$ is said to have the *degree property* if for all arc anonymous communication situations (N, v, A), we have that $\gamma(N, v, A)$ is a multiple of the degree vector $d(N, A) := (d_i(N, A))_{i \in N}$, i.e.,

$$\gamma(N, v, A) = \alpha d(N, A) \quad \text{for some } \alpha \in \mathbb{R}. \tag{14}$$

An equivalent formulation of (13) is given in the following lemma. The proof is straightforward and is therefore omitted.

LEMMA 3.2. *Let* $(N, v, A) \in CS^N$. *Then* (N, v, A) *is arc anonymous if and only if* $r(N, L \setminus \{a\}) = r(N, L \setminus \{b\})$ *for all* $L \subset A$ *and* $a, b \in L$.

The Myerson value μ does not have the degree property as is seen in the next example.

Example 3.2. The communication situation (N, v, A) of Example 2.1 is arc anonymous since (13) holds with $f: \{0, 1, 2\} \to \mathbb{R}$ given by $f(0) = f(1) = 0, f(2) = 1$. Furthermore,

$$d(N,A)=(1,1,2), \qquad \pi(N,v,A)=(\tfrac{1}{4},\tfrac{1}{4},\tfrac{1}{2})=\tfrac{1}{4}d(N,A),$$

and $\mu(N, v, A) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, which is not a multiple of $d(N, A)$.

However, we have the following lemma.

LEMMA 3.3. *The position value $\pi : CS^N \to \mathbb{R}^N$ satisfies the degree property.*

Proof. Let $(N, v, A) \in CS^N$ be arc anonymous. If $A = \varnothing$, then $\pi(N, v, A) = 0 = d(N, A)$. So assume that $A \neq \varnothing$. Let f be as in (13) and let $a \in A$. Since (A, r_N) is a symmetric game (i.e., all arcs are substitutes) we have that

$$\Phi_a(A,r_N)=\frac{1}{|A|}r_N(A)=\frac{1}{|A|}f(|A|).$$

Hence

$$\pi_i(N,v,A)=\sum_{a\in A_i}\frac{1}{2}\Phi_a(A,r_N)=\sum_{a\in A_i}\frac{1}{2}\cdot\frac{f(|A|)}{|A|}=\frac{1}{2}\frac{f(|A|)}{|A|}\cdot|A_i|=\frac{1}{2}\frac{f(|A|)}{|A|}\cdot d_i(N,A)$$

for all $i \in N$. □

Now we prove that the above-mentioned properties of the position value characterize π completely on the subclass CS^N_* of *cycle-free* communication situations where the communication graphs contain no cycles.

THEOREM 3.2. *The (restriction of the) position value π is the unique allocation rule on CS^N_* that satisfies component efficiency, additivity, the superfluous arc property, and the degree property.*

Proof. Let $\gamma : CS^N_* \to \mathbb{R}$ satisfy the four properties stated in the theorem.

Because of Lemmas 3.1 and 3.3, it suffices to show that $\gamma(N, v, A) = \pi(N, v, A)$ for all $(N, v, A) \in CS^N_*$. Let (N, A) be a communication graph without cycles. By additivity and by the fact that $\{u_S \mid |S| \geqq 2\}$ is a basis of the class G^N_0 of zero-normalized games, it remains to prove that

$$\gamma(N,\beta u_S,A)=\pi(N,\beta u_S,A) \tag{15}$$

for all $\beta \in \mathbb{R}$ and $S \in 2^N$ with $|S| \geqq 2$. Let $\beta \in \mathbb{R}$ and $S \in 2^N$, $|S| \geqq 2$, be fixed throughout the proof. Furthermore, for notational convenience we define $w := \beta u_S$. To prove (15) we will distinguish between two cases.

The first case is that there is no component $C \in N/A$ with $S \subset C$. Then $r(N, L) = 0$ for all $L \subset A$. So $\Phi_a(A, r^w_N) = 0$ for all $a \in A$ and, consequently, $\pi_i(N, v, A) = 0$ for all $i \in N$. Furthermore, since in this case each arc is superfluous, the superfluous arc property implies that $\gamma(N, w, A) = \gamma(N, w, \varnothing)$. Then, since $(N, w, \varnothing)$ trivially is arc anonymous, the degree property implies that there is an $\alpha \in \mathbb{R}$ such that

$$\gamma_i(N,w,\varnothing)=\alpha d_i(N,\varnothing)=0$$

for all $i \in N$. Hence $\pi = \gamma$.

Second, let $C \in N/A$ be such that $S \subset C$. Then $(C, A(C))$ is a tree, and there exists a (unique) set $H(S) \subset C$ defined by

$$H(S):=\cap\{T\mid S\subset T\subset C,(T,A(T)) \text{ is a connected (sub)graph}\}, \tag{16}$$

which is called the *connected hull* of S (cf. Owen (1986, Thm. 5)).

It is easy to verify that

$$r^w(N,L)=\begin{cases}\beta & \text{if } A(H(S))\subset L,\\ 0 & \text{else.}\end{cases} \tag{17}$$

Hence

$$\Phi_a(A,r_N^w)=\begin{cases}|A(H(S))|^{-1}\beta & \text{if } a\in A(H(S)),\\ 0 & \text{else,}\end{cases}$$

which implies that

$$\begin{aligned}\pi_i(N,w,A)&=\sum_{a\in A_i\cap A(H(S))}\frac{1}{2}|A(H(S))|^{-1}\beta\\ &=\frac{d_i(N,A(H(S)))}{2|A(H(S))|}\beta=\frac{d_i(N,A(H(S)))}{\sum_{j\in N}d_j(N,A(H(S)))}\beta\end{aligned} \tag{18}$$

for all $i\in N$. Furthermore, (17) implies that each arc $a\notin A(H(S))$ is superfluous, and so the superfluous arc property implies that

$$\gamma(N,w,A)=\gamma(N,w,A(H(S))). \tag{19}$$

The communication situation $(N, w, A(H(S)))$ is arc anonymous because of Lemma 3.2 and the fact that

$$r^w(N,L\setminus\{a\})=r^w(N,L\setminus\{b\})=0$$

for all $L\subset A(H(S)))$ and $a, b\in L$. Therefore the degree property implies that there is an $\alpha\in\mathbb{R}$ such that

$$\gamma_i(N,w,A(H(S)))=\alpha d_i(N,A(H(S))) \quad \text{for all } i\in N. \tag{20}$$

So, especially, $\gamma_i(N, w, A(H(S))) = 0$ for all $i\in N\setminus H(S)$.

Using component efficiency, we find that

$$\sum_{i\in C}\gamma_i(N,w,A(H(S)))=\sum_{i\in H(S)}\gamma_i(N,w,A(H(S)))=w(C)=\beta.$$

So from (20) we may conclude that

$$\alpha=\left(\sum_{i\in H(S)}d_i(N,A(H(S)))\right)^{-1}\beta. \tag{21}$$

Combining these results gives $\gamma=\pi$. □

By introducing one other property for allocation rules, we will be able to provide a new axiomatic characterization of the restriction of the Myerson value to communication situations in CS_*^N.

A communication situation $(N, v, A)\in CS^N$ is called *point anonymous* if there exists a function $f\colon\{0, 1, \cdots, |D|\}\to\mathbb{R}$ such that

$$r(S,A)=f(|S\cap D|) \tag{22}$$

for all $S\in 2^N$, where

$$D:=\{i\in N\mid d_i(N,A)>0\}. \tag{23}$$

An allocation rule $\gamma: CS^N\to\mathbb{R}^N$ is said to have the *communication ability property* if, for all point anonymous communication situations (N, v, A), there exists an $\alpha\in R$

such that

$$\gamma_i(N,v,A)=\begin{cases}\alpha & \text{for all } i\in D,\\ 0 & \text{else.}\end{cases} \tag{24}$$

Then we have the following lemma.

LEMMA 3.4. *The Myerson value $\mu : CS^N \to \mathbb{R}^N$ satisfies the communication ability property.*

Proof. Let (N, v, A) be point anonymous and let f and D be as in (22) and (23), respectively. Because each player $i \in N\backslash D$ is a dummy player in (N, r_A) and all players in D are substitutes, it is found that

$$\mu_i(N,v,A)=\Phi_i(N,r_A)=\begin{cases}\dfrac{r_A(N)}{|D|} & \text{if } i\in D,\\ 0 & \text{else}\end{cases}=\begin{cases}\dfrac{f(|D|)}{|D|} & \text{if } i\in D,\\ 0 & \text{else.}\end{cases}$$ □

THEOREM 3.3. *The (restriction of the) Myerson value μ is the unique allocation rule on CS^N_* that satisfies component efficiency, additivity, the superfluous arc property, and the communication ability property.*

Proof. Let $\gamma : CS^N_* \to \mathbb{R}^N$ satisfy the four properties above. Let the cycle-free communication graph (N, A), the real $\beta \in \mathbb{R}$, and the coalition $S \in 2^N$, $|S| \geqq 2$, be fixed, and define $w := \beta u_S$. Then Lemmas 3.1 and 3.4, and additivity, imply that it suffices to show that $\gamma(N, w, A) = \mu(N, w, A)$.

If there is no component $C \in N/A$ with $S \subset C$, then $r^w(T, A) = 0$ for all $T \in 2^N$. Consequently, $\mu_i(N, w, A) = \Phi_i(N, r^w_A) = 0$ for all $i \in N$. Furthermore, we trivially have that (N, w, A) is point anonymous, so there exists an $\alpha \in \mathbb{R}$ such that $\gamma_i(N, w, A) = \alpha$ for all $i \in N$ with $d_i(N, A) > 0$ and $\gamma_i(N, w, A) = 0$ otherwise. Using component efficiency, we may conclude that $\alpha = 0$ and $\gamma(N, w, A) = \mu(N, w, A)$.

Let $C \in N/A$ be such that $S \subset C$ and let $H(S)$ be as in the proof of Theorem 3.2. Then it is easily checked that

$$r^w(T,A)=\begin{cases}\beta & \text{if } H(S)\subset T,\\ 0 & \text{else}\end{cases}$$

and, consequently,

$$\mu_i(N,w,A)=\begin{cases}(1/|H(S)|)\beta & \text{if } i\in H(S),\\ 0 & \text{else.}\end{cases}$$

Furthermore, as we have seen in the proof of Theorem 3.2, the superfluous arc property implies that

$$\gamma(N,w,A)=\gamma(N,w,A(H(S))).$$

Then, since $(N, w, A(H(S)))$ is point anonymous with $D = H(S)$,

$$f(0)=\cdots=f(|D|-1)=0 \quad \text{and} \quad f(|D|)=\beta,$$

the communication ability property and component efficiency imply that $\gamma(N, w, A) = \mu(N, w, A)$. □

4. A relation between dividends. In this section we derive a result that can be used to compute the position value for various subclasses of communication situations in which the communication graph contains no cycles. This result is based upon the fact that, for each communication situation (N, v, A) in CS^N_*, the coefficients (dividends) of

the corresponding arc game r_N^v, with respect to the basis of (arc) unanimity games, can be expressed in terms of the dividends of the underlying game v. It may be noted that Owen (1986) has derived similar results for computing the Myerson value.

For a communication situation $(N, v, A) \in CS_*^N$ and the corresponding arc game (A, r_N^v), we can write

$$r_N^v = \sum_{L \in 2^A \setminus \{\varnothing\}} \Gamma_v(L) u_L,$$

where, for all $L \in 2^A \setminus \{\varnothing\}$, (A, u_L) is the (arc) unanimity game on L and

$$\Gamma_v(L) := \Delta_{r_N^v}(L) = \sum_{K \subset L} (-1)^{|L|-|K|} r_N^v(K)$$

denotes the dividend for L in the game r_N^v.

Then, extending definition (16) of the connected hull of a coalition S in a cycle-free communication graph (N, A) by setting $H(S) = \varnothing$ if there is no component $C \in N/A$ that contains S, the dividends of the arc game r_N^v and the underlying game v are related in the following way.

THEOREM 4.1. *Let* $(N, v, A) \in CS_*^N$. *Then*

$$\Gamma_v(L) = \sum_{S \in \Sigma(L)} \Delta_v(S) \tag{25}$$

for all $L \in 2^A \setminus \{\varnothing\}$, *where*

$$\Sigma(L) := \{S \in 2^N \setminus \{\varnothing\} \mid H(S) \neq \varnothing, L = A(H(S))\} \tag{26}$$

denotes the set of those coalitions that exactly must use all communication arcs in L *to communicate.*

Proof. Let $L \in 2^A \setminus \{\varnothing\}$. Then

$$\begin{aligned}
\Gamma_v(L) &= \sum_{K \subset L} (-1)^{|L|-|K|} \sum_{C \in N/K} v(C) = \sum_{K \subset L} (-1)^{|L|-|K|} \sum_{C \in N/K} \sum_{S \in 2^N \setminus \{\varnothing\}} \Delta_v(S) u_S(C) \\
&= \sum_{K \subset L} (-1)^{|L|-|K|} \sum_{C \in N/K} \sum_{S \in 2^C \setminus \{\varnothing\}} \Delta_v(S) \\
&= \sum_{S \in 2^N \setminus \{\varnothing\}} \Delta_v(S) \sum_{K \subset L} \sum_{C \in N/K : S \subset C} (-1)^{|L|-|K|} \\
&= \sum_{S \in 2^N \setminus \{\varnothing\}} \Delta_v(S) \sum_{K \subset L} |\{C \in N/K \mid S \subset C\}| (-1)^{|L|-|K|} \\
&= \sum_{S \in 2^N \setminus \{\varnothing\} : H(S) \neq \varnothing} \Delta_v(S) \sum_{K : A(H(S)) \subset K \subset L} (-1)^{|L|-|K|} \\
&= \sum_{S \in \Sigma(L)} \Delta_v(S),
\end{aligned}$$

where the last equality follows from the fact that, for each $S \in 2^N \setminus \{\varnothing\}$ such that $H(S) \neq \varnothing$, and with $l := |L|$ and $a := |A(H(S))|$,

$$\begin{aligned}
\sum_{K : A(H(S)) \subset K \subset L} (-1)^{|L|-|K|} &= \sum_{k=a}^{l} (-1)^{l-k} \binom{l-a}{k-a} \\
&= (-1)^{l-a} \sum_{k=0}^{l-a} (-1)^k \binom{l-a}{k} \\
&= \begin{cases} 1 & \text{if } l = a, \\ 0 & \text{else.} \end{cases}
\end{aligned}$$

□

In Lemma 4.1, below, the set $\Sigma(L)$ of expression (26) is characterized in an easier way. Here we use the following definitions. Let (N, A) be a communication graph. Then, for $L \subset A$, $N(L) \subset N$ is defined to be the set of all players that are endpoints of an arc in L, and, if (N, A) is a *tree* (i.e., a connected graph that contains no cycles), then

$$\text{Ext}\,(N,A) := \{ i \in N \mid d_i(N,A) = 1 \} \tag{27}$$

denotes the nonempty set of *extreme points* of (N, A).

LEMMA 4.1. *Let (N, A) be a (communication) graph that contains no cycles. Let $L \in 2^A \setminus \{\varnothing\}$. Then the following two assertions hold.*

(i) *If $(N(L), L)$ is not a tree, then $\Sigma(L) = \varnothing$.*

(ii) *If $(N(L), L)$ is a tree, then*

$$\Sigma(L) = \{ S \subset N(L) \mid \text{Ext}\,(N(L),L) \subset S \}. \tag{28}$$

Proof. Condition (i) is trivial, and the proof of (ii) is a straightforward but tiresome exercise from the definitions. Its most important ingredients are that, for a tree $(N(L), L)$,

$$H(\text{Ext}\,(N(L),L)) = H(N(L)) = N(L) \quad \text{and} \quad A(N(L)) = L. \qquad \square$$

The above results are illustrated in the following example.

Example 4.1. Consider the four-person communication situation (N, v, A) with $v = u_{\{1,2\}}$ and $A = \{a, b, c\}$, where $a = \{1, 4\}$, $b = \{2, 3\}$, and $c = \{3, 4\}$, as represented in Fig. 3. With, e.g., $L = \{b, c\}$, we find that $N(L) = \{2, 3, 4\}$ and Ext $(N(L), L) = \{2, 4\}$. Lemma 4.1 then implies that $\Sigma(\{b, c\}) = \{\{2, 4\}, \{2, 3, 4\}\}$. In a similar way, we obtain

$$\Sigma(\{a,b\}) = \varnothing, \quad \Sigma(\{a\}) = \{1,4\}, \quad \Sigma(\{b\}) = \{2,3\}, \quad \Sigma(\{c\}) = \{3,4\},$$

$$\Sigma(\{a,c\}) = \{\{1,3\},\{1,3,4\}\}, \quad \text{and}$$

$$\Sigma(\{a,b,c\}) = \{\{1,2\},\{1,2,3\},\{1,2,4\},\{1,2,3,4\}\}.$$

Therefore, since

$$\Delta_v(S) = \begin{cases} 1 & \text{if } S = \{1,2\}, \\ 0 & \text{else,} \end{cases}$$

we have

$$\Gamma_v(L) = \sum_{S \in \Sigma(L)} \Delta_v(S) = \begin{cases} 1 & \text{if } L = \{a,b,c\}, \\ 0 & \text{else.} \end{cases}$$

5. Examples. (i) *Unanimity games.* Using Theorem 4.1, we will re-prove expression (18) (for $\beta = 1$), which describes the position value for each cycle-free communication situation in which the underlying game is a unanimity game.

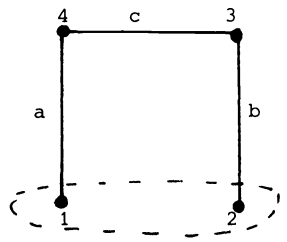


FIG. 3

Let $(N, v, A) \in CS^N_*$ be a fixed communication situation with $v = u_S$ for some $S \in 2^N$, $|S| \geqq 2$. It is clear that for the dividends $\Delta_v(T)$, we have

$$\Delta_v(T) = \begin{cases} 1 & \text{if } T = S, \\ 0 & \text{else.} \end{cases}$$

Furthermore, for all $L \in 2^A \setminus \{\varnothing\}$, the dividends $\Gamma_v(L)$ are given by

$$\Gamma_v(L) = \sum_{T \in \Sigma(L)} \Delta_v(T) = \begin{cases} 1 & \text{if } S \in \Sigma(L), \\ 0 & \text{else} \end{cases} = \begin{cases} 1 & \text{if } H(S) \neq \varnothing \text{ and } L = A(H(S)), \\ 0 & \text{else.} \end{cases}$$

So, if $H(S) = \varnothing$, then $\pi_i(N, v, A) = 0$ for all $i \in N$. Else, if $H(S) \neq \varnothing$, then

$$\pi_i(N, v, A) = \sum_{a \in A_i} \frac{1}{2} \Phi_a(A, r^v_N) = \sum_{a \in A_i} \frac{1}{2} \sum_{L : a \in L} \frac{\Gamma_v(L)}{|L|}$$

$$= \sum_{a \in A_i \cap A(H(S))} \frac{1}{2|A(H(S))|} = \frac{d_i(N, A(H(S)))}{\sum_{j \in N} d_j(N, A(H(S)))}$$

for all $i \in N$.

(ii) *Pure overhead games*. Let T be an arbitrary subset of N. Then the (zero-normalization of the) *pure overhead game* (N, p_T) on T (cf. Owen (1986)) is defined by

$$p_T(S) = \begin{cases} -1 + |S \cap T| & \text{if } S \cap T \neq \varnothing, \\ 0 & \text{else.} \end{cases}$$

It is easily verified that

$$\Delta_{p_T}(S) = \begin{cases} (-1)^{|S|} & \text{if } S \subset T \text{ and } |S| \geqq 2, \\ 0 & \text{else.} \end{cases}$$

Considering a communication situation $(N, p_T, A) \in CS^N_*$, Theorem 4.1 and Lemma 4.1 imply that, for $L \in 2^A \setminus \{\varnothing\}$,

$$\Gamma_{p_T}(L) = \begin{cases} \displaystyle\sum_{S : \text{Ext}(N(L), L) \subset S \subset (N(L) \cap T)} (-1)^{|S|} & \text{if } (N(L), L)) \text{ is a tree,} \\ 0 & \text{otherwise,} \end{cases} \tag{29}$$

$$= \begin{cases} (-1)^{|N(L) \cap T|} & \text{if Ext}(N(L), L) = N(L) \cap T, \\ 0 & \text{otherwise.} \end{cases}$$

A special, simple case occurs when $T = N$.

The condition $\text{Ext}(N(L), L) = N(L) \cap T$ then boils down to $\text{Ext}(N(L), L) = N(L)$, which is only satisfied if $|L| = 1$. Hence,

$$\Gamma_{p_N}(L) = \begin{cases} 1 & \text{if } |L| = 1, \\ 0 & \text{else.} \end{cases}$$

Consequently, we obtain the following simple expression for the position value:

$$\pi_i(N, p_N, A) = \tfrac{1}{2} d_i(N, A) \tag{30}$$

for all $i \in N$. It follows (cf. Owen (1986)) that in this case ($T = N$), the Myerson value is equal to the position value. In the more general case, this need not be true. This is illustrated in the next example.

Example 5.1. Consider the five-person communication situation (N, p_T, A), where $T = \{1, 2, 3\}$ and the communication graph (N, A) is represented in Fig. 4. Then, using expression (29), it is found that

$$\Gamma_{P_T}(L) = \begin{cases} 1 & \text{if } L \in \{\{a,b\},\{a,c\},\{b,c\}\}, \\ -1 & \text{if } L = \{a,b,c\}, \\ 0 & \text{else.} \end{cases}$$

Consequently,

$$\pi_1(N,p_T,A) = \frac{1}{2}\sum_{L:a\in L}\frac{\Gamma_v(L)}{|L|} = \frac{1}{2}\left(\frac{1}{2}+\frac{1}{2}-\frac{1}{3}\right) = \frac{1}{3}.$$

Similarly, we find that

$$\pi_2(N,p_T,A) = \pi_3(N,p_T,A) = \tfrac{1}{3}, \quad \pi_4(N,p_T,A) = 1, \quad \text{and} \quad \pi_5(N,p_T,A) = 0.$$

Furthermore, with some work, we obtain

$$\mu(N,v,A) = (\tfrac{5}{12},\tfrac{5}{12},\tfrac{5}{12},\tfrac{9}{12},0).$$

(iii) *Quadratic measure games*. Let $\omega = (\omega_1, \cdots, \omega_n)$ be a nonnegative (weight-) vector. Then the *quadratic measure game* $q_\omega \in G_0^N$ corresponding to ω (cf. Owen (1986)) is defined by

$$q_\omega(S) := \left(\sum_{i\in S}\omega_i\right)^2 - \sum_{i\in S}\omega_i^2 = \sum_{\{i,j\}\subset S: i\neq j} 2\omega_i\omega_j.$$

So in the quadratic measure game corresponding to ω, the worth of a coalition is completely determined by the worth of its various two-person subcoalitions, which, in turn, are completely determined by the product of the weights attached to each of its two players. It is easily seen that

$$\Delta_{q_\omega}(S) = \begin{cases} 2\omega_i\omega_j & \text{if } S = \{i,j\} \text{ with } i \neq j, \\ 0 & \text{else.} \end{cases}$$

Let us now consider a fixed communication situation $(N, q_\omega, A) \in CS_*^N$. Let $L \subset A$ be a subset of arcs. Then Theorem 4.1 implies that the dividend $\Gamma_{q_\omega}(L)$ is given by

$$\Gamma_{q_\omega}(L) = \begin{cases} 2\omega_s\omega_t & \text{if there are } s,t\in N \text{ such that } A(H(\{s,t\})) = L, \\ 0 & \text{else.} \end{cases}$$

In other words, the dividend $\Gamma_{q_\omega}(L) = 0$, unless L establishes a path in the graph (N, A). If this is a path from player s to player t, then the dividend $\Gamma_{q_\omega}(L)$ equals the worth of the coalition $\{s, t\}$ in the quadratic measure game.

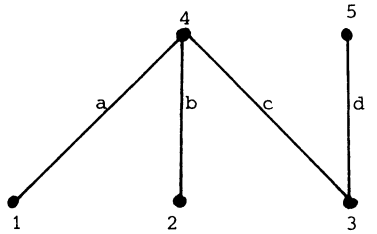


FIG. 4

We now introduce the following definition. Let $s, t \in N$. If $H(\{s, t\}) \neq \varnothing$, so if there exists a path from s to t in the (cycle-free) graph (N, A), then the *distance* $\delta(s, t)$ between s and t is defined by $\delta(s, t) := |A(H(\{s, t\}))|$, which represents the number of arcs supporting the path from s to t. Note that if $L \subset A$ establishes the path from player s to player t, we have that $|L| = \delta(s, t)$.

Computation of the position value will be based on the observation that

$$\Phi_a(A, r_N^{q_\omega}) = \sum_{\{s,t\} \subset N : a \in A(H(\{s,t\}))} \frac{2\omega_s \omega_t}{\delta(s,t)} \tag{31}$$

for all $a \in A$. Note that to apply (31), it is necessary to find all paths supported by a given arc. This can be done in a rather smooth way using (weighted) generating functions.

Let $a = \{i, j\} \in A$ be a fixed arc. If we were to cut this arc, then the component $C \in N/A$ with $i, j \in C$ would split up into two parts, say, C_i and C_j with $i \in C_i$ and $j \in C_j$. The *weighted generating function* θ_i^a describes the weighted number of points in C_i lying at a given distance from the point i. Formally,

$$\theta_i^a(x) = \sum_{k=0}^{d_i} \left(\sum_{s \in C_i : d(s,i) = k} \omega_s \right) x^k, \tag{32}$$

where $d_i := \max_{s \in C_i} d(s, i)$ is the maximum distance between a point in C_i and i. Similarly, we define d_j and θ_j^a.

The following theorem shows that it is possible to rewrite (31) in terms of generating functions.

THEOREM 5.1. *Let* $(N, q_\omega, A) \in CS_*^N$ *and* $a = \{i, j\} \in A$. *Let* θ_i^a *and* θ_j^a *be as in* (32). *Then*

$$\Phi_a(A, r_N^{q_\omega}) = 2 \int_0^1 \theta_i^a(x) \theta_j^a(x) dx. \tag{33}$$

Proof. Note that

$$\begin{aligned} 2 \int_0^1 \theta_i^a(x) \theta_j^a(x) dx &= \sum_{k=0}^{d_i + d_j} \sum_{s \in C_i, t \in C_j : d(s,i) + d(t,j) = k} \int_0^1 2\omega_s \omega_t x^k dx \\ &= \sum_{k=0}^{d_i + d_j} \sum_{s \in C_i, t \in C_j : d(s,t) = k+1} \frac{2\omega_s \omega_t}{\delta(s,t)} \\ &\overset{[(\text{cf. } (31))]}{=} \Phi_a(A, r_N^{q_\omega}). \end{aligned}$$

□

The above concepts and results are illustrated in the following example.

Example 5.2. Consider the nine-person communication situation (N, q_ω, A) with $\omega_i = 1$ for all $i \in N$ and the communication graph (N, A) of Fig. 5.

It is found that

$$\begin{aligned} \theta_1^a(x) &= 1 + 2x + 2x^2, & \theta_2^a(x) &= 1 + 2x + x^2, \\ \theta_1^b(x) &= 1 + 2x + 2x^2 + x^3, & \theta_3^b(x) &= 1 + 2x, \\ \theta_1^c(x) &= 1 + 2x + 4x^2 + x^3, & \theta_4^c(x) &= 1, \end{aligned}$$

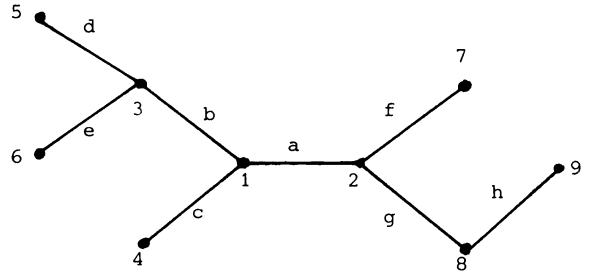


FIG. 5

and so

$$\pi_1(N,q_\omega,A)=\int_0^1 \theta_1^q(x)\theta_2^q(x)dx+\int_0^1 \theta_1^b(x)\theta_3^b(x)dx+\int_0^1 \theta_1^c(x)\theta_4^c(x)dx$$
$$=7\tfrac{7}{30}+6\tfrac{13}{20}+3\tfrac{7}{12}=17\tfrac{7}{15}.$$

In a similar way, it is possible to compute the position value for the other players.

6. Final remarks. (i) An open problem is how to characterize the position value axiomatically for the class CS^N of all communications situations. Furthermore, it would be interesting to find an axiomatic characterization of the position value for the class of all communication situations (N, v, A) with a full communication graph (N, A), i.e., $A = \{\{i,j\} \in 2^N \mid i \neq j\}$, because this class corresponds to the class of all (zero-normalized) games in coalitional form (cf. Example 2.2).

(ii) Having characterized the position value for the class of communication situations in which the communication graph contains no cycles, we might think of extending this concept to general communication situations by using *spanning trees*.

Let $(N, v, A) \in CS^N$. For each component $C \in N/A$, we consider the connected subgraph $(C, A(C))$ and the corresponding set $\mathcal{T}(C)$ of spanning trees, i.e.,

$$\mathcal{T}(C):=\{(C,L) \mid L \subset A(C),(C,L) \text{ is a tree}\}$$

and define an allocation rule $\rho : CS^N \to \mathbb{R}^N$ by

$$\rho_i(N,v,A)=\frac{1}{|\mathcal{T}(C_i)|}\sum_{L\subset A:(C_i,L)\in\mathcal{T}(C_i)}\pi_i(N,v,L)$$

for all $i \in N$, where $C_i \in N/A$ is the component to which player i belongs. It is clear that ρ equals π on the class CS^N_* of cycle-free communication situations. However, the following example shows that $\rho \neq \pi$.

Example 6.1. Consider the four-person communication situation (N, v, A) in which $v = u_{\{1,2,3\}}$ and (N, A) is as described in Fig. 6. There are three spanning trees corresponding

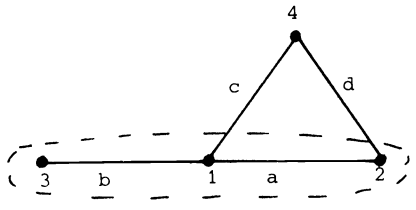


FIG. 6

to $L_1 = \{a, b, c\}$, $L_2 = \{a, b, d\}$, and $L_3 = \{b, c, d\}$, respectively. Then (e.g., using (18)) it is found that

$$\pi(N, v, L_1) = \pi(N, v, L_2) = (\tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}, 0), \qquad \pi(N, v, L_3) = (\tfrac{1}{3}, \tfrac{1}{6}, \tfrac{1}{6}, \tfrac{1}{3}),$$

and so $\rho(N, v, A) = (\frac{4}{9}, \frac{2}{9}, \frac{2}{9}, \frac{1}{9})$, while some calculation shows that $\pi(N, v, a) = (\frac{11}{24}, \frac{4}{24}, \frac{7}{24}, \frac{2}{24})$.

REFERENCES

R. J. AUMANN AND R. B. MYERSON (1988), *Endogenous formation of links between players and of coalitions: an application of the Shapley value*, in The Shapley Value, A. E. Roth, ed., Cambridge University Press, Cambridge, UK, pp. 175–191.

J. C. HARSANYI (1959), *A bargaining model for the cooperative n-person game*, in Contributions to the Theory of Games IV, A. W. Tucker and R. D. Luce, eds., Ann. Math. Studies 40, Princeton University Press, Princeton, NJ, pp. 325–356.

R. MEESSEN (1988), *Communication games*, Master's thesis, Dept. of Mathematics, University of Nijmegen, the Netherlands. (In Dutch.)

R. B. MYERSON (1977), *Graphs and cooperation in games*, Math. Oper. Res., 2, pp. 225–229.

——— (1980), *Conference structures and fair allocation rules*, Internat. J. Game Theory, 9, pp. 169–182.

G. OWEN (1986), *Values of graph-restricted games*, SIAM J. Algebraic Discrete Meth., 7, pp. 210–220.

E. C. ROSENTHAL (1988a), *Communication and its cost in graph-restricted games*, Theory and Decision, 25, pp. 275–286.

——— (1988b), *Communication networks and their role in cooperative games*, Social Networks, 10, pp. 255–263.

L. S. SHAPLEY (1953), *A value for n-person games*, in Contributions to the Theory of Games II, A. W. Tucker and H. W. Kuhn, eds., Ann. Math. Studies 28, Princeton University Press, Princeton, NJ, pp. 307–317.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 321–337, August 1992
© 1992 Society for Industrial and Applied Mathematics
002

POLYHEDRA OF THE EQUIVALENT SUBGRAPH PROBLEM AND SOME EDGE CONNECTIVITY PROBLEMS*

SUNIL CHOPRA†

Abstract. In this paper the problem of finding a minimum weight equivalent subgraph of a directed graph is considered. The associated equivalent subgraph polyhedron $P(G)$ is studied. Several families of facet-defining inequalities are described for this polyhedron. A related problem of designing networks that satisfy certain survivability conditions, as introduced in [M. Grötschel and C. L. Monma, *SIAM Journal on Discrete Mathematics*, 3 (1990), pp. 502–523] is also studied. The low connectivity case is formulated on directed graphs, and the directed formulation is shown to give a better LP-relaxation than the undirected one. It is shown how facet-defining inequalities of $P(G)$ give facet-defining inequalities in this case. Computational results are presented for some randomly generated problems.

Key words. minimum weight equivalent subgraphs (MWES), polyhedron, facets, connectivity

AMS(MOS) subject classifications. 05C40, 90B12, 90C27

1. Introduction. Given a directed graph $G = (V, A)$, a subgraph $ES = (V, F)$ is said to be an equivalent subgraph of G if ES has a directed path from vertex u to v if and only if G does. In this paper, we consider the problem of finding a minimum weight equivalent subgraph of a directed graph (MWES) and some related problems. This problem will be referred to as MWESP in the remainder of the paper. It can be stated as follows: Given a directed graph $G = (V, A)$ with arc weights c_a for all arcs a in A, find a MWES $ES = (V, F)$.

This problem is known to be NP-hard, in general (see Garey and Johnson [9]). Moyles and Thompson [17] and Hsu [13] have shown that it is sufficient to consider the case where G is a strongly connected directed graph. Throughout the paper, we assume that G is strongly connected; i.e., for every pair of nodes $u, v \in V$, there exist a directed path in G from u to v, and one from v to u. Relatively little has been done on this problem. In addition to the papers mentioned above, Martello and Toth [16] use a reduction procedure and branch and bound to solve the problem. They present some computational results. Richey, Parker, and Rardin [21] show that MWESP can be solved in linear time if G is a series-parallel graph.

An important related problem has been studied by Grötschel, Monma, and Stoer [11] in the context of network survivability. In an earlier paper [10], they describe several models of network survivability. All of their models are on undirected graphs. We show how the edge connectivity models of low connectivity (connectivity requirements of 0, 1, or 2) can be transformed to the directed case. We also show that the directed formulation is, in general, superior to the undirected formulation. It is shown how facets of the equivalent subgraph polyhedron can be transformed to obtain facets in this case. Some computational results are presented.

Another related problem is one of finding a 2-edge-connected subgraph of an undirected graph. This has been studied by Mahjoub [15], who gives families of facet-defining inequalities and a complete inequality description of the corresponding polytope when the graph is series-parallel.

In § 2 we give the basic notation used in the paper. In § 3 we give an integer programming formulation for MWESP and consider the equivalent subgraph polyhedron

* Received by the editors August 6, 1990; accepted for publication (in revised form) April 30, 1991.

† Department of Managerial Economics and Decision Sciences, J. L. Kellogg Graduate School of Management, Northwestern University, Evanston, Illinois 60208.

$P(G)$. Several families of facet-defining inequalities are given. In § 4 we develop the relationship between the low-edge connectivity model of Grötschel and Monma [10] and MWESP. Some computational results showing the performance of our formulation are given.

2. Notation and background. $G = (V, A)$ will refer to a directed graph with node set V and arc set A. The arc a directed from node u to node v will be referred to as a or (u, v). The *indegree* of node u is the number of arcs of the form (v, u) in A. The *outdegree* of u is the number of arcs of the form (u, v).

Let n be the number of arcs in G, i.e., $n = |A|$. Given an equivalent subgraph $ES = (V, F)$, define its incidence vector $x(ES)$, where $x \in R^n$ and

$$x(a) = \begin{cases} 1 & \text{if } a \in F, \\ 0 & \text{otherwise.} \end{cases}$$

The elements in x are indexed by the arc set A. Given any vector w indexed by the arc set A and an arc $a = (u, v)$, the corresponding element of w will be referred to as w_a, $w(a)$, or $w(u, v)$.

$UG = (V, E)$ will refer to an undirected graph with node set V and edge set E. The undirected edge between u and v will be referred to as $[u, v]$ or $e[u, v]$. An element of a vector w indexed by the edge set E will be referred to as w_e or $w[u, v]$.

Given a directed graph $G = (V, A)$, we can construct the corresponding undirected graph $UG = (V, E)$, where edge $[u, v] \in E$ if either $(u, v) \in A$ or $(v, u) \in A$.

We give two basic operations of contraction and deletion in directed graphs. Similar operations also hold in the undirected case. Given a directed graph $G = (V, A)$ and an arc $a = (u, v)$, *contracting* arc a consists of identifying the nodes u and v into the node u and deleting arc a to get $G_c = (V_c, A_c)$, where $V_c = V - \{v\}$ and $A_c = A - \{a\}$. All arcs that were incident to either u or v in G are now incident to the combined node u. If both arcs $a_1 = (u, v)$ and $a_2 = (v, u)$ are present, then $A_c = A - \{a_1, a_2\}$. On *deleting* arc a, we get the graph $G_d = (V, A_d)$, where $A_d = A - \{a\}$. The reverse operations are described as *expansion* and *extension*, respectively.

We assume familiarity with basic definitions in polyhedral theory (see, for instance, Bachem and Grötschel [1]).

3. The minimum equivalent subgraph polyhedron. Consider a directed graph $G = (V, A)$ with arc weights $c_a \in R$. Define the arc set $A^- = \{a \in A \mid c_a < 0\}$. Consider the arc weights $\bar{c}$, where

$$\bar{c}_a = \begin{cases} c_a & \text{for } a \in A - A^-, \\ 0 & \text{for } a \in A^-. \end{cases}$$

Let $\bar{E}\bar{S} = (V, \bar{F})$ be the MWES of G with arc weights $\bar{c}$. We can verify that $ES = (V, F)$, where $F = \bar{F} \cup A^-$ is a MWES of G with arc weights c. Thus we can assume that all arc weights are nonnegative, i.e., $c_a \in R_+$ for $a \in A$. As mentioned earlier, we also assume G to be strongly connected. The MWESP in this case is to find a minimum strongly connected subgraph of G.

Given $n = |A|$, define the equivalent subgraph polyhedron

$$P(G) = \text{conv}\{x \mid x \text{ is the incidence vector of an equivalent subgraph of } G\} + R_+^n.$$

Since $c_a \in R_+$ for $a \in A$, we can solve MWES by solving the following LP:

$$\min \sum c_a x_a \qquad \text{s.t. } x \in P(G).$$

It is clear that $P(G)$ is full-dimensional. Throughout this section, we describe several families of facet-defining inequalities for $P(G)$. Given a proper subset of nodes $V_1 \subset V$, define the arcs

$$A(V_1, V-V_1) = \{(u,v) \in A \mid u \in V_1, v \in V - V_1\}.$$

The arcs in $A(V_1, V - V_1)$ form a *directed cut*. Every equivalent subgraph of G must contain at least one arc from each directed cut. Thus the *cut inequality*

$$\sum_{a \in A(V_1, V-V_1)} x_a \geqq 1 \tag{3.1}$$

is valid for $P(G)$ for all $V_1 \subset V$, $1 \leqq |V_1| \leqq |V| - 1$. Define the polyhedron

$$\mathrm{LP}(G) = \{x \in R_+^n \mid x \text{ satisfies (3.1) for all } V_1 \subset V\}.$$

An equivalent subgraph $ES = (V, F)$ is *minimal* if there does not exist a subset $\bar{F} \subset F$, $\bar{F} \neq F$, such that $(V, \bar{F})$ is also an equivalent subgraph of G. Let $\mathscr{E}$ be the set of all minimal equivalent subgraphs of G and C the set of all minimal directed cuts. Given a finite set T, a *clutter* $\mathscr{F}$ is a family of subsets of T such that no member of $\mathscr{F}$ contains another member of $\mathscr{F}$. Let $\mathscr{F}_B$ be the family of all minimal subsets of T having a nonempty intersection with each member of $\mathscr{F}$. $\mathscr{F}_B$ is called the *blocking clutter* or, simply, the blocker of $\mathscr{F}$.

PROPOSITION 3.1. *$\mathscr{E}$ and C form a pair of blocking clutters.*

Proof. Clearly, both $\mathscr{E}$ and C are clutters since their members are minimal. Minimal directed cuts are minimal with respect to the property that they have a nonempty intersection with each equivalent subgraph, mutatis mutandis. The result thus follows. □

From the results of Fulkerson [8] on blocking polyhedra, the following result holds.

PROPOSITION 3.2. *The incidence vector of each minimal equivalent subgraph is a vertex of* LP (G), *and*

$$P(G) = \mathrm{conv}\{x \in \mathrm{LP}(G), x \text{ integer}\}.$$

Proposition 3.2 shows that MWES can be formulated as follows:

$$\min \sum c_a x_a \qquad \text{s.t. } x \in \mathrm{LP}(G), x \text{ integer}.$$

From the results of Fulkerson [8], we can also show the following theorem.

THEOREM 3.1. *The cut inequality* (3.1) *is facet-defining for $P(G)$ if and only if the cut $A(V_1, V - V_1)$ is minimal.*

In Chopra [3] it is shown that LP $(G) = P(G)$ if G is series-parallel.

We have previously defined the operations of contraction and deletion. Given $G = (V, A)$, assume that arcs $a_1 = (u, v)$ and $a_2 = (v, u)$ are present. Contract a_1 (or a_2) to get $G_c = (V_c, A_c)$, where $V_c = V - \{v\}$ and $A_c = A - \{a_1, a_2\}$. Consider any inequality

$$\pi^c x^c \geqq \pi_0 \tag{3.2}$$

that is facet-defining for $P(G_c)$. Define the inequality

$$\pi x \geqq \pi_0, \tag{3.3}$$

where

$$\pi_a = \begin{cases} \pi^c(a) & \text{if } a \in A_c, \\ 0 & \text{if } a \in \{a_1, a_2\}. \end{cases}$$

THEOREM 3.2. *Inequality* (3.3) *is facet-defining for* $P(G)$.

Proof. Let $ES = (V, F)$ be any equivalent subgraph of G. Then $ES_c = (V_c, F_c)$, where $F_c = F - \{a_1, a_2\}$ is an equivalent subgraph of G_c. Thus the inequality (3.3) is valid for $P(G)$. Furthermore, if the incidence vector $x(ES_c)$ satisfies (3.2) with equality, then $x(ES)$ satisfies (3.3) with equality. Since (3.2) is facet-defining for $P(G_c)$, there are $|A_c|$ independent equality solutions to (3.2). Each can be extended to an equality solution of (3.3) by adding the arcs a_1 and a_2. Let $\bar{x}$ be any one such extension. Define $\bar{x}_1$ and $\bar{x}_2$, where

$$\bar{x}_i(a) = \begin{cases} \bar{x}(a) & \text{if } a \in A - \{a_i\}, \quad i = 1,2, \\ \bar{x}(a) + 1 & \text{if } a = a_i, \quad i = 1,2. \end{cases}$$

This gives a set of $|A|$ independent equality solutions to (3.3). The result thus follows. □

Given $G = (V, A)$, assume that $a_1 = (u, v)$ and $a_2 = (u, v)$ are parallel arcs. Delete a_2 to get $G_d = (V, A_d)$. Consider any inequality

$$\pi^d x^d \geqq \pi_0 \tag{3.4}$$

that is facet-defining for $P(G_d)$. Define the inequality

$$\pi x \geqq \pi_0, \tag{3.5}$$

where

$$\pi_a = \begin{cases} \pi^d(a) & \text{if } a \in A_d, \\ \pi^d(a_1) & \text{if } a = a_2. \end{cases}$$

THEOREM 3.3. *Inequality* (3.5) *is facet-defining for* $P(G)$.

Proof. Let $ES = (V, F)$ be any equivalent subgraph of G. If $a_2 \notin F$, then ES is also an equivalent subgraph of G. If $a_2 \in F$, then $\bar{E}\bar{S} = (V, \bar{F})$ is also an equivalent subgraph of G, where $\bar{F} = F \cup \{a_1\} - \{a_2\}$. Thus (3.5) is valid for $P(G)$. Since (3.4) is facet-defining for $P(G_d)$, there are $|A_d|$ independent equality solutions to (3.4). Let $\bar{x}$ be any equality solution with $\bar{x}(a_1) = 1$. There must be at least one such solution. Define $\tilde{x}$, where

$$\tilde{x}(a) = \begin{cases} \bar{x}(a) & \text{if } a \in A_d - \{a_1\}, \\ 0 & \text{if } a = a_1, \\ 1 & \text{if } a = a_2. \end{cases}$$

This gives a set of $|A|$ independent equality solutions to (3.5). The result thus follows. □

$G' = (V', A')$ is a *minor* of G if G' is obtained from G by a sequence of contractions and deletions. G' is a *contraction minor* of G if the only deletions allowed are on parallel arcs. Theorems 3.2 and 3.3 allow us to lift any facet of $P(G')$ to a facet of $P(G)$ if G' is a contraction minor of G.

3.1. Wheelbarrow and bicycle inequalities. In this section we describe two classes of facet-defining inequalities on small graphs that can be lifted to larger graphs using Theorems 3.2 and 3.3. The names for these inequalities are borrowed from the travelling salesman problem literature (see Cornuejols, Fonlupt, and Naddef [7]), since similar structures arise there, also.

For any odd integer $k \geqq 3$, define the *k-wheelbarrow* $BW_k = (V, A_k)$, where

$$V = \{Y\} \cup \{u_j^i \mid i \in \{1,2,\cdots,k\}, j \in \{1,2\}\}.$$

For ease of notation, we refer to Y as u_0^i for $i \in \{1, 2, \cdots, k\}$. The arc set A_k is defined as

$$A_k = B_1 \cup B_2, \quad \text{where}$$

$$B_1 = \{(u_j^i, u_{j+1}^i), (u_{j+1}^i, u_j^i) \mid i \in \{1,2,\cdots,k\}, j \in \{0,1\}\},$$

$$B_2 = \{(u_2^i, u_2^{i+1}), (u_2^{i+1}, u_2^i) \mid i \in \{1,2,\cdots,k\}\}.$$

All indices are modulo k. Define $C_i = \{(u_0^i, u_1^i), (u_2^i, u_1^i)\}$ and $B = (\cup_i C_i) \cup B_2$ for $i \in \{1, 2, \cdots, k\}$. The 3-wheelbarrow BW_3 is as shown in Fig. 3.1. The arcs in B are shown with solid lines, and those in $A_k - B$ with broken lines.

On the k-wheelbarrow BW_k, define the *wheelbarrow inequality*

$$\sum_{a \in B} x_a \geqq \lceil 3k/2 \rceil = (3k+1)/2. \tag{3.6}$$

LEMMA 3.1.1. *The wheelbarrow inequality* (3.6) *is valid for* $P(BW_k)$.

Proof. Note that at least one arc from each set C_i must be used in any equivalent subgraph. If r arcs from B_2 are used in an equivalent subgraph for $r \leqq \lfloor k/2 \rfloor$, then at least $2r + 2(k - 2r)$ other arcs must be present from C_i for $i \in \{1, 2, \cdots, k\}$. This gives a total of $2k - r$ arcs from B. Clearly, $2k - r \geqq \lceil 3k/2 \rceil$ for $r \leqq \lceil k/2 \rceil$. For $r \geqq \lceil k/2 \rceil + 1$, at least k other arcs must be present, one from each set C_i. Here $k + r \geqq \lceil 3k/2 \rceil$. The result thus follows. □

The following result is stated without proof. A detailed proof can be found in [4].

THEOREM 3.1.1. *The wheelbarrow inequality* (3.6) *is facet-defining for* $P(BW_k)$.

Remark 3.1.1. Consider a graph $G = (V, A)$ that contains BW_k as a subgraph, i.e., $A_k \subseteq A$. Suppose that $A - A_k$ contains the arc sets $\bar{B}$ and $\tilde{B}$, where $\bar{B} = \{(u_0^i, u_2^i), (u_2^i, u_0^i), (u_1^i, u_1^j), (u_1^i, u_2^j)\}$ and $\tilde{B} = \{(u_2^i, u_1^j)\}$ for all $i \neq j \in \{1, \cdots, k\}$. The inequality

$$\sum_{a \in B \cup \bar{B}} x_a + 2 \sum_{a \in \tilde{B}} x_a \geqq \lceil 3k/2 \rceil \tag{3.7}$$

is facet-defining for $P(G)$.

Similarly, for any odd integer k, we can define the *k-bicycle* configuration $BB_k = (V, A_k)$, where $V = \{u_j^i \mid i \in \{1, 2, \cdots, k\}, j \in \{1, 2\}\}$ and

$$A_k = \{(u_1^i, u_2^i), (u_2^i, u_1^i), (u_2^i, u_2^{i+1}), (u_2^{i+1}, u_2^i), (u_1^i, u_1^{i+1}), (u_1^{i+1}, u_1^i) \mid i \in \{1,2,\cdots,k\}\}.$$

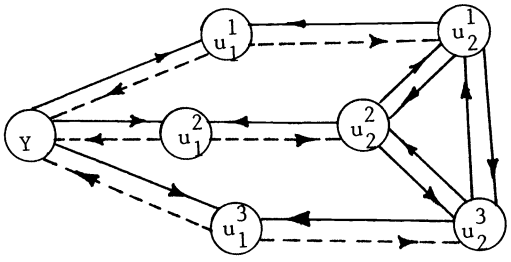


FIG. 3.1

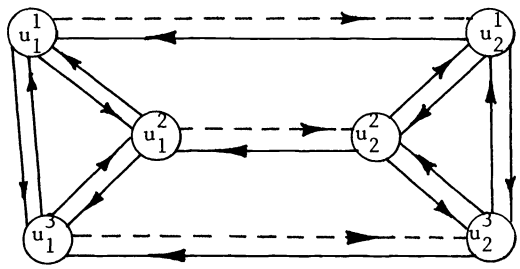


FIG. 3.2

Define the arc set

$$B = \{(u_2^i, u_1^i), (u_2^i, u_2^{i+1}), (u_2^{i+1}, u_2^i), (u_1^i, u_1^{i+1}), (u_1^{i+1}, u_1^i) \mid i \in \{1, 2, \cdots, k\}\}.$$

BB_3 is as shown in Fig. 3.2. The arcs in B are shown with solid lines, and those in $A_k - B$ with broken lines.

On the k-bicycle configuration BB_k, define the *bicycle inequality*

$$\sum_{a \in B} x_a \geqq \lceil 3k/2 \rceil. \tag{3.8}$$

LEMMA 3.1.2. *The bicycle inequality* (3.12) *is valid for* $P(BB_k)$.

Proof. BB_k is a subgraph of the graph obtained by adding all arcs (u_1^i, u_1^{i+1}), (u_1^{i+1}, u_1^i) to the k-wheelbarrow BW_k. Thus, with an argument similar to that in Cornuejols, Fonlupt, and Naddef [7], we can show that (3.12) is valid for $P(BB_k)$. □

The following result is stated without proof. A detailed proof can be found in [4].

THEOREM 3.1.2. *The bicycle inequality* (3.8) *is facet-defining for* $P(BB_k)$.

Remark 3.1.2. Consider a graph $G = (V, A)$ that contains BB_k as a subgraph. Define $\bar{B} = \{(u_1^i, u_2^j)\}$ and $\tilde{B} = \{(u_2^i, u_1^j)\}$ for all $i \neq j \in \{1, \cdots, k\}$. The inequality

$$\sum_{a \in B \cup \bar{B}} x_a + 2 \sum_{a \in \tilde{B}} x_a \geqq \lceil 3k/2 \rceil \tag{3.9}$$

is facet-defining for $P(G)$.

3.2. Crown inequalities. Given any positive integer k, define the *k-crown* configuration $CR_k = (V, A_k)$, where

$$V = \{u_i \mid i \in \{1, 2, \cdots, 4k\}\},$$

$$A_k = \{(u_i, u_{i+1}), (u_i, u_{i+2k}) \mid i \in \{1, 2, \cdots, 4k\}\}.$$

The arcs (u_i, u_{i+2k}) are called *diagonal* arcs. All indices are modulo $4k$ with $0 = 4k$. Define the arc set

$$B = \{(u_{i+2k}, u_i)(u_{i+1}, u_{i+2k+1}) \mid i \in \{1, 3, \cdots, 2k-1\}\}$$

$$\cup \{(u_i, u_{i+1}) \mid i \in \{2, 4, \cdots, 2k-2\} \cup \{2k+1, 2k+3, \cdots, 4k-1\}\}.$$

A similar configuration was introduced by Naddef and Rinaldi [18]. CR_2 is as shown in Fig. 3.3, with the arcs in B shown in broken lines.

On the k-crown configuration, define the *crown inequality*

$$\sum_{a \in A_k - B} x_a \geqq 2k+1. \tag{3.10}$$

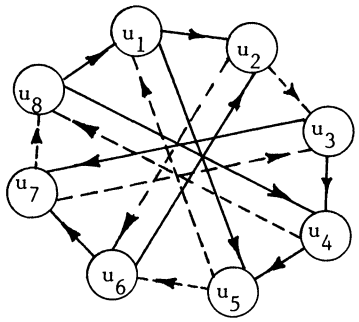


FIG. 3.3

LEMMA 3.2.1. *The crown inequality* (3.10) *is valid for* $P(CR_k)$.

Proof. The following proof was suggested by an anonymous referee. Define the sets $V_1 = \{1, 3, \cdots, 2k-1, 2k+2, 2k+4, \cdots, 4k\}$ and $V_2 = \{2, 4, \cdots, 2k, 2k+1, 2k+3, \cdots, 4k-1\}$. Consider the following cut inequalities:

$$x(A(\{u_i\}, V-\{u_i\})) \geqq 1 \quad \text{for } i \in V_1;$$

$$x(A(V-\{u_i\}, \{u_i\})) \geqq 1 \quad \text{for } i \in V_2;$$

$$x(A(V-\{u_1, u_{2k+1}\}, \{u_1, u_{2k+1}\})) \geqq 1.$$

Adding all the above inequalities, dividing the result by two, and rounding up the right side and the coefficients of the left side gives inequality (3.10). □

The following result is stated without proof. A detailed proof can be found in [4].

THEOREM 3.2.1. *The crown inequality* (3.17) *is facet-defining for* $P(CR_k)$ *for* $k \geqq 1$.

Remark 3.2.1. Consider a complete graph $G = (V, A)$ that contains CR_k as a spanning subgraph. Define the arc sets

$$B_1 = \{(u_i, u_j) | j \in V_2\} \cup \{(u_i, u_j) | \{i, u\} \subseteq V_1\} \cup \{(u_i, u_1) | i \in V_1 \cup V_2\}$$
$$\cup \{(u_i, u_{2k+1}) | i \in V_2\},$$
$$B_2 = \{(u_i, u_{2k+1}) | i \in V_1 - \{1\}\}.$$

Consider the inequality

$$\sum_{B_1 \cup A_k - B} x_a + 2 \sum_{a \in B_2} x_a \geqq 2k+1. \tag{3.11}$$

The validity of inequality (3.11) for $P(G)$ follows from the proof of Lemma 3.2.1. It can also be shown that inequality (3.11) is facet-defining for $P(G)$.

3.3. Odd wheel inequalities. For any odd integer $k \geqq 3$, consider the *k-wheel* configuration $OW_k = (V, A_k)$, where

$$V = \{u\} \cup \{u_i | i \in \{1, 2, \cdots, k\}\},$$
$$A_k = \{(u_i, u), (u, u_{i+1}) | i \in \{1, 3, \cdots, k-2\}\} \cup \{(u, u_k), (u_k, u)\}$$
$$\cup \{(u_i, u_{i+1}) | i \in \{1, 2, \cdots, k\}\} \cup \{(u_{i+1}, u_i) | i \in \{1, 3, \cdots, k-2\}\}.$$

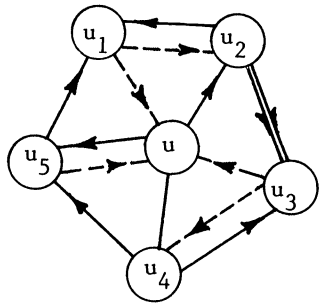


FIG. 3.4

All indices used are modulo k. Define the arc sets

$$B_1 = \{(u_i, u_{i+1}), (u_i, u) \mid i \in \{1, 3, \cdots, k-2\}\} \cup \{(u_k, u)\},$$

$$B_2 = \{(u_i, u_{i+1}) \mid i \in \{2, \cdots, k-3\}\}.$$

For $k = 3$, B_2 is empty. OW_5 is as shown in Fig. 3.4. Arcs in B_1 are shown with broken lines, and arcs in B_2 with double lines.

On the k-wheel configuration, define the *odd wheel inequality*

$$\sum_{a \in A_k - \{B_1 \cup B_2\}} x_a + 2 \sum_{a \in B_2} x_a \geqq k. \tag{3.12}$$

LEMMA 3.3.1. *The odd wheel inequality* (3.12) *is valid for* $P(OW_k)$.

Proof. Let $r = \lceil k/2 \rceil$. Consider the following cut inequalities with weights as shown:
$x(V - \{u_{2j-1}\}, \{u_{2j-1}\}) \geqq 1$ with a weight of j for $j \in \{1, \cdots, r-1\}$;
$x(\{u_{2j}\}, V - \{u_{2j}\}) \geqq 1$ with a weight of $r - j$ for $j \in \{1, \cdots, r-1\}$;
$x(V - \{u_j, u_{j+1}\}, \{u_j, u_{j+1}\}) \geqq 1$ with a weight of $r - 1$ for
$j \in \{1, 2, \cdots, k-2\}$;
$x(V - \{u_k\}, \{u_k\}) \geqq 1$ with a weight of $r - 1$;
$x(\{u\}, V - \{u\}) \geqq 1$ with a weight of 1.
Adding all the above inequalities, dividing the result by r, and rounding up the right side and the coefficients on the left side gives the inequality (3.12). □

The following result is stated without proof. A detailed proof can be found in [4].

THEOREM 3.3.1. *The odd wheel inequality* (3.12) *is facet-defining for* $P(OW_k)$, $k \geqq 3$.

Remark 3.3.1. Let $G = (V, A)$ be a complete graph that contains OW_k as a spanning subgraph. The proof of Lemma 3.3.1 allows us to give coefficients for an inequality valid for $P(G)$, whose restriction to arcs in A_k is (3.12).

3.4. Composition of facets. Consider two graphs, $G_1 = (V_1, A_1)$ and $G_2 = (V_2, A_2)$. Consider $\{u_i, v_i\} \subseteq V_i$, $i = 1, 2$ and assume that $\{(u_i, v_i), (v_i, u_i)\} \subseteq A_i$, $i = 1, 2$. Assume that the inequality

$$\pi^i x^i \geqq \pi_0^i, \qquad i = 1, 2 \tag{3.13}$$

is facet-defining for $P(G_i)$ such that

$$\pi(u_i, v_i) = 0, \qquad i = 1, 2;$$

$$\pi(v_i, u_i) = \alpha, \qquad i = 1, 2.$$

Note that all the facet-defining inequalities previously discussed have nodes $\{u, v\}$ for which the above condition holds. On the other hand, the same facets provide examples to show that the above condition need not hold for every pair of nodes u and v for which both arcs (u, v) and (v, u) are present.

Compose G_1 and G_2 by identifying u_1 and u_2 into u, and v_1 and v_2 into v, to get $G = (V, A)$. Here $A = A_1 \cup A_2 \cup \{(v, u)\} - \{(v_1, u_1), (v_2, u_2)\}$. The graphs G_1, G_2, and G are shown in Fig. 3.5.

Define the inequality $\pi x \geqq \pi_0$, where

$$\pi_a = \begin{cases} \pi_a^i & \text{for } a \in A_i - \{(v_i, u_i)\}, \quad i = 1, 2, \\ \alpha & \text{for } a = (v, u); \end{cases} \tag{3.14}$$

$$\pi_0 = \pi_0^1 + \pi_0^2 - \alpha.$$

THEOREM 3.4.1. *Inequality* (3.14) *is facet-defining for* $P(G)$.

Proof. First, we prove that inequality (3.14) is valid for $P(G)$. Let $ES = (V, F)$ be any equivalent subgraph of G. Let $F(1)$ and $F(2)$ be the restriction of F to A_1 and A_2, respectively. Define $E_i = (V_i, F(i))$ for $i = 1, 2$. Note that if $(v, u) \in F$, then

$$\pi x(ES) + \alpha = \pi^1 x^1(E_1) + \pi^2 x^2(E_2), \tag{3.15}$$

and, if $(v, u) \notin F$, then

$$\pi x(ES) = \pi^1 x^1(E_1) + \pi^2 x^2(E_2). \tag{3.16}$$

Note that at least one of the graphs E_i, $i = 1, 2$, has a directed path from v_i to u_i. There are two possible cases.

Case A. E_i contains a directed path from v_i to u_i for $i = 1, 2$. In this case, $ES_i = (V_i, F(i) \cup \{(u_i, v_i)\})$ is an equivalent subgraph in G_i, $i = 1, 2$. Thus

$$\pi^i x^i(ES_i) \geqq \pi_0^i \quad \text{for } i = 1, 2.$$

From (3.15) and (3.16), it thus follows that $x(ES)$ satisfies (3.14).

Case B. E_1 contains a directed path from v_1 to u_1, while E_2 does not contain a path from v_2 to u_2. Define $ES_2 = (V_2, F(2) \cup \{(u_2, v_2), (v_2, u_2)\})$. $ES_1 = (V_1, F(1) \cup \{(u_1, v_1)\})$ is an equivalent subgraph of G_1, and ES_2 is an equivalent subgraph of G_2. Thus

$$\pi^1 x^1(ES_1) \geqq \pi_0^1; \qquad \pi^2 x^2(ES_2) + \alpha \geqq \pi_0^2.$$

From (3.16) it thus follows that $x(ES)$ satisfies (3.14). Thus inequality (3.14) is valid for $P(G)$.

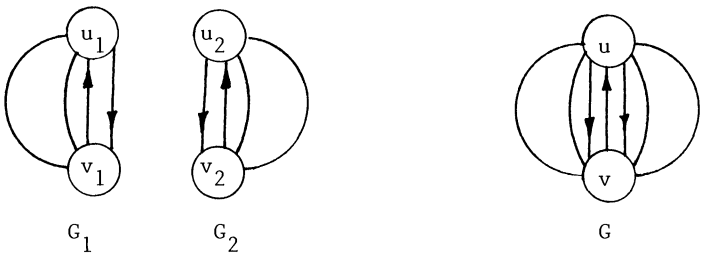


FIG. 3.5

Define $a_i = (v_i, u_i)$ for $i = 1, 2$. Since $\pi^i x^i \geqq \pi_0^i$ is facet-defining for $P(G_i)$, there is a nonsingular matrix C_i, each row of which satisfies $\pi^i x^i \geqq \pi_0^i$ with equality. Matrix C_i is as shown below for $i = 1, 2$:

$$C_i = \begin{vmatrix} A_i - \{a_i\} & a_i \\ D_i & 1 \\ B_i & 0 \end{vmatrix}.$$

D_i cannot be empty since there must be at least one equality solution to $\pi^i x^i \geqq \pi_0^i$ with support in a_i. Let d_i be a row of D_i, $i = 1, 2$. Define the following matrix C on the arc set A, where $a = (v, u)$:

$$C = \begin{vmatrix} A_1 - \{a_1\} & a & A_2 - \{a_2\} \\ \bar{d}_1 & 1 & D_2 \\ \bar{d}_1 & 0 & B_2 \\ D_1 & 1 & \bar{d}_2 \\ B_1 & 0 & \bar{d}_2 \end{vmatrix},$$

where $\bar{d}_i$, $i = 1, 2$ is a matrix with each row equal to d_i, $i = 1, 2$. Each row of C is a vector in $P(G)$ and satisfies (3.14) at equality. It is easy to verify that C has full column rank. Thus inequality (3.14) is facet-defining for $P(G)$. □

4. Network survivability and $P(G)$. In the context of network survivability, there are problems whose polyhedra are closely related to the equivalent subgraph polyhedron $P(G)$. Grötschel and Monma [10] introduced integer polyhedra associated with several network survivability problems. In [11], they give some polyhedral results related to these problems. We restate the edge survivability problem discussed by them. Consider an undirected graph $UG = (V, E)$ with nonnegative edge weights w_e for each edge e in E. w_e is the cost of establishing the link e. We assume that UG has no parallel edges, but several copies of a link may be established; i.e., an edge may be used more than once. For each node $w \in V$, a nonnegative integer r_v, called the *type* of v, is specified. For each pair of nodes $u, v \in V$, define $r_{uv} = \min\{r_u, r_v\}$. Define network $N = (V, F, M)$, where $F \subseteq E$ and $M(e)$ gives the number of copies, or *multiplicity*, of edge e in N. By $N(T)$, we denote the multigraph obtained by replicating every edge e as many times as its multiplicity in N. We say that N satisfies the *edge survivability* conditions if, for each pair $u, v \in V$ of distinct nodes, $N(T)$ contains at least $r_{u,v}$ edge disjoint paths from u to v. The weight of the network N is given by $\sum_{e \in F} M(e) w_e$. The problem is to find a minimum weight network N that satisfies the edge survivability conditions. Define the incidence vector $x(N)$, where

$$x_e(N) = \begin{cases} 0 & \text{for } e \in E - F, \\ M(e) & \text{for } e \in F. \end{cases}$$

Given the graph UG and the vector of node types r, define the polyhedron

$$\text{ECON}(UG; r) = \text{conv}\{x(N) \mid \text{network } N \text{ satisfies edge survivability conditions}\} + R_+^E.$$

We restrict our attention to the case where all node types are 0, 1, or 2. This is referred to as the *low connectivity* case in [11].

Given $W \subseteq V$, define

$$\text{con}(W) = \max\{r_{uv} \mid u \in W, v \in V - W\}$$

and the edge set

$$\delta(W) = \{[u,v] \in E \mid u \in W, v \in V - W\}.$$

The edge set $\delta(W)$ defines a cut of UG for $1 \leqq |W| \leqq |V| - 1$. Define the polyhedron

$$\text{LPECON}(UG;r) = \{x \in R_+^E \mid \sum_{e \in \delta(W)} x_e \geqq \text{con}(W) \quad \text{for all } W \subseteq V, 1 \leqq |W| \leqq |V| - 1\}.$$

Grötschel and Monma [10] show the following result.

PROPOSITION 4.1. *It holds that* ECON $(UG; r)$ = conv $\{x \in$ LPECON $(UG; r)$, x integer$\}$.

Thus a minimum weight network that satisfies the edge survivability conditions can be found by solving the following problem:

$$\min \sum w_e x_e \quad \text{s.t. } x \in \text{LPECON}(UG;r); x \text{ integer.} \tag{4.1}$$

For the low connectivity case, we can reformulate the problem. Given $UG = (V, E)$, define the directed graph $G = (V, A)$, where both arcs (u, v) and $(v, u) \in A$ if edge $[u, v] \in E$. The arcs in A are assigned weights c, where

$$c(u,v) = c(v,u) = w[u,v].$$

Let N_1 be the set of nodes of type 1, and N_2 the set of nodes of type 2. If $|N_2| \geqq 1$, any node $v_r \in N_2$ is declared the root; otherwise, any node $v_r \in N_1$ is declared the root. Let $ND = (V, B)$ be a subgraph of G such that there is a directed path in ND from v_r to each node in N_1 and, for every pair $u_1, u_2 \in N_2$, there is a directed path in ND from u_1 to u_2. Form the network $N = (V, F, M)$, where $e = [u, v] \in F$ if either (u, v) or $(v, u) \in B$. Furthermore,

$$M(e) = |\{(u,v),(v,u)\} \cap B|.$$

We can verify that N satisfies the edge survivability conditions. We say that ND also satisfies the *edge survivability* conditions. We can verify that, given a network $N = (V, F, M)$ that satisfies the edge survivability conditions, we can construct the corresponding directed network ND (see, for example, Lovasz [14]). Thus the problem of finding the minimum weight undirected network N on UG is equivalent to finding the minimum weight directed network ND on G. Define the incidence vector $y(ND)$, where

$$y_a(ND) = \begin{cases} 1 & \text{if } a \in B, \\ 0 & \text{otherwise.} \end{cases}$$

Given the directed graph $G = (V, A)$ with root v_r and the vector of node types r, define the polyhedron

$$\text{DECON}(G;r) = \text{conv}\{y(ND) \mid \text{network } ND \text{ satisfies edge survivability conditions}\} + R_+^A.$$

Given $W \subseteq V$, $1 \leqq |W| \leqq |V| - 1$, the directed cut $A(W, V - W)$ is as defined earlier. Consider the polyhedron

$$\text{LPDECON}(G;r) = \{y \in R_+^A \mid \sum_{a \in A(W, V-W)} y_a \geqq 1 \text{ for all } W \subseteq V \text{ such that}$$

$$|W \cap N_2| \geqq 1 \text{ and } |(V-W) \cap N_2| \geqq 1 \text{ or } v_r \in W \text{ and } |(V-W) \cap N_1| \geqq 1\}.$$

We can verify the following result.

PROPOSITION 4.2. *It holds that* DECON $(G; r)$ = conv $\{x \in$ LPDECON $(G; r)$, x integer$\}$.

Proof. Let x be an integer vertex of LPDECON $(G; r)$. From Fulkerson [8], it follows that x is a 0, 1 vector. Define $B = \{a \in A \mid x_a = 1\}$. $ND = (V, B)$ has a directed path from the root to every node in N_1 and a directed path between every pair of nodes in N_2; otherwise, one of the cut inequalities would be violated. Thus ND satisfies the edge survivability conditions.

Conversely, let $ND = (V, B)$ be a minimal network satisfying the survivability conditions, i.e., there does not exist $\bar{B} \subseteq B$, $\bar{B} \neq B$ such that $\bar{N}\bar{D} = (V, \bar{B})$ satisfies the edge survivability conditions. Thus, for each arc $\bar{a} \in B$, there exists a valid directed cut inequality

$$\sum_{a \in A(W, V-W)} y_a \geqq 1 \tag{4.2}$$

such that $A(W, V - W) \cap B = \{\bar{a}\}$. Consider the incidence vector $y(ND)$. Clearly, $y(ND) \in$ LPDECON $(G; r)$. If $y(ND)$ is not a vertex of LPDECON $(G; r)$, then $y(ND) = (y^1 + y^2)/2$, where $y^1, y^2 \in$ LPDECON (G, r). If $y_a(ND) = 0$, then $y_a^1 = y_a^2 = 0$. If $y^1 \neq y^2$, there exists an arc $\tilde{a}$ such that $y_{\tilde{a}}(ND) = 1$, $y_{\tilde{a}}^1 < 1$ and $y_{\tilde{a}}^2 > 1$. y^1 violates the directed cut inequality (4.1) for which $A(W, V - W) \cap B = \{\tilde{a}\}$. This shows that $y^1 = y^2$; i.e., $y(ND)$ is a vertex of LPDECON $(G; r)$. The result thus follows. □

Thus, in the low connectivity case, we can find a minimum weight directed network that satisfies the edge survivability conditions by solving the following problem:

$$\min \sum c_a y_a \qquad \text{s.t. } y \in \text{LPDECON}(G;r), y \text{ integer.} \tag{4.3}$$

For an undirected graph, we can use formulations (4.1) or (4.3) to find the minimum weight network.

For ECON $(UG; r)$, Grötschel, Monma, and Stoer [11] gave a class of facet-defining inequalities called the *partition inequalities*. We state the inequality as defined by them. A collection $W_1, \cdots, W_p$ of subsets of V is called a *proper partition* of V if

$$W_i \neq 0, \qquad i = 1, \cdots, p;$$

$$W_i \cap W_j \neq \varnothing, \qquad i \leqq i < j \leqq p;$$

$$\cup W_i = V;$$

$$|W_i \cap (N_1 \cup N_2)| \geqq 1, \qquad i = 1, 2, \cdots, p;$$

$$|W_i \cap N_2| \geqq 1 \text{ for at least two sets } W_i.$$

Let $EP = U_i \delta(W_i)$. The partition inequality induced by a proper partition is given by

$$\sum_{e \in EP} x_e \geqq p. \tag{4.4}$$

Reference [11] gives conditions under which these inequalities are facet-defining for ECON (UG; r). Define the polyhedron

$$\text{LPECON1}(UG;r) = \{x \in \text{LPECON}(UG;r) \mid x \text{ satisfies all partition inequalities (4.3)}\}.$$

The polyhedra ECON (UG; r) and DECON (G; r) are related by the following linear transformation:

$$\text{ECON}(UG;r) = \{x \mid x[u,v] = y(u,v) + y(v,u), \forall [u,v] \in E, y \in \text{DECON}(G;r)\}.$$

This linear transformation allows us to compare LPDECON1 and LPDECON using techniques of Balas and Pulleyblank [2] or Padberg and Tsung [19]. Using a procedure similar to that in Chopra and Rao [5], we can show the following result.

THEOREM 4.1. *It holds that*

$$\min\{\textstyle\sum w_e x_e \mid x \in \text{LPECON1}(UG;r)\} \geqq \min\{\textstyle\sum c_a y_a \mid y \in \text{LPDECON}(G;r)\}.$$

Proof. The details of the proof are as in Chopra and Rao [5].

It is easy to see that a directed cut inequality on projection gives the corresponding undirected cut inequality. Given the partition $W_1, \cdots, W_p$ described earlier, consider the directed cut inequalities

$$\sum_{a \in A(V - W_i, W_i)} y_a \geqq 1, \qquad i = 1, \cdots, p.$$

Since $|W_i \cap N_2| \geqq 1$ for at least two sets W_i, each of these inequalities is valid. The partition inequality is obtained as the sum of all the above inequalities on projection.

Thus the polyhedron obtained on projecting LPDECON (G; r) is contained in LPECON1 (UG; r). □

Theorem 4.1 shows that the LP-relaxation of the directed formulation (4.2) will, in general, give a better lower bound than the LP-relaxation of the undirected formulation. Also note that the directed cut inequalities can be identified in polynomial time, while identification of the partition inequalities is difficult. This suggests that bidirecting the graph and solving the resulting directed problem may be better, even though the number of variables is doubled. This makes the study of DECON (G; r) important.

Given a directed graph $G = (V, A)$, let $(W_i, i = 1, \cdots, p)$ be a partition of the node set such that $|W_i \cap N_2| \geqq 1$, $i = 1, \cdots, p$. Let $G(W_i)$ be the graph induced by the node set W_i. Let $\hat{G} = (\hat{V}, \hat{A})$ be the graph obtained by identifying each of the node sets W_i into a node w_i, i.e., $\hat{V} = \{w_i \mid i = 1, \cdots, p\}$ and $\hat{A} = \cup_i A(W_i, V - W_i)$. $\hat{G}$ must be strongly connected if any subgraph of G is to satisfy the connectivity requirements. Let $ND = (V, B)$ be a network that satisfies all edge connectivity requirements on G, and let $\hat{N} = (\hat{V}, \hat{B})$ be its restriction to $\hat{G}$. By the construction of $\hat{G}$, $\hat{N}$ must be strongly connected.

PROPOSITION 4.3. *Let $\hat{N} = (\hat{V}, \hat{B})$ be any equivalent subgraph of $\hat{G}$. It can be expanded to $ND = (V, B)$ such that ND satisfies all edge survivability requirements on G and $B \cap \hat{A} = \hat{B}$.*

Proof. Let $G_i = (W_i, A_i)$ be equivalent subgraphs of $G(W_i)$ for $i = 1, 2, \cdots, p$. Define $B = \hat{B} \cup (\cup_i A_i)$. $ND = (V, B)$ is an equivalent subgraph of G and thus satisfies all edge connectivity requirements. □

Let

$$\hat{\pi}\hat{y} \geqq \pi_0 \tag{4.5}$$

be any facet-defining inequality for the equivalent subgraph polyhedron $P(\hat{G})$. Define the inequality

$$\pi y \geqq \pi_0, \tag{4.6}$$

where

$$\pi_a = \begin{cases} \hat{\pi}_a & \text{for } a \in \hat{A}, \\ 0 & \text{for } a \in A - \hat{A}. \end{cases}$$

THEOREM 4.2. *Inequality* (4.5) *is facet-defining for* DECON $(G; r)$.

Proof. Let $ND = (V, B)$ be any network in G that satisfies the edge connectivity requirements, and $\hat{N} = (\hat{V}, \hat{B})$ its reduction to $\hat{G}$. As discussed earlier, $\hat{N}$ is a strongly connected graph and thus an equivalent subgraph of $\hat{G}$. Thus the incidence vector $\hat{y}(\hat{N})$ satisfies (4.4), which implies that $y(ND)$ satisfies (4.5).

To show that inequality (4.5) is facet-defining for DECON $(G; r)$, we exhibit $|A|$ linearly independent equality solutions to (4.5). Since (4.4) is facet-defining for $P(\hat{G})$, there are $|\hat{A}|$ linearly independent equality solutions to it. Define $G_i = (W_i, A_i)$ to be equivalent subgraphs of $G(W_i)$ for $i = 1, 2, \cdots, p$. Let $\hat{y}_1 \in P(\hat{G})$ be any equality solution to (4.4). Define y_1, where

$$y_1(a) = \begin{cases} \hat{y}_1(a) & \text{for } a \in \hat{A}, \\ 1 & \text{for } a \in \bigcup_i A_i, \\ 0 & \text{otherwise.} \end{cases}$$

$y_1 \in$ DECON $(G; r)$ and satisfies (4.5) at equality. This gives $|\hat{A}|$ equality solutions to (4.5). Choose any solution y_1 from this set. For each arc, $b \in A - \hat{A}$ define y_1^b, where

$$y_1^b(a) = \begin{cases} y_1(a) & \text{for } a \neq b, \\ y_1(a) + 1 & \text{for } a = b. \end{cases}$$

$y_1^b \in$ DECON $(G; r)$ and satisfies (4.5) at equality. This gives $|A|$ independent equality solutions to (4.5) and proves that it is facet-defining for DECON $(G; r)$. □

Remark 4.1. The polyhedron DECON $(G; r)$ is also closely related to the Steiner tree polyhedron STP. Chopra and Rao [5], [6] give several families of facet-defining inequalities in this case. Given a directed graph $G = (V, A)$, let $(W_i, i = 1, \cdots, p)$ be a partition of the node set V such that $N_2 \subseteq W_i$ for some i. Let $\hat{G} = (\hat{V}, \hat{A})$ be the graph obtained by identifying each of the node sets W_i into a node w_i. A node $w_i \in \hat{V}$ must be spanned if $|W_i \cap (N_1 \cup N_2)| \geqq 1$. Let

$$\hat{\pi}\hat{y} \geqq \pi_0 \tag{4.7}$$

be any facet-defining inequality for the Steiner tree polyhedron STP $(\hat{G})$. Define the inequality

$$\pi y \geqq \pi_0, \tag{4.8}$$

where

$$\pi_a = \begin{cases} \hat{\pi}_a & \text{for } a \in \hat{A}, \\ 0 & \text{for } a \in A - \hat{A}. \end{cases}$$

With an argument similar to that in the proof of Theorem 4.2, we can show that (4.7) is facet-defining for DECON $(G; r)$.

4.1. Computational results. In this section we give some computational results for low-edge connectivity survivable networks. The main purpose of our study is to show the effectiveness of the directed formulation (4.3). Grötschel, Monma, and Stoer [12] give encouraging computational results using the undirected formulation on some real-world problems. Since these problems are classified, we restricted ourselves to randomly generated problems.

The problems considered are of four types: (a) random-sparse; (b) Euclidean-sparse; (c) random-complete; (d) Euclidean-complete.

For (a) and (b), a connected-sparse graph is generated at random. For (a), integer edge weights are generated at random from a uniform distribution between 0 and 500. For (b), nodes are assigned randomly on a 500×500 grid, and edges are assigned weights equal to the distance between the endnodes. For (c) and (d), the graphs considered are complete, and edge weights are assigned as in (a) and (b), respectively. Nodes in N_1 and N_2 are generated at random, and one of the nodes in N_2 declared the root.

The initial problem solved is the following:

$$\min \sum c_a y_a \qquad \text{s.t. } y \in \text{LPDECON}(G; r). \tag{4.9}$$

Let LP be the optimal solution to (4.8). In the case where (4.8) has a fractional optimum, we go into branch and cut (see Padberg and Rinaldi [20] for a detailed description) to get the integer optimum OPT, i.e., the solution to (4.2). At each stage the only inequalities added are directed cuts (3.1), and these can easily be identified using an efficient max-flow algorithm. None of the other inequalities described in § 3 were included, since the main objective of this computational study is to show that the solution to (4.8) gives a very good lower bound for the optimal solutions. Inclusion of other inequalities should improve the performance of the branch and cut solver. In Tables 1–4, which contain computational results, we use the following abbreviations:

RAT: ratio of solution to (4.8) and integer optimum; RAT = LP/OPT;
NI: number of problems where solution to (4.8) is integer;
TT: total time to solve (4.8) (includes I/O) in seconds.

TABLE 1
Random-sparse problems.

$\|V\|$	$\|E\|$	$\|N_1\|$	$\|N_2\|$	Avg. RAT	Max RAT	Min RAT	Avg. TT	Max TT	Min TT	NI
40	80	22	8	.998	1	.994	10.6	21.2	6	3
60	120	33	12	.999	1	.998	39.7	47.9	36.2	3
80	160	44	16	.999	1	.997	291.5	614.3	81.8	2
100	200	55	20	.999	1	.997	1112	1476	731	2

TABLE 2
Euclidean-sparse problems.

$\|V\|$	$\|E\|$	$\|N_1\|$	$\|N_2\|$	Avg. RAT	Max RAT	Min RAT	Avg. TT	Max TT	Min TT	NI
40	80	22	8	.999	1	.999	19.3	33.6	4.7	4
60	120	33	12	.999	1	.998	103	181	25.5	2
80	160	44	16	.999	1	.999	324	831	93	3
100	200	55	20	.998	1	.995	902	1522	171	1

TABLE 3
Random-complete problems.

$\|V\|$	$\|E\|$	$\|N_1\|$	$\|N_2\|$	Avg. RAT	Max RAT	Min RAT	Avg. TT	Max TT	Min TT	NI
30	435	17	6	1	1	1	13.1	17.1	7.1	5
40	780	22	8	1	1	1	27.6	48.9	17.5	5
50	1225	27	10	.999	1	.999	103	178	57.2	4
30	435	24	6	.999	1	.999	7.4	10.3	2.6	4
40	780	32	8	1	1	1	18.9	21.5	15.2	5
50	1225	40	10	.999	1	.999	30.4	50.7	15.3	4

TABLE 4
Euclidean-complete problems.

$\|V\|$	$\|E\|$	$\|N_1\|$	$\|N_2\|$	Avg. RAT	Max RAT	Min RAT	Avg. TT	Max TT	Min TT	NI
30	435	17	6	1	1	1	14.9	19.5	11.6	5
40	780	22	8	.999	1	.999	62.3	119	36.4	4
50	1225	27	10	1	1	1	249	336	105	5
30	435	24	6	.999	1	.999	11.4	14.4	8.7	3
40	780	32	8	1	1	1	25	33.6	13.5	5
50	1225	40	10	.999	1	.999	80.7	143	38.8	3

For each problem size, five different, randomly generated problems are solved. The results contain the average, minimum, and maximum value for the five runs.

The computational results were run on a VAX 8700, and the LP solver used was XMP, written by Roy Marsten.

For the sparse graphs, the number of edges $|E| = 2|V|$. The problems solved have $|N_1 \cup N_2| = 3|V|/4$ and $|N_2| = |V|/5$. For the complete graphs, we also consider the case where $V = N_1 \cup N_2$, $|N_2| = |V|/5$.

Reference [12] gives several reduction procedures, which are very effective. However, since our objective is only to show the effectiveness of the directed formulation, none of the reduction procedures are used.

The main observations from the computational runs are as follows:

(1) Solving the LP-relaxation (4.8) gives the integer optimum a significant number of times. On average, it brings us to 99.9 percent of optimality and, in the worst case (in a total of 100 problems solved), to 99.4 percent of optimality.

(2) Increasing $|N_1 \cup N_2|$ from $3|V|/4$ to $|V|$, while keeping $|N_2|$ fixed, reduces the time taken to solve (4.8) for the complete graphs on which we ran this comparison (a total of 30 problems of each type were solved). The reduction in running time seems to increase with the size of the problem.

Our results are promising, and we feel that the directed formulation may allow us to solve larger problems than the undirected one. Most of the fractional vertices were cut off by wheelbarrow and bicycle inequalities, and their compositions formed by using the procedure in § 3.4. Heuristic procedures to identify these inequalities should improve the performance of our cutting plane procedure. However, all problems considered by us were solved to optimality at the end of branch and cut.

Acknowledgments. I thank Edgar Gorres for help with the computational results. I also thank the referees for a careful reading of the paper and several valuable suggestions, in particular, the proof of Lemma 3.2.1.

REFERENCES

[1] A. BACHEM AND M. GRÖTSCHEL, *New Aspects of Polyhedral Theory*, in Modern Applied Mathematics: Optimization and Operations Research, North-Holland, Amsterdam, 1982, pp. 51–106.

[2] E. BALAS AND W. PULLEYBLANK, *The perfectly matchable subgraph polytope of a bipartite subgraph*, Networks, 13 (1983), pp. 495–516.

[3] S. CHOPRA, *The equivalent subgraph and directed cut polyhedra on series-parallel graphs*, SIAM J. Discrete Math., 5(1992), to appear.

[4] ———, *Polyhedra of the equivalent subgraph problem and some edge connectivity problems*, Tech. Report, KGSM, Northwestern University, Evanston, IL, 1990.

[5] S. CHOPRA AND M. R. RAO, *The Steiner tree problem* I: *Formulations, compositions and extension of facets*, Math. Programming, submitted, 1988.

[6] ———, *The Steiner tree problem* II: *Properties and classes of facets*, Math. Programming, submitted, 1988.

[7] G. CORNUEJOLS, J. FONLUPT, AND D. NADDEF, *The travelling salesman problem on a graph and some related integer polyhedra*, Math. Programming, 33 (1985), pp. 1–27.

[8] D. R. FULKERSON, *Blocking and anti-blocking pairs of polyhedra*, Math. Programming, 1 (1971), pp. 168–194.

[9] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractibility: A Guide to the Theory of* NP-*completeness*, W. H. Freeman, San Francisco, CA, 1979.

[10] M. GRÖTSCHEL AND C. L. MONMA, *Integer polyhedra associated with certain network design problems with connectivity constraints*, SIAM J. Discrete Math., 3 (1990), pp. 502–523.

[11] M. GRÖTSCHEL, C. L. MONMA, AND M. STOER, *Facets for polyhedra arising in the design of communication networks with low connectivity constraints*, Report No. 187, Universität Augsburg, Augsburg, Germany, 1989.

[12] ———, *Computational results in the design of communication networks with low-connectivity constraints*, Report No. 188, Universität Augsburg, Augsburg, Germany, 1989.

[13] H. T. HSU, *An algorithm for finding a minimum equivalent graph of a digraph*, J. Assoc. Comput. Mach., 22 (1975), pp. 11–16.

[14] L. LOVASZ, *Combinatorial Problems and Exercises*, North-Holland, Amsterdam, 1979.

[15] A. R. MAHJOUB, *Two edge connected spanning subgraphs and polyhedra*, Tech. Report No. 88520-OR, University of Bonn, Bonn, Germany, 1988.

[16] S. MARTELLO AND P. TOTH, *Finding a minimum equivalent graph of a digraph*, Networks, 12 (1982), pp. 89–100.

[17] D. M. MOYLES AND G. L. THOMPSON, *An algorithm for finding a minimum equivalent graph of a digraph*, J. Assoc. Comput. Mach., 16 (1969), pp. 455–460.

[18] D. NADDEF AND G. RINALDI, *The crown inequalities for the symmetric traveling salesman polytope*, Report R.249, IASI-CNR, 1988, Math. Oper. Res., to appear.

[19] M. PADBERG AND T. Y. TSUNG, *An analytical comparison of different formulations of the travelling salesman problem*, New York University Research Report, New York, 1988.

[20] M. PADBERG AND G. RINALDI, *A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems*, Report R.247, IASI-CNR, Rome, 1988.

[21] M. B. RICHEY, R. C. PARKER, AND R. L. RARDIN, *An efficiently solvable case of the minimum weight equivalent subgraph problem*, Networks, 15 (1985), pp. 217–228.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 338–344, August 1992
© 1992 Society for Industrial and Applied Mathematics
003

ALMOST SAFE GOSSIPING IN BOUNDED DEGREE NETWORKS*

KRZYSZTOF DIKS† AND ANDRZEJ PELC‡

Abstract. A variant of the well-known gossip problem is studied. Each of n members of a communication network has a piece of information that should be made known to everybody else. This is to be done by placing a sequence of two-party phone calls along the lines of the network. During each call, the two participants exchange all information they currently have, in a unit of time. It is assumed that calls fail independently with fixed probability $0 < p < 1$ and that no information is exchanged during a failed call. For communication networks of bounded degree, efficient schemes of calls are shown that assure complete communication with probability converging to 1 as n grows. Both the number of calls and the time they use are of minimal order.

Key words. communication networks, gossip schemes, random faults, reliability

AMS(MOS) subject classifications. 05C38, 94C15

1. Introduction. Each of n members of a communication network (modeled by a simple connected undirected graph) has a piece of information unknown to others, which should be made known to everybody else. This is to be accomplished by a series of two-party telephone calls satisfying the following assumptions:

1. Each call requires one unit of time (measured by a global clock);
2. Everybody can participate in, at most, one call per time unit;
3. Calls can be placed only along the links of the network;
4. During a call, the two participants exchange all their current information.

Two important parameters of such a series of calls, referred to as a gossip scheme, are the total number of calls and the total time used. The problem of finding gossip schemes minimizing the above parameters has been extensively studied, first for complete networks (cf., e.g., [2], [3], [5], [10], [14], [16], [17]), and then for other communication graphs [6], [8], [11], [13]. For more references, see the survey [12].

The problem of constructing gossip schemes optimal in the above sense becomes more complicated if we assume that some calls may fail and that no information is exchanged during a failed call. In [4], [9] bounds on the minimum total number of calls in a complete network are obtained, assuming that up to k calls may fail. The results imply that $\theta(nk)$ calls are sufficient and necessary. It follows from [15] that the minimum time required by a gossip scheme is at least $\lceil \log n \rceil + k$ under the same assumptions. Liestman [15] studies networks with the fewest possible links in which gossip schemes working in minimum time, assuming k failed calls, can be constructed.

Diks and Pelc [7] originated a probabilistic approach to gossiping with failures. They assumed that links of the complete network fail independently with fixed probability $0 < p < 1$. Faults were supposed permanent (the fault status of a link did not change during the scheme), and no information could be exchanged during a call along a faulty link. In [7] efficient gossip schemes were constructed that assured complete information exchange with probability converging to 1, as n grows. Although not all links of the complete network were used in executing those schemes, $O(n \log n)$ links were necessary,

* Received by the editors May 2, 1990; accepted for publication (in revised form) April 30, 1991.

† Instytut Informatyki, Uniwersytet Warszawski, PKIN, p. 850, 00-901 Warszawa, Poland. This work was done during this author's visit at the Université du Québec à Hull, Québec, Canada.

‡ Département d'Informatique, Université du Québec à Hull, C.P. 1250, succ. "B," Hull, Québec J8X 3X7, Canada. This author's research was supported in part by Natural Sciences and Engineering Research Council of Canada grant OGP 0008136.

and it was proved that this asymptotic bound cannot be improved for reliable schemes in the random link failure model. This implies that no reliable schemes are possible in this model for bounded maximum degree networks, including such important examples of communication networks as rings, grids, or hexagonal meshes.

In this paper, we pursue the study of gossiping with random failures under a different scenario. Instead of assuming permanent link failures, we suppose that individual calls, rather than links, are subject to independent failures with fixed probability $0 < p < 1$. Now, even failures of calls placed along the same link are independent. We assume that both parties attempting a failed call become aware of the failure. Our goal is again to find reliable gossip schemes using a minimum number of calls and/or working in minimum time.

The reliability of a gossip scheme working on a communication network G is defined as the probability that complete information exchange will be achieved upon completing the scheme. For any family $\{G_n : n \geqq 1\}$ of n-node networks, a scheme working on these networks is called *almost safe* if its reliability for G_n converges to 1 as $n \to \infty$.

As usual in the presence of failures, two ways of constructing a gossip scheme are possible. One way is *nonadaptive*; that is, all calls must be predetermined (by specifying which pairs of people communicate at a given time unit) before the scheme is started, without the possibility of modifying the sequence of calls depending on which calls succeeded and which failed. This approach has been traditionally adopted in literature [4], [9], [15]. (In [9] it was called static.) Another way (cf. [7]) is *adaptive*; that is, everybody can decide whom to call in a given time unit, depending on the outcome of previous calls. However, in making this decision, we can only take advantage of the information we currently have; that is, we do not assume the existence of any central monitor supervising the execution of the scheme.

Our main result concerns nonadaptive almost safe gossip schemes. Contrary to the situation in [7], under our present scenario such schemes do exist for any class of networks, even with bounded maximum degree. More precisely, we show that for any class $\{G_n : n \geqq 1\}$ of n-node networks with bounded maximum degree and diameters $D(n)$, there is an almost safe gossip scheme working in time $O(D(n))$ and using $O(n \log n)$ calls. None of these asymptotic bounds can be improved.

An elementary part of a gossip scheme is an individual call with specified communicating parties x, y and specified time unit counted since the beginning of the scheme. We say that in time unit i, a call is placed between x and y, or that x calls y.

2. Preliminaries. A *communication network* is a finite connected undirected graph without self-loops or multiple edges. Nodes (vertices) of this graph represent members of the network, and links (edges) represent telephone lines along which calls can be placed. The *degree* of a node is the number of adjacent nodes, and the maximum degree of the network is the maximum over all degrees of its nodes. Nodes of degree 1 are called *leaves* of the network. A *tree* is a cycle-free network, and a *spanning tree* of a network G is a tree that is a subnetwork of G with the same set of nodes. A *path* is a tree of maximum degree $\leqq 2$, and the number of links in a path is called its *length*. The *distance* between two nodes u, v of a network is the length of the shortest path in G with leaves u and v. The *diameter* of a network is the largest distance between any pair of its nodes.

Every tree with a specified node r, called the root, is called a *rooted tree*. The ith level of a rooted tree is the nonempty set of nodes at distance i from the root. The height of a rooted tree is the maximum over indices of its levels. For any node x on level i, the adjacent nodes on level $i + 1$ are called the *children* of x, and the adjacent node on level $i - 1$ is called the *parent* of x (denoted $p(x)$). For every x, we fix an enumeration of

children of x and call them the first, second, $\cdots$, jth, $\cdots$ child. For any pair of distinct nodes x, y of a rooted tree, x is called an *ancestor* of y, and y a *descendant* of x, if x is a node on the path between y and the root.

In our probabilistic considerations, we use the following estimate of the probability that at most $\lfloor(1-\varepsilon)mq\rfloor$ successes will happen in a Bernoulli series with parameters m, q. This is a version of Chernoff bound.

LEMMA 2.1 (see [1]). *It holds that*

$$\sum_{k=0}^{\lfloor(1-\varepsilon)mq\rfloor}\binom{m}{k}q^k(1-q)^{m-k}\leqq e^{-\varepsilon^2 mq/2}$$

for $0\leqq q\leqq 1$ *and* $0\leqq\varepsilon\leqq 1$.

3. Main result. The following proposition (cf. Diks and Pelc [7]) gives an asymptotic lower bound on the total number of calls that a nonadaptive almost safe gossip scheme must use. We include the short proof for completeness.

PROPOSITION 3.1. *Let* $\{G_n : n\geqq 1\}$ *be any class of n-node networks. For any constant* $c<-1/(4\log p)$ *and any nonadaptive gossip scheme on* G_n *using at most* $cn\log n$ *calls, the probability of complete information exchange converges to* 0, *as* $n\to\infty$.

Proof. It is enough to prove that, with probability converging to 1, there exists a person in the network G_n such that all calls in which x participates are faulty. Assume that a gossip scheme uses at most $cn\log n$ calls. Let R be the set of people participating in at most $4c\log n$ calls. Hence R has size at least $n/2$. For any $x\in R$, denote by $N(x)$ the set of those $y\in R$ that are called by x. We construct a subset $S\subset R$ by induction. Start with any $x_0\in R$. If $x_0,\cdots,x_k$ are already constructed and $V_k=R\setminus\{x_0,\cdots,x_k\}\setminus N(x_0)\setminus\cdots\setminus N(x_k)$ is nonempty, let x_{k+1} be any element of V_k. If $V_k=\phi$, we put $S=\{x_0,\cdots,x_k\}$. Since R has size at least $n/2$ and at each step at most $1+4c\log n$ elements are eliminated, S has size at least $\lfloor n/(2(1+4c\log n))\rfloor$, thus at least $n/(9c\log n)$, for sufficiently large n. By construction, S consists of people who do not call each other. For $x\in S$, let E_x be the event that at least one call in which x participates is not faulty. Events E_x are independent. Let E be the intersection of all events E_x. We have that

$$\text{Prob}\,(E_x)\leqq 1-p^{4c\log n}=1-n^{4c\log p}$$

and

$$\text{Prob}\,(E)\leqq(1-n^{4c\log p})^{n/(9c\log n)}.$$

If $c<-1/(4\log p)$, we have $4c\log p>-1$, and hence

$$(1-n^{4c\log p})^{n/(9c\log n)}\to 0.$$ □

It is obvious that every almost safe gossip scheme working for a class of n-node networks $\{G_n : n\geqq 1\}$ of diameters $D(n)$ must use $\Omega(D(n))$ time units. Thus our main result provides asymptotically optimal almost safe gossip schemes, both regarding the number of calls and the time used, for a large variety of communication networks.

THEOREM 3.2. *Let* $\{G_n : n\geqq 1\}$ *be a class of n-node networks with bounded maximum degree* $d\geqq 2$ *and diameters* $D(n)$. *There exists a nonadaptive almost safe gossip scheme for* G_n *working in time* $O(D(n))$ *and using* $O(n\log n)$ *calls.*

The proof of the theorem is split into a sequence of lemmas.

LEMMA 3.3. *Let P_n be an n-node path, for $n > 4$. Let c be an integer constant $> 4/(1-p)$. There exists a nonadaptive gossip scheme working on P_n in time $4c(n-2)$ with reliability exceeding $1 - 2e^{-c(1-p)n/8}$.*

Proof. Suppose that consecutive nodes of the path P_n are labelled by integers $1, \cdots, n$. Let $T = c(n-2)$. Consider the following algorithm:

PATH TRANSMISSION
for $i := 1$ *to* T *do*
begin
in time unit $2i-1$ each odd node $1 \leqq j < n$ calls its neighbor $j+1$;
in time unit $2i$ each even node $1 < j < n$ calls its neighbor $j+1$
end.

We prove that this scheme (using $2c(n-2)$ time units) communicates information from all nodes $1, \cdots, n-1$ to node n with probability exceeding $1 - e^{-c(1-p)n/8}$. A symmetric scheme transmitting information from node n to all nodes $n-1, n-2, \cdots, 1$ with the same probability can be applied upon completion of PATH TRANSMISSION. The gossip scheme consisting of these two consecutive phases will have reliability exceeding $1 - 2e^{-c(1-p)n/8}$ and will take $4c(n-2)$ time units.

Consider successive periods $p_1, p_2, \cdots, p_T$, each consisting of two time units, counted since the beginning of the scheme. In each period, every node $1 \leqq j \leqq n-1$ calls node $j+1$ exactly once. The probability of the event E_n that, upon completing PATH TRANSMISSION node n gets information held by all other nodes, is clearly equal to the probability that information from node 1 reaches node n. Suppose that information from node 1 is currently at node j. A weighted coin with heads probability $1-p$ is tossed at time unit i if and only if j calls $j+1$ in this time unit according to PATH TRANSMISSION; distinct tosses are independent. If at least $n-1$ heads are obtained in T periods, the event E_n holds. Since in each time period the coin is tossed once or twice, the probability of event E_n is not smaller than the probability of event F_n consisting in getting at least $n-1$ heads in T tosses. Consider the probability R_n of the complement of F_n. We have that

$$R_n = \sum_{j=0}^{n-2} \binom{T}{j} p^{T-j}(1-p)^j.$$

In view of Lemma 2.1, with $m = T$, $q = 1-p$, and $\varepsilon = 1 - 1/(cq)$, we have that $(1-\varepsilon)mq = n-2$; hence

$$R_n \leqq e^{-\varepsilon^2 mq/2} = e^{-\varepsilon^2 c(n-2)(1-p)/2}.$$

Since $\varepsilon = 1 - 1/(cq) > \frac{3}{4}$,

$$R_n < e^{-c(1-p)9(n-2)/32} < e^{-c(1-p)n/8} \quad \text{for } n > 4,$$

which gives the required lower bound on the probability of E_n. □

LEMMA 3.4. *Let T_n be an n-node rooted tree of height $h > 3$ and maximum degree $d \geqq 2$. Let c be an integer constant $> 4/(1-p)$. There exists a nonadaptive gossip scheme working on T_n in time $4dc(h-1)$ with reliability exceeding $1 - 2ne^{-c(1-p)(h+1)/8}$.*

Proof. Let v_0 be the root of T_n and $v_1, \cdots, v_k$, for $k < n$, its (other) leaves. Our gossip scheme consists of two identical consecutive phases: in the first phase, information held by all nodes is communicated to the root, and in the second, the root broadcasts

all obtained information to other nodes. Each of these two phases consists of $t = c(h-1)$ repetitions of the following procedure:

SENDING
for $i := 1$ *to* d *do*
in time unit i each ith child x on odd levels calls $p(x)$;
for $i := 1$ *to* d *do*
in time unit $d+i$ each ith child x on even levels calls $p(x)$;
(time units are counted since the beginning of procedure execution and rth repetition starts upon completion of $(r-1)$th repetition).

The above procedure clearly takes $2d$ time units; thus the entire scheme works in $4dc(h-1)$ time units. Consider any leaf v_i, $1 \leqq i \leqq k$, and the path $P^{(i)}$ between v_0 and v_i. Let h_i be the length of this path. Without loss of generality, we may assume that $h = h_1 \geqq h_2 \geqq \cdots \geqq h_k$. Let E_i be the event that, upon completion of our gossip scheme, complete information exchange among nodes of the path $P^{(i)}$ is not achieved (E_i is equal to the event that either v_i does not transmit his information to v_0 in the first phase, or that v_0 does not transmit acquired information to v_i in the second phase). Denote the probability of E_i by p_i. Clearly, $p_1 \geqq p_2 \geqq \cdots \geqq p_k$. Let $E = \cup \{E_i : i \leqq k\}$. If the event E does not hold, complete information exchange is achieved in the tree T_n. The probability of E does not exceed $k \cdot p_1$, and Lemma 3.3 implies that

$$p_1 < 2e^{-c(1-p)(h+1)/8}.$$

Since $k < n$, it follows that the probability of E is smaller than $2ne^{-c(1-p)(h+1)/8}$. □

Our final gossip scheme is a refinement of the previous one, and it enables us to control both the time and the number of calls used.

LEMMA 3.5. *Let T_n be an n-node rooted tree with maximum degree $d \geqq 2$ and such that $\lceil \log_d n \rceil > 4$. Let h be the height of T_n, and c an integer constant $> 4/(1-p)$. There exists a nonadaptive gossip scheme working on T_n in time $\leqq 8cdh$ with reliability exceeding $1 - 2n^2 e^{-c(1-p)\lceil \log_d n \rceil/8}$ and using $\leqq 4cn\lceil \log_d n \rceil$ calls.*

Proof. By definition, $h \geqq \lceil \log_d n \rceil - 1$. Let $L = \lfloor h/(\lceil \log_d n \rceil - 1) \rfloor - 1$. We say that a node v is a root in the ith layer, for $0 \leqq i \leqq L$, if v is a node on level $i(\lceil \log_d n \rceil - 1)$ of T_n and has at least one descendant on level $(i+1)(\lceil \log_d n \rceil - 1)$. A subtree in the ith layer is a subtree of T_n whose set of nodes consists of a root in the ith layer and all of its descendants, none of whose ancestors is a root in the $(i+1)$th layer. It follows that

1. For every $0 \leqq i \leqq L$, every subtree in the ith layer has height between $\lceil \log_d n \rceil - 1$ and $2\lceil \log_d n \rceil - 3$;

2. Every root in the ith layer, for $1 \leqq i \leqq L$, is a leaf of a subtree in the $(i-1)$th layer.

Our gossip scheme consists of two consecutive phases. In the first phase, information held by all nodes is communicated to the root of T_n (which is the root in the zeroth layer), and, in the second phase, the root broadcasts all obtained information to other nodes. The two phases, however, are not identical, but the second one is a "mirror image" in time of the first.

The first phase works by layers. First, information is accumulated in roots in the Lth layer, then in roots in the $(L-1)$th layer, and so forth, and finally in the root in the zeroth layer—the root of T_n. Transmission in the ith layer starts upon completing transmission in the $(i+1)$th layer. For a fixed i, transmission in the ith layer consists in repeating the procedure SENDING, from the proof of Lemma 3.4, $c(2\lceil \log_d n \rceil - 4)$

times in each subtree in the ith layer in parallel. This completes the description of the first phase. The second phase also works by layers, but here we start with transmission in the zeroth layer, and transmission in the $(i+1)$th layer starts upon completing transmission in the ith layer.

Since one run of the procedure SENDING takes $2d$ time units, the total time used by our gossip scheme is

$$4(L+1)cd(2\lceil \log_d n\rceil - 4) \leqq 8cd\lfloor h/(\lceil \log_d n\rceil - 1)\rfloor(\lceil \log_d n\rceil - 2) \leqq 8cdh.$$

During the entire scheme, $2c(2\lceil \log_d n\rceil - 4)$ calls are placed between every node and its parent; thus a total of at most $4cn\lceil \log_d n\rceil$ calls is required.

It remains to estimate the reliability of our scheme. Let T' be a subtree of height h' in the ith layer, for some $0 \leqq i \leqq L$. Denote by E' the event that complete information exchange is not achieved in the subtree T' upon completion of our gossip scheme. In view of Lemma 3.4, the probability of E' is at most

$$2ne^{-c(1-p)(h'+1)/8}.$$

Since $h' \geqq \lceil \log_d n\rceil - 1$, we can overestimate this probability by

$$2ne^{-c(1-p)\lceil \log_d n\rceil/8}.$$

Let E be the union of events E' over all subtrees in all layers $i \leqq L$. There are clearly less than n such subtrees, hence the probability of E is smaller than

$$2n^2e^{-c(1-p)\lceil \log_d n\rceil/8}.$$

Clearly, if the event E does not happen, complete information exchange is achieved in the entire tree T_n. Thus the reliability of our scheme exceeds

$$1 - 2n^2e^{-c(1-p)\lceil \log_d n\rceil/8}. \qquad \square$$

Proof of Theorem 3.2. For every $n \geqq 1$, let T_n be a spanning tree of the network G_n, with diameter $D(n)$. In each T_n, consider as the root such a leaf that $D(n)$ is the height of the resulting rooted tree. The gossip scheme described for T_n in Lemma 3.5 can be considered to work on networks G_n.

Let $c > 16 \log_e d/(1-p)$ (which also implies that $c > 4/(1-p)$, as required in Lemma 3.5). Hence

$$e^{-c(1-p)\lceil \log_d n\rceil/8} = n^s$$

for some $s < -2$, and, consequently, the reliability of our scheme—which in view of Lemma 3.5 exceeds $1 - 2n^2n^s$—converges to 1 as $n \to \infty$. It follows immediately from Lemma 3.5 that the scheme works in time $O(D(n))$ and uses $O(n \log n)$ calls. $\square$

4. Conclusion. We have shown nonadaptive almost safe gossip schemes working for bounded degree networks and using the number of calls and time of minimal order. It remains to find such schemes for arbitrary networks, in particular, for hypercubes.

In the adaptive version of the problem, the total number of calls and the total time needed are random variables; hence the expected values of these parameters seem to be reasonable measures of adaptive gossip scheme performance. A modification of our techniques gives gossip schemes working for bounded degree networks of diameters $D(n)$ in expected time $O(D(n))$ and using an expected number of $O(n)$ calls.

REFERENCES

[1] D. ANGLUIN AND L. G. VALIANT, *Fast probabilistic algorithms for Hamiltonian circuits and matchings*, J. Comput. System Sci., 18 (1979), pp. 155–193.
[2] R. BAKER AND R. SHOSTAK, *Gossips and telephones*, Discrete Math., 2 (1972), pp. 191–193.
[3] A. BAVELAS, *Communication patterns in task-oriented groups*, J. Acoust. Soc. Amer., 22 (1950), pp. 725–730.
[4] K. A. BERMAN AND M. HAWRYLYCZ, *Telephone problems with failures*, SIAM J. Algebraic Discrete Meth., 7 (1986), pp. 13–17.
[5] R. BUMBY, *A problem with telephones*, SIAM J. Algebraic Discrete Meth., 2 (1981), pp. 13–18.
[6] N. COT, *Extensions of the telephone problem*, in Proc. 7th SE Conf. on Combinatorics, Graph Theory and Computing, Utilitas Mathematica, Winnipeg, Manitoba, Canada, 1976, pp. 239–256.
[7] K. DIKS AND A. PELC, *Reliable gossip schemes with random link failures*, in Proc. of the 28th Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, 1990, pp. 978–987.
[8] A. FARLEY AND A. PROSKUROWSKI, *Gossiping in grid graphs*, J. Combin. Inform. Systems Sci., 5 (1980), pp. 161–172.
[9] R. HADDAD, W. ROY, AND A. A. SCHAFFER, *On gossiping with faulty telephone lines*, SIAM J. Algebraic Discrete Meth., 8 (1987), pp. 439–445.
[10] A. HAJNAL, E. C. MILNER, AND E. SZEMEREDI, *A cure for the telephone disease*, Canad. Math. Bull., 15 (1972), pp. 447–450.
[11] F. HARRARY AND A. J. SCHWENK, *The communication problem on graphs and digraphs*, J. Franklin Inst., 297 (1974), pp. 491–495.
[12] S. M. HEDETNIEMI, S. T. HEDETNIEMI, AND A. L. LIESTMAN, *A survey of gossiping and broadcasting in communication networks*, Networks, 18 (1988), pp. 319–349.
[13] R. LABAHN, *The telephone problem for trees*, Elektron. Informationsverarb. u. Kybernet., 22 (1986), pp. 475–485.
[14] H. G. LANDAU, *The distribution of completion times for random communication in a task-oriented group*, Bull. Math. Biophys., 16 (1954), pp. 187–201.
[15] A. L. LIESTMAN, *Fault-tolerant broadcast graphs*, Networks, 15 (1985), pp. 159–171.
[16] A. SERESS, *Quick gossiping without duplicate transmissions*, Graphs Combin., 2 (1986), pp. 363–383.
[17] R. TIJDEMAN, *On a telephone problem*, Nieuw Arch. Wisk., 19 (1971), pp. 188–192.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 345–365, August 1992
© 1992 Society for Industrial and Applied Mathematics
004

GALOIS GROUPS AND FACTORING POLYNOMIALS OVER FINITE FIELDS*

LAJOS RÓNYAI†

Abstract. Let p be a prime and F be a polynomial with integer coefficients. Suppose that the discriminant of F is not divisible by p. Denote by m the degree of the splitting field of F over Q and by L, the maximal size of the coefficients of F. Then, assuming the generalized Riemann hypothesis (GRH), the irreducible factors of F modulo p in (deterministic) time polynomial in deg F, m, L, and log p can be found. This is a generalization of a result of Huang [*Riemann hypothesis and finding roots over finite fields*, in Proceedings of the 17th ACM Symposium on Theory of Computing, Providence, Rhode Island, 1985, pp. 121–130]. As an application, under GRH certain equations of the form $nP = R$ can be solved, where R is a given and P is an unknown point of an elliptic curve defined over $GF(p)$ in polynomial time (n is counted in unary). Finally, an elliptic analogue of a result obtained recently by von zur Gathen [*Theoretical Computer Science*, 52 (1987), pp. 77–89] and independently by Mignotte and Schnorr [*Comptes Rendus de l'Académie des Sciences. Série* I. *Mathematique*, 306 (1988), pp. 467–472] is proved, and thus a step is taken toward enlarging the set of primes p for which, under GRH, polynomials over $GF(p)$ in deterministic polynomial time can be factored.

Key words. factoring polynomials, finite fields, Galois groups, elliptic curves, generalized Riemann hypothesis

AMS(MOS) subject classifications. 68Q25, 68Q40, 11Y16

1. Introduction.

We consider here the problem of factoring polynomials over finite fields. The problem can be formulated as follows. We have a polynomial $f \in GF(p^r)[x]$ (p is a prime), and our aim is to find polynomials $f_i \in GF(p^r)[x]$ irreducible over $GF(p^r)$ such that $f = f_1 f_2 \cdots f_k$. The polynomials f_i are unique up to scalar multiples and permutations. The problem can be solved deterministically in time $(p + r + \deg f)^{O(1)}$ and by randomized (Las Vegas) methods in time $(\log p^r + \deg f)^{O(1)}$ (cf. Berlekamp [B1], [B2], for variants and refinements; Rabin [Ra]; Ben-Or [B]; Cantor and Zassenhaus [CZ]). The randomized algorithms run in time polynomial in the input size (with respect to the standard encoding) and work quite well in practice. However, it is an open problem whether there exist deterministic methods running in time $(\log p^r + \deg f)^{O(1)}$.

Recently, some results related to the deterministic complexity of the problem were obtained that use the generalized Riemann hypothesis (GRH). Deterministic polynomial time methods are known with GRH

—for factoring binomials (Adleman, Manders, and Miller [AMM], Huang [H1]);
—for the modulo p reduction of an irreducible polynomial $F \in Z[x]$ for which the Galois group Gal (F) of F over Q is Abelian (Huang [H2]);
—when the preceding result is extended to the case when Gal (F) is solvable (Evdokimov [E]);
—for arbitrary f, if the prime factors of $p - 1$ are small (Moenck [Mo], von zur Gathen [G], Mignotte and Schnorr [MS]);
—if f has a bounded number of irreducible factors (Rónyai [R1]).

* Received by the editors March 20, 1989; accepted for publication April 30, 1991. This research was partially supported by Hungarian National Foundation for Scientific Research grants 1812 and 2581. A preliminary version of this paper appeared in the Proceedings of the 30th Institute for Electrical and Electronics Engineers (IEEE) Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, 1989.

† Department of Computer Science, University of Chicago, 1100 E58th Street, Chicago, Illinois 60637. Permanent address, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, Victor Hugo u. 18–22, H-1132 Hungary.

In this paper, we continue this line of research. The main result is Theorem 5.5. If $F \in Z[x]$ and the discriminant $D(F)$ is not divisible by p, then, assuming GRH, the reduction f of F modulo p can be factored over $GF(p)$ in time $(\deg F + m + \log p + L)^{O(1)}$, where m is the degree of the splitting field of F over Q, and L is the maximum size of the coefficients of F. The result is a generalization of Huang's result from [H2], mentioned above. Furthermore, some key ideas come from, or were inspired by, that paper.

The organization of the paper is as follows. Sections 2–5 correspond to the major steps of the proof. In § 2 we show how an automorphism of a finite semisimple commutative algebra $\mathcal{A}$ can be used to find a nontrivial ideal of $\mathcal{A}$. In § 3 we give an algorithm to find all the minimal ideals of $\mathcal{A}$, provided that we know sufficiently many automorphisms of $\mathcal{A}$. Application of results of this type to algebras of form $\mathcal{A} = GF(p)[x]/(f)$, $f \in GF(p)[x], f \mid x^p - x$ can be used to obtain factorization algorithms (Corollaries 2.4 and 3.2). In § 4 we settle the case when F defines a Galois extension over Q, i.e., when $m = \deg F$. We essentially work with the finite algebra $\mathcal{A} = Z_p[\alpha]/pZ_p[\alpha]$, where α is a root of F. The elements of the Galois group of F induce automorphisms of $\mathcal{A}$, and the results of § 3 are applicable. Here we prove an unconditional factoring result, also (Theorem 4.3). The proof of the main theorem is concluded in § 5.

The rest of the paper is devoted to applications related to elliptic curves defined over $GF(p)$. We consider elliptic curves $\mathcal{E}$ defined over $GF(p)$ and given by a Weierstrass equation (cf. Silverman [Si], Lang [L2]) of the form

$$Y^2 = X^3 + aX + b, \quad a, b \in GF(p), \quad 4a^3 + 27b^2 \neq 0.$$

The curve $\mathcal{E}$ has an Abelian group structure. Assuming GRH, we solve equations of the form $nP = R$, where R is a given, P is an unknown point of $\mathcal{E}$, and either R is the 0 element of the group or the x-coordinate of R is in $GF(p)$. Multiplication by n is understood with respect to the group law on the curve.

In § 7 we prove an elliptic analogue of the result of von zur Gathen [G] and Mignotte and Schnorr [MS]: under GRH, we can factor arbitrary polynomials over $GF(p^r)$ in deterministic polynomial time if we have an elliptic curve $\mathcal{E}$ defined over $GF(p)$ such that the order of the subgroup of $\mathcal{E}$ consisting of points defined over $GF(p^2)$ has small prime divisors only. Here the latter group plays essentially the same role that the multiplicative group $GF(p)^*$ has in [G] and [MS]. In the algorithm, we use the linear algebra approach from [R1] and [R2].

We intend to make use of GRH transparent. To this end, we define two preconditions depending on a positive integer parameter d.

Precondition $\mathbf{P}_d$. We have a list of polynomials $f_r, g_r \in GF(p)[x]$ for every prime $r \mid d$ such that f_r is an irreducible factor of the rth cyclotomic polynomial over $GF(p)$, $\deg g_r < \deg f_r$, and $b_r = g_r \bmod f_r$ is an rth nonresidue in the field $F_r = GF(p)[x]/(f_r)$.

Precondition $\mathbf{P}_{\leq d}$ is defined similarly, except that we have f_r and g_r for *all* primes r not exceeding d.

We have no unconditional deterministic polynomial time algorithm to find lists of polynomials satisfying precondition $\mathbf{P}_d$ or $\mathbf{P}_{\leq d}$. However, from Huang [H2, § 4], it follows that under GRH both conditions can be met in time $(d + \log p)^{O(1)}$. Throughout the paper, we use GRH only in assuming a precondition of type $\mathbf{P}_d$ or $\mathbf{P}_{\leq d}$ for an appropriate d. Thus, in the statements, no explicit reference is made to GRH.

2. Finite algebras, automorphisms, and factoring. In this section, we collect some facts about finite algebras and their automorphisms, which we need later. Also, we present

here an important auxiliary algorithm (Theorem 2.3). This method will be our principal tool in obtaining partial factorization of polynomials with the help of automorphisms.

Let $\mathscr{A}$ be a finite semisimple commutative algebra over the field $GF(p)$. By Wedderburn's theorem (cf. Herstein [He], Kertész [Ke], Pierce [P]), $\mathscr{A}$ is a direct sum of its minimal ideals $\mathscr{A}_i$

$$\mathscr{A} = \mathscr{A}_1 \oplus \mathscr{A}_2 \oplus \cdots \oplus \mathscr{A}_k,$$

where the $\mathscr{A}_i$ are finite extension fields of $GF(p)$. In particular, $\mathscr{A}_i$ contains a subfield $\mathscr{B}_i$ isomorphic to the prime field $GF(p)$. Following von zur Gathen [G], we call the direct sum

$$B(\mathscr{A}) = \mathscr{B}_1 \oplus \mathscr{B}_2 \oplus \cdots \oplus \mathscr{B}_k$$

the *Berlekamp subalgebra* of $\mathscr{A}$. The Berlekamp subalgebra can be characterized as the set of fixed points of the Frobenius map of $\mathscr{A}$

$$B(\mathscr{A}) = \{x \in \mathscr{A};\ x^p = x\}.$$

Let e_i denote the identity element of $\mathscr{A}_i$ (or, what is the same, the identity element of $\mathscr{B}_i$). These elements are the *primitive idempotents* of $\mathscr{A}$.

Let $\mathscr{A}$ be a semisimple commutative algebra over $GF(p)$. Suppose that $\mathscr{A}$ contains a field extension F of $GF(p)$. Then, clearly, we can consider $\mathscr{A}$ as an F-algebra, as well. By an *F-automorphism* of $\mathscr{A}$, we mean an invertible F-linear map $\psi : \mathscr{A} \to \mathscr{A}$ for which $\psi(xy) = \psi(x)\psi(y)$ holds for every $x, y \in \mathscr{A}$. A $GF(p)$-automorphism is simply referred to as an *automorphism*. Every F-automorphism of $\mathscr{A}$ maps primitive idempotents to primitive idempotents. If

$$\mathscr{A} \cong F \oplus F \oplus \cdots \oplus F (k \text{ times}),$$

then every permutation of the k primitive idempotents induces a unique F-automorphism of the algebra $\mathscr{A}$. We conclude that the automorphism group $\mathscr{A}$ is isomorphic to the symmetric group S_k on k letters. Every element $u \in \mathscr{A}$ can be written uniquely as

$$u = c_1 e_1 + c_2 e_2 + \cdots + c_k e_k,$$

where $c_i \in F$. If ϕ is an F-automorphism of $\mathscr{A}$, then

$$\phi(u) = c_1 \phi(e_1) + c_2 \phi(e_2) + \cdots + c_k \phi(e_k). \tag{2.1}$$

By setting $F = GF(p)$, we obtain that the group of automorphisms of an algebra of the form $B(\mathscr{A})$ with k primitive idempotents is isomorphic to S_k and that an automorphism is completely described (2.1) by its action on the primitive idempotents.

The problem of factoring polynomials over $GF(p)$ is closely related to the problem of finding invariant subspaces of matrices over $GF(p)$. Indeed, if

$$f(x) = x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n \in GF(p)[x]$$

is a polynomial to be factored, then we can form the *companion matrix* A_f of f,

$$A_f = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & -a_n \\ 1 & 0 & 0 & \cdots & 0 & -a_{n-1} \\ 0 & 1 & 0 & \cdots & 0 & -a_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -a_1 \end{pmatrix}.$$

A_f is an $n \times n$ matrix over $GF(p)$, and, for the characteristic polynomial, we have $\det(A_f - xI) = (-1)^n f(x)$. If we have a nontrivial invariant subspace U of A_f (acting

on the linear space of column vectors V_n of length n over $GF(p)$), then we can consider the action of A_f on U. In this way, we obtain a linear transformation of U, and its characteristic polynomial is a nontrivial divisor of f. Conversely, it is easy to verify that if g is a divisor of f, then Ker $(g(A_f))$ is a nontrivial invariant subspace of A_f. Note that forming A_f and computing the characteristic or the minimal polynomial of a matrix from $M_n(GF(p))$ all can be done in time $(n + \log p)^{O(1)}$.

An algebra $\mathcal{A}$ over $GF(p)$ can be given by *structure constants*. If $a_1, a_2, \cdots, a_n$ is a linear basis of $\mathcal{A}$ over $GF(p)$, then multiplication can be specified by representing the products $a_i a_j$ as linear combinations of the a_i as follows:

$$a_i a_j = \gamma_{ij1} a_1 + \cdots + \gamma_{ijn} a_n.$$

The coefficients $\gamma_{ijk} \in GF(p)$ are called structure constants.

Recall that a nonzero element $u \in \mathcal{A}$ is a *zero divisor* in $\mathcal{A}$ if there exists $0 \neq v \in \mathcal{A}$ such that $uv = 0$.

PROPOSITION 2.1. *The following algorithmic problems are equivalent up to deterministic polynomial time reductions*:

(a) *finding a nontrivial factor of* $f \in GF(p)[x]$;

(b) *finding (a basis of) an ideal* $\mathcal{I}$, $(0) < \mathcal{I} < \mathcal{A}$ *in a semisimple algebra* $\mathcal{A}$ *over* $GF(p)$; $\mathcal{A}$ *is given by structure constants*;

(c) *finding a zero divisor* $u \in \mathcal{A}$ *in a semisimple algebra* $\mathcal{A}$ *over* $GF(p)$; $\mathcal{A}$ *is given by structure constants*;

(d) *finding (a basis of) an ideal* $\mathcal{I}$, $(0) < \mathcal{I} < \mathcal{A}$ *in an algebra* $\mathcal{A}$ *over* $GF(p)$ *of the form* $GF(p)[x]/(f)$, *where* $f \in GF(p)[x]$ *is a given polynomial and* $f(x) \mid x^p - x$;

(e) *finding a nontrivial invariant subspace* $(0) < U < V_n$ *of the companion matrix* A_f, *where* $f \in GF(p)[x]$ *and* $f(x) \mid x^p - x$.

Proof. For the fact that (b) can be solved using (a), we refer to [R3]. The equivalence of (b) and (c) is obvious because $u\mathcal{A}$ is a proper ideal of $\mathcal{A}$ if and only if u is a zero divisor in $\mathcal{A}$. Problem (d) is a special case of problem (b). We now give a reduction of (a) to (d). Let $f \in GF(p)[x]$ be a polynomial to be factored. By Berlekamp's reduction, we can assume that $f(x) \mid x^p - x$. We select a nontrivial ideal $\mathcal{I}$ of $\mathcal{A} = GF(p)[x]/(f)$ with the help of an algorithm, solving (d). Let y be the image of x in the algebra $\mathcal{B} = \mathcal{A}/\mathcal{I}$. Clearly, we have $f(y) = 0$; hence if $m(x) \in GF(p)[x]$ denotes the minimal polynomial of y in $\mathcal{B}$, then $m \mid f$. Moreover, $m \neq 0$ because y generates $\mathcal{B}$. Now, $\dim_{GF(p)} \mathcal{B} < \deg f$ implies that m is a proper divisor. Finally, note that if $\mathcal{I}$ is given, then m can be computed efficiently. We have already discussed the equivalence of (a) and (e); the proof is complete. □

If we have a proper ideal $\mathcal{I}$ of a semisimple commutative algebra $\mathcal{A}$, then its direct complement (i.e., the unique ideal $\mathcal{T}$ of $\mathcal{A}$ for which $\mathcal{A} = \mathcal{I} \oplus \mathcal{T}$ holds) can be found in polynomial time. Indeed, we can use the relation

$$\mathcal{T} = \{ x \in \mathcal{A} ; xy = 0 \text{ for every } y \in \mathcal{I} \}.$$

Having (a basis of) $\mathcal{I}$, we can find a basis of $\mathcal{T}$ by solving a system of linear equations derived from the above (linear) characterization of $\mathcal{T}$.

An extension of the argument of Proposition 2.1. gives the following simple but useful statement.

PROPOSITION 2.2. *Let* $f \in GF(p)[x]$ *be a given polynomial having no multiple factors. The following algorithmic problems are equivalent up to deterministic polynomial time reductions*:

(a) *finding the complete factorization of* $f(x)$ *into irreducible polynomials over* $GF(p)$;

(b) *finding (bases of) the minimal ideals (i.e., the Wedderburn decomposition) of the algebra* $\mathscr{A} = GF(p)[x]/(f)$.

Proof. Let $f = g_1 g_2 \cdots g_k$ be the factorization of f into irreducible polynomials over $GF(p)$. By the Chinese remainder theorem, we have that

$$\mathscr{A} = GF(p)[x]/(f) \cong GF(p)[x]/(g_1) \oplus \cdots \oplus GF(p)[x]/(g_k).$$

We infer that the minimal ideals of $\mathscr{A}$ are $(f/g_i)\mathscr{A}$. This takes care of the reduction from (b) to (a). The converse statement can be verified similarly to the reduction from (a) to (d) in Proposition 2.1. If $\mathscr{T}$ is a given minimal ideal of $\mathscr{A}$, then the direct complement $\mathscr{I}$ to $\mathscr{T}$ is a maximal ideal and is easily computable. From the maximality of $\mathscr{I}$, we infer that m is an irreducible factor of f. Moreover, by the Chinese remainder theorem, the maximal (minimal) ideals of $\mathscr{A}$ correspond uniquely to the irreducible factors of f. □

Suppose that we have an automorphism $\phi \neq 1$ of $\mathscr{A}$. An automorphism is completely described by the sequence of elements $\phi(a_i)$, where $a_1, \cdots, a_n$ is a basis of $\mathscr{A}$ over $GF(p)$.

THEOREM 2.3. *Let* $\mathscr{A}$ *be an algebra over* $GF(p)$, $\dim_{GF(p)} \mathscr{A} = n > 1$, *given by structure constants such that* $B(\mathscr{A}) = \mathscr{A}$. *Suppose also that condition* $\mathbf{P}_n$ *holds and, as part of the input, we have a nontrivial automorphism* ϕ *of* $\mathscr{A}$. *Then, in deterministic time* $(n + \log p)^{O(1)}$, *we can find (a basis of) an ideal* $\mathscr{I}$, $(0) < \mathscr{I} < \mathscr{A}$ *of* $\mathscr{A}$.

Proof. We observe first that at least one of the following statements (A), (B) must hold:

(A) There is an i, $1 \leqq i < n$ and a primitive idempotent e of $\mathscr{A}$ such that $\phi^i \neq \mathrm{id}$ and $\phi^i(e) = e$;

(B) $\phi^n = \mathrm{id}$.

To prove the claim, we view ϕ as a permutation of the primitive idempotents of $\mathscr{A}$. If ϕ has a fixed point, then (A) obviously holds. Otherwise, let $k > 1$ be the length of the smallest cycle of ϕ and let e be an idempotent from a cycle of length k. If $\phi^k \neq \mathrm{id}$, then (A) holds because $k < n$ and $\phi^k(e) = e$. In the remaining case, $\phi^k = \mathrm{id}$, therefore all cycles of ϕ have length k. We infer that $k \mid n$, which implies (B).

If (A) holds, then we obtain a zero divisor in $\mathscr{A}$ as follows. For each i, $(1 \leqq i < n)$ such that $\phi^i \neq \mathrm{id}$, we can easily find an element $b_i \in \mathscr{A}$ for which $\phi^i(b_i) \neq b_i$. Now if the conditions of (A) are met by ϕ^j (and some e), then $d_j = \phi^j(b_j) - b_j$ must be a zero divisor in $\mathscr{A}$ because $d_j \neq 0$, but the e-component of d_j is 0. Note that the b_i can be computed by solving systems of linear equations. Bases for the ideals $d_i\mathscr{A}$ are easily obtained.

We now turn to case (B). We first select an automorphism $\eta \in G$ such that $\eta \neq \mathrm{id}$ and $\eta^r = \mathrm{id}$ for a prime $r \mid n$. The computations will be carried out in $\mathscr{C} = \mathscr{A} \otimes_{GF(p)} F_r$, the F_r-algebra obtained from $\mathscr{A}$ by extending scalars. Recall that F_r is the splitting field of the polynomial $x^r - 1$ over $GF(p)$. F_r is available by condition $\mathbf{P}_n$. From $\dim_{F_r} \mathscr{A} = n$ and $B(\mathscr{C}) = \mathscr{A}$, we immediately see that it suffices to find a proper ideal $(0) < \mathscr{I} < \mathscr{C}$ because, in this case, $\mathscr{I} \cap \mathscr{A}$ is a proper ideal of $\mathscr{A}$.

We put $\tau = \eta \otimes \mathrm{id}_{F_r}$. Clearly, $\tau \neq \mathrm{id}$ is an F_r-automorphism of $\mathscr{C}$ and $\tau^r = \mathrm{id}$. These imply the existence of a primitive rth root of unity $c \in F_r$ and a $0 \neq v \in \mathscr{C}$ such that $\tau(v) = cv$. By condition $\mathbf{P}_n$, the primitive rth roots of unity are available; therefore such c and v can be found efficiently by solving (at most $r - 1$) systems of linear equations over F_r. If v is a zero divisor in $\mathscr{C}$, then $v\mathscr{C}$ is a proper ideal of $\mathscr{C}$, and we have achieved our goal.

We are left with the case when v is invertible. Write $q = \#F_r$, $q - 1 = r^k l$, where $(r, l) = 1$. We now use a variant of the factoring method from [R1].

Algorithm 2.1.

INPUT: An invertible element $v \in \mathscr{C}$, such that $\tau(v) = cv$.

OUTPUT: A proper ideal of the F_r-algebra $\mathscr{C}$, i.e., an ideal different from $\mathscr{C}$ and (0).

Step 1. Compute $w = v^l$ using fast exponentiation. Next, form the sequence of elements

$$w, w^r, w^{r^2}, \cdots, w^{r^m}$$

until we obtain a scalar multiple of $1_{\mathscr{C}}$: $w^{r^m} = d \cdot 1_{\mathscr{C}}$ for some $d \in F_r$. Let z denote the next-to-last element of the sequence.

(* w is not a scalar multiple of $1_{\mathscr{C}}$ because $\tau(w) = c^l w \neq w$; therefore "next-to-last element" exists. This also implies that the polynomial $x^r - d$ splits into linear factors in F_r. *)

Step 2. With the help of the given rth nonresidue $b_r \in F_r$, find the roots $d_1, d_2, \cdots, d_r \in F_r$ of $x^r - d$, using the Tonelli–Shanks algorithm [T], [Sh], [H2, § 2].

Step 3. Compute bases for the ideals $\mathscr{T}_i = (z - d_i \cdot 1_{\mathscr{C}})$ until $\mathscr{T}_i < \mathscr{C}$ is achieved. In this case, output $\mathscr{T}_i$.

(* We have that

$$0 = z^r - d \cdot 1_{\mathscr{C}} = (z - d_1 \cdot 1_{\mathscr{C}}) \cdots (z - d_r \cdot 1_{\mathscr{C}});$$

consequently, not all factors on the right-hand side can be invertible elements in $\mathscr{C}$. On the other hand, the definition of z implies that these factors are nonzero elements of $\mathscr{C}$. *)

End.

Algorithm 2.1 gives a way to find a nontrivial ideal of $\mathscr{A}$. The correctness of the algorithm is clear. The length $m + 1$ of the sequence of elements at Step 1 is bounded by $\log q + 1 \leqq n \log p + 1$. The linear algebra computations employed can be done in polynomial time (using standard methods). We conclude that Algorithm 2.1 runs in time polynomial in n and $\log p$. This completes the proof of Theorem 2.3 as well. □

A straightforward combination of Theorem 2.3 and the reduction in Proposition 2.1 gives the next statement.

COROLLARY 2.4. *Let $f \in GF(p)[x]$, $f(x) \mid x^p - x$, $\deg f = n > 1$ be a given polynomial. Suppose also that condition $\mathbf{P}_n$ holds and, as part of the input, we have a nontrivial automorphism ϕ of $\mathscr{A} = GF(p)[x]/(f)$. Then, in deterministic time $(n + \log p)^{O(1)}$, we can find a nontrivial factor of f.*

3. Algebras with transitive groups of automorphisms. The results of the previous section make it intuitively clear that if we have sufficiently many automorphisms, then we can find the minimal ideals of $\mathscr{A}$. This is indeed the case in the following sense.

THEOREM 3.1. *Let $\mathscr{A}$ be a finite commutative algebra over $GF(p)$, $\dim_{GF(p)} \mathscr{A} = n$, for which $B(\mathscr{A}) = \mathscr{A}$, given by structure constants. Suppose that we have an automorphism group $G = \langle \phi_1, \phi_2, \cdots, \phi_m \rangle$ of $\mathscr{A}$ acting transitively on the set of primitive idempotents of $\mathscr{A}$. Suppose also that condition $\mathbf{P}_n$ holds. Then we can find the minimal ideals of $\mathscr{A}$ in time $(m + n + \log p)^{O(1)}$.*

Proof. First, we select an automorphism $1 \neq \pi \in G$ and find a proper ideal $(0) < \mathscr{I} < \mathscr{A}$ by using the algorithm of Theorem 2.3. This is possible, barring the trivial case where $n = 1$. Then compute the ideal $\mathscr{T} < \mathscr{A}$ for which $\mathscr{A} = \mathscr{I} \oplus \mathscr{T}$. Now, using this

initial decomposition, we want to reduce our original problem to a smaller instance of the same problem. The typical step of this reduction is as follows. Suppose that we have already found a decomposition of $\mathscr{A}$ as a direct sum of ideals, below:

$$\mathscr{A} = \mathscr{I}_1 \oplus \mathscr{I}_2 \oplus \cdots \oplus \mathscr{I}_k, k > 1, \mathscr{I}_j \neq (0).$$

We use this decomposition as an input to the subsequent algorithm.

Algorithm 3.1.
$j := 1$;
while $j \leqq m$ **do**
Compute the ideals
$\phi_j(\mathscr{I}_1), \phi_j(\mathscr{I}_2), \cdots, \phi_j(\mathscr{I}_k)$
and for $1 \leqq s, t \leqq k$ form the intersections
$\mathscr{I}_s \cap \phi_j(\mathscr{I}_t)$.
if there exist indices s, t such that
$(0) < \mathscr{I}_s \cap \phi_j(\mathscr{I}_t) < \mathscr{I}_s$
then
select a finer decomposition from the nonzero ideals of the form $\mathscr{I}_s \cap \phi_j(\mathscr{I}_t)$
$\mathscr{A} = \mathscr{I}_1 \oplus \mathscr{I}_2 \oplus \cdots \oplus \mathscr{I}_l, \quad l > k$
$k := l$;
$j := 1$;
else $j := j + 1$ **end while**
End.

Algorithm 3.1 computes a decomposition of $\mathscr{A}$ that is (not necessarily properly) finer than the original decomposition given in the input and that is invariant as a whole under the action of G. The latter condition means that upon termination the primitive idempotents of an ideal $\mathscr{I}_i$ form a block under the action of G. Concerning the timing of the algorithm, we note that the number $l(m+1)+j$ always increases upon completing an iteration of the **while** loop and never exceeds $(n+1)(m+1)$. Computing $\phi_j(\mathscr{I}_s)$ and forming intersections of subspaces can be done in polynomial time; therefore Algorithm 3.1 runs in time polynomial in n, m, and $\log p$.

G preserves the direct decomposition obtained; consequently, the subgroup

$$St(G, \mathscr{I}_i) = \{\pi \in G, \pi(\mathscr{I}_i) \subseteq \mathscr{I}_i\}$$

acts transitively on the primitive idempotents of $\mathscr{I}_i$. In particular, if $\dim_{GF(p)} \mathscr{I}_i > 1$, then there exists an automorphism $\pi \in St(G, \mathscr{I}_i)$ that is not the identity of $\mathscr{I}_i$. If we can find one such π, then we can obtain a proper decomposition of $\mathscr{I}_i$ and thus a finer decomposition of $\mathscr{A}$ by using the method of Theorem 2.3. Note also that $\mathbf{P}_n$ suffices because $\dim_{GF(p)} \mathscr{I}_i$ divides n.

To find such an i and $\pi \in St(G, \mathscr{I}_i)$, we first select an element u of $\mathscr{I}_1$ that is not a scalar multiple of $1_{\mathscr{I}_1}$. Next we compute some (at most $k(m+1)$) elements from the G-orbit of u as follows.

Algorithm 3.2.
$H := \{u\}$;
while $|H| < k + 1$ **do begin**
$K := H$;
for $v \in K$ **do**
for $1 \leqq i \leqq m$ **do**
$H := H \cup \{\phi_i(v)\}$;
End.

There are at least $k+1$ elements in the G-orbit of u because G acts transitively on the set of primitive idempotents of $\mathcal{A}$. Also, every execution of the body of the **while** loop increases the size of H. These observations imply that the body of the outer loop runs at most $k+1$ times, and from this we infer that the total time required is polynomial in n, m, and $\log p$. Upon termination, there exist an i and $x \neq y \in H$ such that $x, y \in \mathcal{I}_i$. Clearly, there exists a $\pi \in G$ such that $\pi(x) = y$, and therefore $\pi \in St(G, \mathcal{I}_i)$. Moreover, such π can be found efficiently by keeping track of the automorphisms ϕ_i contributing new elements $\phi_i(v)$ to H. We can thus apply Theorem 2.3 to obtain a decomposition of $\mathcal{I}_i$ and then a finer decomposition of $\mathcal{A}$.

Starting from the initial decomposition $\mathcal{A} = \mathcal{I} \oplus \mathcal{T}$, we find that at most $\log_2 n$ execution of the method of Theorem 2.3 followed by Algorithms 3.1 and 3.2 gives the minimal ideals of $\mathcal{A}$. Since the basic ingredients (the method of Theorem 2.3 and Algorithms 3.1 and 3.2) run in polynomial time, the theorem follows. □

Remarks. (1) In the subsequent applications, the group of automorphisms we have acts regularly on the set of primitive idempotents; therefore we can find the automorphisms required by an exhaustive search of the whole group.

(2) We can dispense with $\mathbf{P}_n$ in Theorem 3.1 if we have a stronger assumption on the group of automorphisms G. If G distinguishes the primitive idempotents in the strong sense that $St(G, e_i) \neq St(G, e_j)$ if $i \neq j$, then a modification of Algorithm 3.2 can provide an initial decomposition of $\mathcal{A}$ and, later, of any other ideal from a G-stable decomposition without resorting to Theorem 2.3.

The next statement is an easy consequence of Proposition 2.2 and Theorem 3.1.

COROLLARY 3.2. *Let $f \in GF(p)[x], f(x) \mid x^p - x$ be a given polynomial. Suppose also that condition $\mathbf{P}_n$ holds and, as part of the input, we have an automorphism group $G = \langle \phi_1, \phi_2, \cdots, \phi_m \rangle$ of $\mathcal{A} = GF(p)[x]/(f)$ acting transitively on the set of primitive idempotents of $\mathcal{A}$. Then, in deterministic time $(n + m + \log p)^{O(1)}$, we can find the complete factorization of f.*

4. Polynomials with regular Galois groups. In this section, let $F \in Z[x]$,

$$F(x) = x^n + a_1 x^{n-1} + \cdots + a_n,$$

be a monic polynomial that is irreducible over Q such that $K = Q[x]/(F)$ is a Galois extension of Q. Let L denote the maximum length of the coefficients a_i. Suppose also that the discriminant $D(F)$ is not divisible by p. In this case, the reduced polynomial $f = F(\text{mod } p)$ has only simple roots (in some extension of $GF(p)$). Our objective is to factor f over $GF(p)$. This will be accomplished by using the methods developed in the previous sections.

Let D denote the integral closure of Z in K, and D_p (respectively, Z_p) denote the localization of D (respectively, Z) with respect to the prime p of Z. Let $\alpha = \alpha_1, \alpha_2, \cdots, \alpha_n$ denote the roots of $F(x)$ in K. From $K = Q(\alpha)$, we infer that there exist uniquely determined polynomials $H_1, H_2, \cdots, H_n \in Q[x]$, deg $H_i < n$ such that $\alpha_i = H_i(\alpha)$. Also, there exists a unique automorphism $\rho_i \in \text{Gal}(K/Q)$ for which $\rho_i(\alpha) = \alpha_i$.

As we have polynomial time algorithms to factor polynomials over algebraic number fields (cf. Lenstra, Lenstra, and Lovász [LLL], Lenstra [Le], Grigoryev and Chistov [CG], Landau [La]), the polynomials H_i can be found in time $(n + L)^{O(1)}$. This, in particular, implies that the H_i have size polynomial in n and L.

By [L1, Chap. 3, Prop. 16], we have that $D_p = Z_p[\alpha]$; therefore the H_i have p-integral coefficients. Now let $A_1 = A_F \in M_n(Q)$ be the companion matrix of F and, for $2 \leqq i \leqq n$, put $A_i = H_i(A_1)$. Let $\mathcal{A} \leqq M_n(Q)$ denote the matrix algebra generated by A_F over Q. As F is irreducible over Q, we have the isomorphism of Q-algebras $K \cong \mathcal{A}$.

We can assume that the isomorphism sends A_1 to α, and thus A_i to α_i, for $1 \leqq i \leqq n$. This implies that the minimal polynomial of A_i over Q is F for $2 \leqq i \leqq n$ as well. Using this isomorphism, we can describe the multiplication table of Gal (K/Q) in terms of the H_i and A_1. We have $\rho_i \circ \rho_j = \rho_k$ if and only if

$$H_k(A_1) = H_j(H_i(A_1)). \tag{4.1}$$

Now let $B_i \in M_n(GF(p))$ be the matrix obtained from A_i by reduction modulo p. This definition makes sense because the entries of A_i are p-integral. Let $\mathscr{B} \leqq M_n(GF(p))$ denote the matrix algebra generated by the matrices B_i. $\mathscr{B}$ is a commutative algebra; $\dim_{GF(p)} \mathscr{B} = n$ and the matrices B_i all have the same minimal polynomial, namely f. This follows from the fact that f is the characteristic polynomial (up to sign) of B_i. Thus $\mathscr{B}$ is semisimple. For every i, $1 \leqq i \leqq n$, there exists a unique automorphism π_i of $\mathscr{B}$ for which $\pi_i(B_1) = B_i$. The reduction of (4.1) modulo p shows that the map $\rho_i \mapsto \pi_i$ is a homomorphism from Gal (K/Q) to Aut $(\mathscr{B})$.

THEOREM 4.1. *The automorphisms* $\pi_1 = \mathrm{id}, \pi_2, \cdots, \pi_n$ *form a subgroup* G *of* Aut $(\mathscr{B})$ *isomorphic to* Gal (K/Q). *Moreover,* G *acts transitively on the set of primitive idempotents of* $\mathscr{B}$.

Proof. Let $\mathscr{A} \leqq M_n(Q)$ denote the matrix algebra generated by A_F over Q. Recall that we have an isomorphism of Q-algebras K and $\mathscr{A}$, sending α_i to A_i. Using this and the fact that the elements of Q are represented in $\mathscr{A}$ as scalar matrices, we obtain the relation

$$(-1)^{n(n-1)/2} \prod_{i \neq j} (A_j - A_i) = D(F)I,$$

where I is the identity matrix from $M_n(Q)$. Reduction modulo p gives

$$(-1)^{n(n-1)/2} \prod_{i \neq j} (B_j - B_i) = D(f)I, \tag{4.2}$$

where now I stands for the identity matrix from $M_n(GF(p))$. As $D(f) \neq 0$, we infer that the matrices B_i are all different. This implies that the action of Gal (K/Q) is faithful (and thus regular) on the set $\{B_1, B_2, \cdots, B_n\}$. This proves at once the first two statements.

Now let E be the splitting field of f and consider the E-algebra $\mathscr{C} = \mathscr{B} \otimes_{GF(p)} E$. We have $\dim_E \mathscr{C} = n$, $\mathscr{C}$ is generated as an algebra over E by the element $c = B_1 \otimes 1$, and $f(c) = 0$ holds. These facts imply that the Wedderburn decomposition of $\mathscr{C}$ is $\mathscr{C} \cong E \oplus E \oplus \cdots \oplus E$ (n times). Any $GF(p)$-automorphism π of $\mathscr{B}$ can be lifted to an E-automorphism of $\mathscr{C}$ by the law $X \otimes \beta \mapsto \pi(X) \otimes \beta$. In this way, we obtain an action of G on $\mathscr{C}$. We show first that this action is transitive on the set of primitive idempotents of $\mathscr{C}$. To this end, we tensor (4.2) with $1 \in E$, below:

$$(-1)^{n(n-1)/2} \prod_{i \neq j} (B_j \otimes 1 - B_i \otimes 1) = D(f)I \otimes 1.$$

This shows that for $i \neq j$, $B_j \otimes 1 - B_i \otimes 1$ is neither 0 nor a zero divisor in $\mathscr{C}$. Now suppose that there exists a primitive idempotent $u \in \mathscr{C}$ and an integer $j \neq 1$ such that $\pi_j(u) = u$. This would imply that $u(B_j \otimes 1 - B_1 \otimes 1) = 0$, which is a contradiction. Thus G acts semiregularly on the set of primitive idempotents of $\mathscr{C}$. This set has exactly n elements and $\#G = n$; hence the action must be regular and therefore transitive.

Next, we remark that $\mathscr{B}$ is isomorphic to the $GF(p)$ subalgebra $\mathscr{B} \otimes 1 \leqq \mathscr{C}$ via the map $X \mapsto X \otimes 1$, and this isomorphism commutes with the action of the elements of G. Thus it suffices to prove that G acts transitively on the primitive idempotents of $\mathscr{B} \otimes 1$.

For every primitive idempotent $e \in \mathscr{B} \otimes 1$, there exists a set $S(e)$ of primitive idempotents of $\mathscr{C}$ such that $e = \sum_{u \in S(e)} u$, and, if $e_1 \neq e_2$, then $S(e_1)$ and $S(e_2)$ are disjoint. Now, for $i = 1, 2$, let $u_i \in S(e_i)$ and $\pi \in G$ such that $\pi(u_1) = u_2$. As $\pi(e_1)$ is again a primitive idempotent of $\mathscr{B} \otimes 1$, the only possibility is $\pi(e_1) = e_2$. This completes the proof of the theorem. □

We have the following theorem.

THEOREM 4.2. *Let* $F \in Z[x]$,

$$F(x) = x^n + a_1 x^{n-1} + \cdots + a_n$$

be a monic irreducible polynomial over Q *such that* $K = Q[x]/(F)$ *is a Galois extension of* Q. *Let* L *denote the maximum length of the coefficients* a_i. *Suppose also that the discriminant* $D(F)$ *is not divisible by the prime* p. *Suppose further that* $\mathbf{P}_n$ *holds. Then the irreducible factors over* $GF(p)$ *of the reduced polynomial* $f = F \pmod p$ *can be found in time* $(L + n + \log p)^{O(1)}$.

Proof. The matrices A_i can be constructed in time $(L + n)^{O(1)}$. Then we obtain the B_i in time $(L + n + \log p)^{O(1)}$. These matrices provide a transitive group of automorphisms of the algebra $B(\mathscr{B})$. A basis of $B(\mathscr{B})$ can be obtained by solving a system of linear equations expressing the fact that $B(\mathscr{B})$ is the set of fixed points of the Frobenius automorphism of $\mathscr{B}$. Now Theorem 3.1 is applicable because $\dim_{GF(p)} B(\mathscr{B})$ divides n. We can find the minimal ideals of $B(\mathscr{B})$ in time $(n + \log p)^{O(1)}$. From these ideals the irreducible factors of f over $GF(p)$ can be recovered in time $(n + \log p)^{O(1)}$. The proof is complete. □

Next, we prove an *unconditional* factoring result. Let F, K, D, and p be as in the beginning of this section, and let $\mathscr{P}_i$ $(i = 1, 2, \cdots, k)$ be the prime ideals of D over p. Note that by the explicit factorization theorem of Dedekind [L1], [H1, Thm. 2.2], f has exactly k irreducible factors over $GF(p)$. Also, $D/\mathscr{P}_i$ are finite fields containing $GF(p)$, the Frobenius automorphism of $D/\mathscr{P}_i$ is induced by an element $\phi_i \in \mathrm{Gal}\,(K/Q)$, and the elements ϕ_i form a conjugacy class C in $\mathrm{Gal}\,(K/Q)$. Suppose now that the elements ϕ_i are different or, in other words, $\#C = k$. Our condition on the discriminant implies that p is unramified in K, and thus $\#C = k$ is equivalent to the statement that one (and therefore all) of the ϕ_i are self-centralizing elements of $\mathrm{Gal}\,(K/Q)$. Consider the elements $u_i = \alpha^p - \phi_i(\alpha) \in \mathscr{P}_i$. Clearly, we have that

$$u = u_1 u_2 \cdots u_k \in \mathscr{P}_1 \cdots \mathscr{P}_k = pD. \tag{4.3}$$

We show that

$$u/u_i \notin pD \quad \text{for } i = 1, \cdots, k. \tag{4.4}$$

By symmetry, it suffices to show this for $i = 1$. From $u/u_1 \in pD$ and from the primality of $\mathscr{P}_1$, we infer that $u_j \in \mathscr{P}_1$ for some $j > 1$. This implies that $\phi_1(\alpha) - \phi_j(\alpha) \in \mathscr{P}_1$, which, together with $\phi_1(\alpha) \neq \phi_j(\alpha)$, supplies contradiction to the fact that $D(F)$ is not divisible by p. To obtain algorithms, we translate (4.3) and (4.4) to the language of polynomials. Let $G_i \in Z_p[x]$, $\deg G_i < n$ be the unique polynomial for which $\phi_i(\alpha) = G_i(\alpha)$ and put $g_i(x) = x^p - G_i(x) \pmod p$. We define a map $Z_p[\alpha] \to \mathscr{B}$ by sending α to B_1. This map clearly remains surjective if we restrict the domain to D to obtain a ring-homomorphism $\eta : D \to \mathscr{B}$. We claim that $\ker \eta = pD$. Indeed, $pD \leqq \ker \eta$ is obvious. The reverse containment follows from the fact that the lattices of ideals of $\mathscr{B}$ and D/pD are isomorphic (to the lattice of all subsets of a k-element set). For $g = g_1 g_2 \cdots g_k \in GF(p)[x]$, this shows that in $\mathscr{B}$ we have that $g(B_1) = 0$ and $(g/g_i)(B_1) \neq 0$ for $i = 1, \cdots, k$; therefore g is divisible by f, and g/g_i is not divisible by f. Let $s_i =$

gcd (f, g_i). We have that lcm $(s_1, \cdots, s_k) = f$, and, for every i, there exists an irreducible factor f_i of f over $GF(p)$, which divides s_j if and only if $i = j$. Consequently, f_i and s_i are the same up to constant factors, and thus $f_1, \cdots, f_k$ are the irreducible factors of f. This suggests the following very simple algorithm to find the irreducible factorization of f.

1. Compute the polynomials H_i for $i = 1, \cdots, n$ defined at the beginning of this section.

2. Compute in $GF(p)[x]$ the polynomials $f_i = \gcd(f, h_i)$, where $h_i = x^p - H_i(x) (\mathrm{mod}\ p)$. (Note that the gcd can be computed in time $(n + \log p)^{O(1)}$ using fast exponentiation modulo f.) The nonconstant polynomials among the f_i give the irreducible factors of f. We have the following theorem.

THEOREM 4.3. *Let F, n, K, p, and D be as in the beginning of this section. Suppose that the Frobenius automorphism belonging to a prime ideal $\mathfrak{P}$ over p in D is a self-centralizing element of* Gal (K/Q). *Then we can factor f over $GF(p)$ in deterministic time $(n + L + \log p)^{O(1)}$.*

Remarks. (1) A slight modification of the method of Theorem 4.3 gives a partial factorization of f if $k > \#C > 1$.

(2) It is possible to prove Theorem 4.3 along the lines of Remark 2 after Theorem 3.1. The Frobenius automorphism of $\mathfrak{B}$ and G give a sufficiently large group of automorphisms of $\mathfrak{B}$ to find the factorization of f using the methods of § 3. We present here a considerably simpler and more efficient algorithm.

Example. If deg $F = 6$ and the Galois group is S_3, the symmetric group on 3 letters, then we can factor f over $GF(p)$ whenever it has no roots in $GF(p)$. This follows because every nonidentity element of S_3 is self-centralizing. More specifically, we consider the polynomial from [GR, p. 220]

$$F(x) = x^6 - 3x^5 + 8x^4 - 11x^3 + 8x^2 - 3x + 1.$$

We have $D(F) = -686000 = -2^4 5^3 7^3$. The factorizations

$$f(x) = (x^2 + x + 1)^3 \quad \text{for } p = 2$$

and

$$f(x) = (x^3 - x^2 + x + 1)(x^3 + x^2 - x + 1) \quad \text{for } p = 3$$

show that F is irreducible over Q. From the identities $F(x) = F(1 - x)$ and $x^6 F(1/x) = F(x)$, we see that $F(\alpha) = 0$ implies that the roots of F are

$$\alpha, \quad 1 - \alpha, \quad \frac{1}{\alpha}, \quad \frac{\alpha - 1}{\alpha}, \quad \frac{\alpha}{\alpha - 1}, \quad \frac{1}{1 - \alpha}. \tag{4.5}$$

Note that these elements must be different, for otherwise α would satisfy a quadratic equation over Q. Thus F splits in $Q(\alpha)$, and, by inspection of (4.5), we see that the Galois group of F is isomorphic to S_3.

The polynomials H_i corresponding to the automorphisms in (4.5) are $H_1(x) = x$, $H_2(x) = 1 - x$, $H_3(x) = -x^5 + 3x^4 - 8x^3 + 11x^2 - 8x + 3$, $H_4(x) = 1 - H_3(x)$, $H_5(x) = -x^5 + 2x^4 - 6x^3 + 5x^2 - 3x + 1$, and $H_6(X) = 1 - H_5(x)$. Now, for a prime p, let $h_i \in GF(p)[x]$ be the polynomial $x^p - H_i(x) (\mathrm{mod}\ p)$. For $p = 3$, the algorithm gives

$$\gcd(f, h_4) = x^3 - x^2 + x + 1, \qquad \gcd(f, h_6) = x^3 + x^2 - x + 1.$$

For $p = 23$, we obtain that

$$\gcd(f, h_2) = x^2 - x + 10, \quad \gcd(f, h_3) = x^2 + 5x + 1, \quad \gcd(f, h_5) = x^2 - 7x + 7.$$

In both cases, we have the irreducible factors of f over $GF(p)$.

5. Galois extensions. We extend the results of the preceding section to the case when # Gal $(F) >$ deg F. More precisely, let $F \in Z[x]$,

$$F(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_n$$

be a polynomial irreducible over Q. Let K denote the splitting field of F over Q and put $m = [K : Q]$. Let L denote the maximum length of the coefficients a_i. Suppose that the discriminant $D(F)$ is not divisible by the prime p. Our aim is to factor the reduced polynomial $f = F(\text{mod } p)$ in time $(m + L + \log p)^{O(1)}$. The condition on the discriminant implies that $(a_0, p) = 1$. Thus we can instead consider the monic polynomial $F^*(x) = a_0^{n-1} F(x/a_0) \in Z[x]$. This causes only a polynomial transformation in the input size, and, from the modulo p factorization of $F^*(\text{mod } p)$, the factorization of f is easily recovered. Clearly, F^* is irreducible over Q, and we have also that $(D(F^*), p) = 1$. From this point, we assume that $a_0 = 1$.

The idea is to find an integral primitive element α of K/Q such that the minimal polynomial H of α over Q has size polynomial in m and L, and such that $D(H)$ is not divisible by p. Then we can apply Theorem 4.2. First, we need some preparation.

LEMMA 5.1. *Let F_1, F_2 be monic irreducible polynomials over Z,*

$$F_1(x) = (x - \beta_1)(x - \beta_2) \cdots (x - \beta_k) \qquad \beta = \beta_1;$$

$$F_2(x) = (x - \gamma_1)(x - \gamma_2) \cdots (x - \gamma_l) \qquad \gamma = \gamma_1$$

such that $D(F_i)$ is not divisible by the prime number p for $i = 1, 2$. Suppose further that $p > (kl)^2$. Then there exists an integer s, $0 \leqq s \leqq (kl)^2$ such that $Q(\beta, \gamma) = Q(\beta + s\gamma)$, and, if H is the minimal polynomial of $\beta + s\gamma$ over Q, then the discriminant $D(H)$ is not divisible by p.

Proof. Let $K_1 \geqq Q(\beta, \gamma)$ be a finite Galois extension of Q and let $\mathcal{P}$ be a prime ideal of K_1 over p. We claim that there exists an integer s in the interval indicated such that

$$\beta_{i_1} + s\gamma_{j_1} \equiv \beta_{i_2} + s\gamma_{j_2} (\text{mod } \mathcal{P})$$

if and only if $i_1 = i_2$ and $j_1 = j_2$. Indeed, if $i_1 \neq i_2$ or $j_1 \neq j_2$, then the above congruence has, at most, one solution $s(\text{mod } \mathcal{P})$. Thus the number of forbidden residue classes is $< (kl)^2$. As the integers from the interval $[0, (kl)^2]$ are pairwise incongruent mod $\mathcal{P}$, the claim follows. Now, for this s, we consider the element $\beta + s\gamma$ and its minimal polynomial H over Q. The textbook proofs of the theorem on primitive elements (such as van der Waerden [W, § 46]) show that $\beta + s\gamma$ generates $Q(\beta, \gamma)$. We have that $H \in Z[x]$ and, using the fact that the conjugates of $\beta + s\gamma$ are all of the form $\beta_i + s\gamma_j$, we obtain that $D(H) \notin \mathcal{P}$, and hence $(D(H), p) = 1$. The proof is complete. □

We need a bound on the size of the primitive elements obtained during the computation.

LEMMA 5.2. *Let F, K, L, and m be as in the beginning of this section. Let α_1, α_2, $\cdots$, α_n be the roots of F in K and*

$$\beta = s_1\alpha_1 + s_2\alpha_2 + \cdots + s_k\alpha_k,$$

where $s_i \in Z$, $k \leqq m$ and $0 \leqq s_i \leqq m^4$. Then the length of the coefficients of the minimal polynomial of β over Q is bounded by $m(L + 1 + 6 \log_2 m) + 1$.

Proof. By [KLL, Prop. 1.3], we have that $|\alpha_i| \leqq n2^L$ (α_i is now viewed as a complex number). From this, we obtain for an arbitrary algebraic conjugate $\beta^{(j)}$ of β that $|\beta^{(j)}| \leqq m^4kn2^L$. Now using the fact that there are at most m conjugates, we obtain that the absolute value of the ith elementary symmetric polynomial of these elements is, at most,

$$\binom{m}{i}(m^4kn2^L)^m \leqq (2m^6 2^L)^m = 2^{m(L+1+6\log_2 m)}.$$

Considering the sign, the lemma follows. □

Now assume that $p > m^4$. We construct an integral primitive element α of K/Q such that the minimal polynomial H of α has size polynomial in m and L, and such that $D(H)$ is not divisible by p, as follows. Let $K_i = Q(\alpha_1, \alpha_2, \cdots, \alpha_i)$ and suppose that $\beta_i = s_1\alpha_1 + s_2\alpha_2 + \cdots + s_i\alpha_i$, $0 \leqq s_i \leqq m^4$ is a primitive element of K_i/Q; we have that $(D(H_i), p) = 1$, where H_i is the minimal polynomial of β_i over Q. For $i = 1$ $\beta_1 = \alpha_1$, $H_1 = F$ suffices. Suppose that we have already found β_i and H_i. We have $\deg(H_i) \leqq m$, and, by Lemma 5.2, the size of the coefficients of H_i is bounded by $m(L + 1 + 6\log_2 m) + 1$. Next, we factor F over K_i. Let F_{i+1} be the irreducible factor of F for which $F_{i+1}(\alpha_{i+1}) = 0$. With the help of H_i and F_{i+1}, we can compute the minimal polynomial $F^{(s)}$ of $\beta_i + s\alpha_{i+1}$ over Q for $s = 0, 1, \cdots, m^4$ until we obtain that $(D(F^{(s)}), p) = 1$ and $K_i(\beta_i + s\alpha_{i+1}) = K_{i+1}$. The latter property is attained if and only if $\deg F^{(s)} = [K_i : Q]\deg F_{i+1}$. Lemma 5.1 ensures the existence of such s. As the polynomials F_{i+1} are factors of F, their size is bounded by a polynomial of m and L. This is also the case for the polynomials $F^{(s)}$ by Lemma 5.2. If a good s is found, then we can put $\beta_{i+1} = \beta_i + s\alpha_{i+1}$ and $H_{i+1} = F^{(s)}$. The computations involved take time polynomial in m and L. Finally, if $\deg H_j = m$ is attained, then we have $K = K_j$, and we can put $\alpha = \beta_j$, $H = H_j$. We have proved the following theorem.

THEOREM 5.3. *Let F, K, L, and m be as in the beginning of this section. Suppose that p is a prime and $p > m^4$. Then we can find an integral primitive element α of K/Q such that the coefficients of the minimal polynomial H of α over Q have size bounded by $m(L + 1 + 6\log_2 m) + 1$, and such that $D(H)$ is not divisible by p. Moreover, this algorithm runs in time $(m + L)^{O(1)}$.*

Now we consider the question of factoring of $f = F(\text{mod } p)$ over $GF(p)$, where p is an arbitrary prime. If $p \leqq m^4$, then we can afford to use Berlekamp's method [B1]. Thus we can assume that $p > m^4$. In this case, Theorem 5.3 gives an integral primitive element $\alpha \in K$ and a polynomial $H \in Z[x]$ meeting the requirements of that theorem. Next, form $A_H \in M_m(Q)$, the companion matrix of H, and then the matrix $A_h \in M_m(GF(p))$ obtained by reduction of A_H modulo p. It is clear that A_h is the companion matrix of the polynomial $h \in GF(p)[x]$ obtained by modulo p reduction of H. Let $\mathcal{B}$ denote the matrix algebra generated by A_h over $GF(p)$. With the algorithm of Theorem 4.2, we can factor $h = H(\text{mod } p)$ over $GF(p)$ in time $(L + m + \log p)^{O(1)}$, provided that, for every prime divisor r of m, we have F_r and an rth nonresidue $b_r \in F_r$. In terms of the matrix algebra $\mathcal{B} \leqq M_m(GF(p))$, this means that we have found the Wedderburn decomposition of $\mathcal{B}$. Now let $\alpha_1 \in K$ be a root of F. As in § 4, we see that $\alpha_1 \in Z_p[\alpha]$. Moreover, we can find a polynomial $F_1 \in Z_p[x]$, $\deg(F_1) < m$ for which $\alpha_1 = F_1(\alpha)$ in time $(m + L)^{O(1)}$ by finding a root of F in $Q[x]/(H)$ (any of the roots suffices). Now we compute the matrix $X = F_1(A_H) \in M_m(Z_p)$. As the matrix algebra generated by A_H over Q is isomorphic to $Q[x]/(H)$, we obtain that $F(X) = 0$. This implies at once that the characteristic polynomial of X is $F^{m/n}$, up to sign. Now let $Y \in M_m(GF(p))$ be the reduction of X modulo p. Clearly, $Y \in \mathcal{B}$, the characteristic polynomial of Y is $f^{m/n}$, up to sign; and we have that $f(Y) = 0$. These imply that the minimal polynomial of Y over

$GF(p)$ is f; consequently, the matrix algebra $\mathscr{C} \leqq \mathscr{B}$ generated by Y is isomorphic to the subalgebra $\langle A_f \rangle \leqq M_n(GF(p))$ generated by the companion matrix A_f, and $Y \mapsto A_f$ can be extended to such an isomorphism. As this isomorphism is efficiently computable, we can factor f if we find the minimal ideals of the algebra $\mathscr{C}$. In doing so, the following easy lemma will be helpful.

LEMMA 5.4. *Let $\mathscr{C} \leqq \mathscr{B}$ be finite-dimensional semisimple commutative algebras over a (finite) field E. Let S denote the set of primitive idempotents of $\mathscr{B}$. For $e \in S$, we put*

$$\mathscr{I}(e) = \{x \in \mathscr{C}, xe = 0\}.$$

Then $\mathscr{I}(e)$ is an ideal of $\mathscr{C}$, and, if $\mathscr{I}(e) \neq \mathscr{C}$, then $\mathscr{I}(e)$ is a maximal ideal of $\mathscr{C}$. Moreover, for every maximal ideal $\mathscr{I}$ of $\mathscr{C}$, there exists an $e \in S$ such that $\mathscr{I} = \mathscr{I}(e)$.

Proof. It is easy to check that $\mathscr{I}(e)$ is an ideal. Let $f_1, \cdots, f_l$ be the primitive idempotents of $\mathscr{C}$. As they are idempotents of $\mathscr{B}$ as well, we can write $f_i = \sum_{e \in S_i} e$, where $S_i \subseteq S$ and $S_i \cap S_j = \varnothing$ if $i \neq j$. Thus, if $e \in S_i$, then $f_i \notin \mathscr{I}(e)$, but $f_j \in \mathscr{I}(e)$ for $i \neq j$. This shows that $\mathscr{I}(e)$ is the direct complement of the minimal ideal of $\mathscr{C}$ generated by f_i. This proves the lemma. □

Lemma 5.4 and the Wedderburn decomposition of $\mathscr{B}$ allows us to find the minimal ideals of $\mathscr{C}$ in time $(m + \log p)^{O(1)}$ as follows. First, we compute the primitive idempotents of $\mathscr{B}$ (they are characterized as the identity elements of the minimal ideals of $\mathscr{B}$ and thus obtained by solving a system of linear equations over $GF(p)$). Similarly, for a given primitive idempotent e, (a basis of) $\mathscr{I}(e)$ can be obtained by solving a system of linear equations. Lemma 5.4 shows that after $\#S \leqq m$ such steps, we have the maximal ideals of $\mathscr{C}$. If $\mathscr{I}$ is a maximal ideal of $\mathscr{C}$, then its direct complement $\mathscr{T}$ (i.e., the unique ideal $\mathscr{T}$ such that $\mathscr{C} = \mathscr{I} \oplus \mathscr{T}$) can be obtained in time $(m + \log p)^{O(1)}$ (see § 2). We note that $\mathscr{T}$ is a minimal ideal, and every minimal ideal is a complement of a maximal ideal. Thus we have the Wedderburn decomposition of $\mathscr{C}$, and this allows us to compute the Wedderburn decomposition of $\langle A_f \rangle \leqq M_n(GF(p))$, from which we obtain the irreducible factors of f over $GF(p)$ in time $(n + \log p)^{O(1)}$. We have proved the following theorem.

THEOREM 5.5. *Let $F \in Z[x]$ be an irreducible polynomial over Q. Let L denote the maximum length of the coefficients of F and let m denote the order of the Galois group of F. Suppose that the discriminant $D(F)$ is not divisible by the prime p. Assume that precondition $\mathbf{P}_m$ holds. Then we can factor the reduced polynomial $f = F \pmod p$ in time $(m + L + \log p)^{O(1)}$.*

The requirement of irreducibility of F can be dropped because Theorem 5.5 can be applied to the irreducible factors of F over Q. Note that the irreducible factors can be found in polynomial time, and their size is bounded by a polynomial of the input size (cf. [LLL], [M]).

COROLLARY 5.6. *Let $F \in Z[x]$ be a polynomial (not necessarily irreducible over Q), L be the maximum length of the coefficients of F, and m denote the degree of the splitting field of F over Q. Suppose that the discriminant $D(F)$ is not divisible by the prime p. Assume that precondition $\mathbf{P}_m$ holds. Then we can factor the reduced polynomial $f = F \pmod p$ in time $(\deg F + m + L + \log p)^{O(1)}$.*

6. Division polynomials of elliptic curves. First, we recall some facts about elliptic curves. For proofs and details, the reader is referred to Silverman [Si], Lang [L2]. We consider elliptic curves $\mathscr{E}$ defined over a field K, given by a Weierstrass equation of the form

$$y^2 = x^3 + Ax + B, \quad A, B \in K, \quad 4A^3 + 27B^2 \neq 0. \tag{6.1}$$

In the subsequent considerations, K will be either Q or a finite field. For a field $E \supseteq K$, we denote by $\mathscr{E}(E)$ the set of solutions $(x, y) \in E^2$ of the above equation, together with the ideal point P_∞ of the vertical lines of the x, y-plane. It is known that $\mathscr{E}(E)$ carries a nice Abelian group structure. The 0 element is P_∞, and the group law can be succinctly described as "three collinear points add up to zero." $\mathscr{E}$ denotes the curve (and the group) over the algebraic closure of K. For a positive integer n, let $\mathscr{E}[n]$ denote the set of elements $P \in \mathscr{E}$ for which $nP = P_\infty$. If $(n, \text{char } K) = 1$, then $\mathscr{E}[n]$ is isomorphic to the direct sum of two copies of Z/nZ. If $p = \text{char } K$, then either $\mathscr{E}[p^n] = (0)$ or $\mathscr{E}[p^n] \cong Z/p^nZ$.

We define the polynomials $\psi_m \in Z[A, B, x, y]$ inductively as follows:

$$\psi_1 = 1, \quad \psi_2 = 2y, \quad \psi_3 = 3x^4 + 6Ax^2 + 12Bx - A^2,$$

$$\psi_4 = 4y(x^6 + 5Ax^4 + 20Bx^3 - 5A^2x^2 - 4ABx - 8B^2 - A^3),$$

$$\psi_{2m+1} = \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3 \qquad (m > 1),$$

$$2y\psi_{2m} = \psi_m(\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2) \qquad (m > 1).$$

The polynomials ψ_m (for m odd) and $(2y)^{-1}\psi_m$ (for m even) belong to $Z[A, B, x, y^2]$. Thus the polynomials

$$f_m = \psi_m \quad \text{if } m \text{ is odd}, \tag{6.2}$$

$$f_m = \frac{\psi_m}{2y} \quad \text{if } m \text{ is even} \tag{6.3}$$

can be understood as elements of $Z[A, B, x]$ by eliminating y using relation (6.1). If $(\text{char } K, n) = 1$, then we have that

$$\deg f_n = \tfrac{1}{2}(n^2 - 1) \quad \text{if } n \text{ is odd}, \tag{6.4}$$

$$\deg f_n = \tfrac{1}{2}(n^2 - 4) \quad \text{if } n \text{ is even}. \tag{6.5}$$

The polynomials f_i are called *division polynomials* of $\mathscr{E}$. We record some important facts about the polynomials ψ_m and f_m in the next proposition.

PROPOSITION 6.1. *Let $P = (x, y)$ be an affine point of $\mathscr{E}$ and let n be a positive integer.*

(a) *If $P \notin \mathscr{E}[2]$, then $P \in \mathscr{E}[n]$ if and only if $f_n(x) = 0$. In particular, if $(n, \text{char } K) = 1$, then f_n has no multiple roots.*

(b) *Suppose that $P \notin \mathscr{E}[n]$. Then*

$$nP = \left(x - \frac{\psi_{n-1}\psi_{n+1}}{\psi_n^2}, \frac{\psi_{n+2}\psi_{n-1}^2 - \psi_{n-2}\psi_{n+1}^2}{4y\psi_n^3}\right).$$

(c) *Let K_n denote the extension field of K generated by the coordinates of the points of $\mathscr{E}[n]$. Then K_n/K is a Galois extension, and $\#\,\text{Gal}\,(K_n/K) \leq \#GL_2(Z/nZ) \leq n^4$.*

Now assume that $K = Q$, the field of rational numbers, and that $A, B \in Z$. For a polynomial $f \in Z[X, Y]$, we denote by $|f|$ the maximum absolute value of the coefficients of f. Also, we put $a_i = \max_{1 \leq j \leq i}\{|\psi_j|\}$.

LEMMA 6.2. *There exists an absolute effectively computable positive constant c such that $|a_n| \leq (2(|A| + |B|))^{cn^3}$ for every positive integer n.*

Proof. Using the recursion formulas for $m > 1$ and information on the degrees of the polynomials involved, we obtain that

$$|\psi_{2m+1}| \leqq 2(2m+1)^8 a_{m+2}^4 \quad \text{and} \quad |\psi_{2m}| \leqq 2(2m)^8 a_{m+2}^4.$$

Now the stated bound can be proved using induction on m. □

Remark. The above bound is far from being precise. In [L2, Chap. 2, Thm. 3.1], a better bound is obtained for A and B fixed. For our purposes, it suffices to see that the polynomials ψ_m have size polynomial m and the sizes of A and B. Lemma 6.2 implies that the polynomials f_m also have size polynomial in m and sizes of A and B.

After these preliminaries, we consider factorization problems related to elliptic curves over $GF(p)$.

THEOREM 6.3. *Suppose that we have an elliptic curve $\mathscr{E}$ over $GF(p)$, given by a Weierstrass equation of the form*

$$y^2 = x^3 + ax + b, \quad a, b \in GF(p), \quad 4a^3 + 27b^2 \neq 0.$$

Let n be a positive integer and suppose that $\mathbf{P}_{\leqq n^4}$ holds. Then we can factor the division polynomial f_n of $\mathscr{E}$ in time $(n + \log p)^{O(1)}$.

Proof. Clearly, we can assume that $n < p$; otherwise, we can use Berlekamp's method. Let $0 \leqq A, B < p$ be integers such that $a = A \pmod p$ and $b = B \pmod p$, and consider the Weierstrass equation $Y^2 = X^3 + AX + B$ over Q. By $4A^3 + 27B^2 \neq 0$, it is implied that this equation defines an elliptic curve $\mathscr{C}$ over Q. Moreover, the recursion formulas show that the division polynomial f_n can be obtained as the modulo p reduction of (the coefficients of) the corresponding division polynomial $F_n \in Z[X]$ of $\mathscr{C}$. By Lemma 6.2, the size of the coefficients of F_n is bounded by a polynomial of n and $\log p$. By Proposition 6.1(a), $n \neq p$ implies that $(D(F_n), p) = 1$, and from Proposition 6.1(c), we infer that the degree of the splitting field of F_n over Q is $\leqq n^4$. Now the theorem follows from Corollary 5.6. □

Remarks. (1) By a theorem of Serre [Si, Thm. 19.1] for an elliptic curve $\mathscr{C}$ over Q and a prime n, the typical case is $\mathrm{Gal}(Q_n/Q) \cong GL(2, Z/nZ)$. Thus $\mathrm{Gal}(Q_n/Q)$ is, in general, not solvable. This implies that Theorem 6.3 cannot be obtained from Evdokimov's theorem [E].

(2) If $\mathbf{P}_{\leqq n^4}$ holds, then we can find the points of $\mathscr{E}[n]$ in polynomial time. Indeed, the algorithm of Theorem 6.3 gives the x-coordinates, and, from the x-coordinate of a point $P \in \mathscr{E}$, the y-coordinate (and thus P itself) is obtained by solving a quadratic equation. Moreover, we can efficiently solve quadratic equations in finite extension fields of $GF(p)$ if we have a quadratic nonresidue from $GF(p)$.

The problem of factoring f_n is the elliptic analogue of the question of factoring cyclotomic polynomials. The elliptic versions of binomial equations are problems of type

(6.6) $\qquad nP = R$, where R is a given and P is an unknown point of $\mathscr{E}$.

The case where $R = P_\infty$ is covered by the preceding remark. We now give a solution for the case when $R \neq P_\infty$ and the x coordinate $x(R)$ of R is in $GF(p)$. We assume that $\mathbf{P}_{\leqq 2n^6}$ holds. As before, it suffices to consider the case where $n < p$. From $R \neq P_\infty$, we infer that, for any solution P_0 of (6.6), $P_0 \notin \mathscr{E}[n]$ holds. We can then apply the multiplication formula from Proposition 6.1(b). For the (unknown) point $P = (u, v)$, we have that

$$x(R) = x(nP) = u - \frac{\psi_{n-1}(u, v)\psi_{n+1}(u, v)}{\psi_n(u, v)^2}.$$

It suffices to give algorithms for the cases when either (i) n is odd, or (ii) $n = 2$ and $x(R) \in E$, where E is a finite extension of $GF(p)$.

We restrict our attention to finding the x-coordinate of the solutions P_0. From $x(P_0)$, we obtain two candidates for $y(P_0)$ by solving the quadratic equation arising from the Weierstrass equation of the curve. The correct value of $y(P_0)$ can then be selected by substituting into (6.6). A nice feature of the multiplication formulas is that v can be eliminated from them. Indeed, (6.2) and (6.3) gives

$$x(R) = x(nP) = u - \frac{f_{n-1}(u) f_{n+1}(u) 4(u^3 + au + b)}{f_n(u)^2} \tag{6.7}$$

for n odd, and

$$x(R) = x(nP) = u - \frac{f_{n-1}(u) f_{n+1}(u)}{4(u^3 + au + b) f_n(u)^2}$$

for n even. In particular, for $n = 2$, we have that

$$x(2P) = \frac{u^4 - 2au^2 - 8bu + b^2}{4(u^3 + au + b)}.$$

The latter formula shows that solving $2P = R$ can be done by factoring a polynomial of degree at most four over E. By [R1, Thm. 1.1], this can be done in time $(\log(\#E))^{O(1)}$, provided that $\mathbf{P}_{\leq 3}$ holds. Case (ii) is settled. We now turn to (i); i.e., we assume that n is odd and that $x(R) \in GF(p)$.

When clearing the denominators in (6.7), we obtain a polynomial equation $h_n(u) = 0$, $h_n \in GF(p)[u]$ for the first coordinates $u = x(P)$ of the points $P \in \mathscr{E}$ such that $nP = R$. Using (6.4) and (6.5), we obtain that $\deg h_n \leqq n^2$. First, we dispose of the case when $R = -R$. Then $R \in \mathscr{E}[2]$, and thus R itself, is a solution of (6.6). All the solutions can be obtained as sums of the form $R + S$, where $S \in \mathscr{E}[n]$. The elements of $\mathscr{E}[n]$ can be found using the method of Theorem 6.3, and, observing that sums of points on $\mathscr{E}$ can be computed efficiently, this case is settled.

We can therefore assume that $R \neq -R$. Now $n \neq p$ implies that (6.6) has exactly n^2 solutions (any solution can be obtained as $P_0 + S$, where P_0 is an arbitrary solution and $S \in \mathscr{E}[n]$). Using the fact that n is odd, we obtain that $x(P_0) \neq x(P_1)$ if P_0 and P_1 are two different solutions of (6.6). Indeed, $P_0 \neq P_1$ and $x(P_0) = x(P_1)$ together imply that $P_0 = -P_1$. On the other hand, P_0 and $-P_0$ cannot be both solutions of (6.6). We have the important facts that $\deg h_n = n^2$ and that h_n has no multiple roots over $GF(p)$.

Again, we consider the elliptic curve

$$\mathscr{C}: \quad Y^2 = X^3 + AX + B, \quad A, B \in Z, \qquad a = A \pmod p, \quad b = B \pmod p,$$

and $0 \leqq A, B < p$. Thus $\mathscr{E}$ is obtained by modulo p reduction of the coefficients of $\mathscr{C}$. Also, let $0 \leqq C < p$ be an integer such that $x(R) = C \pmod p$ and let $T \in \mathscr{C}$ be a point such that $x(T) = C$. Consider the equation $nP = T$ in $\mathscr{C}$, where P is an unknown point of $\mathscr{C}$. As in the case of $\mathscr{E}$, we obtain for $P = (u, v)$ a condition analogous to (6.7),

$$C = x(nP) = u - \frac{F_{n-1}(u) F_{n+1}(u) 4(u^3 + Au + B)}{F_n(u)^2}, \tag{6.8}$$

where $F_i(u) \in Z[u]$ is the division polynomial of $\mathscr{C}$ defined by (6.2) and (6.3). By clearing the denominators, we obtain an equation of the form $H_n(u) = 0$, $H_n \in Z[u]$, $h_n = H_n \pmod p$. Lemma 6.2 and (6.8) show that $|H_n|$ is bounded by a polynomial of n and $\log p$. As in the finite case, we infer that $\deg H_n = n^2$ and that the roots of H_n are

precisely the x-coordinates of the points $P \in \mathscr{C}$ such that $nP = T$. Now let $P_0 \in \mathscr{C}$ be a solution of this equation and consider the field $K' = Q(x(P_0), y(P_0))$ generated by the coordinates of P_0. We have that $[K' : Q] \leqq 2n^2$. Recall that Q_n denotes the field generated by the coordinates of the points from $\mathscr{C}[n]$. Clearly, H_n splits in the composite $K'' = Q_n K'$. From Proposition 6.1 (c), we obtain that $[K'' : Q] \leqq 2n^6$; i.e., we have a bound polynomial in n for the degree of the splitting field of H_n. We can thus apply Corollary 5.6 to factor h_n and eventually solve (6.6). We can summarize the preceding discussion in the following theorem.

THEOREM 6.4. *Suppose that we have an elliptic curve $\mathscr{E}$ over $GF(p)$, given by a Weierstrass equation of the form*

$$y^2 = x^3 + ax + b, \quad a, b \in GF(p), \quad 4a^3 + 27b^2 \neq 0.$$

Let $R \in \mathscr{E}$ be a given point such that $x(R) \in GF(p)$ and n be a positive integer. Suppose that precondition $\mathbf{P}_{\leqq 2n^6}$ holds. Then we can find the points $P \in \mathscr{E}$ satisfying $nP = R$ in time $(n + \log p)^{O(1)}$.

7. Factoring polynomials modulo special primes. Recently, von zur Gathen [G] and, independently, Mignotte and Schnorr [MS] have shown that under GRH we can factor polynomials over $GF(p)$ in deterministic polynomial time if the multiplicative group $GF(p)^*$ is smooth (i.e., the prime factors of the order are not exceeding $O(\log^c p)$ for a positive constant c). We prove a result of this type here. In our case, the subgroup of rational points over $GF(p^2)$ of an elliptic curve defined over $GF(p)$ plays the role that the multiplicative group $GF(p)^*$ had in the above results.

Let p be an odd prime and suppose that we have an elliptic curve $\mathscr{E}$ over $GF(p)$

$$y^2 = x^3 + ax + b, \quad a, b \in GF(p), \quad 4a^3 + 27b^2 \neq 0. \tag{7.1}$$

Let $\mathscr{C} = \mathscr{E}(GF(p^2))$ denote the subgroup of points of $\mathscr{E}$ rational over $GF(p^2)$ and let $k = \#\mathscr{C} = p_1^{e_1} \cdots p_r^{e_r}$ be the prime factorization of the order of $\mathscr{C}$. We remark that, for given $a, b \in GF(p)$, k can be computed in time $O(\log^9 p)$ (cf. Schoof [S]). We put $t = \max_{1 \leqq i \leqq r} \{p_i\}$.

THEOREM 7.1. *Let p, $\mathscr{E}$, t be as above. Let $f \in GF(p)[x]$, $\deg f = n$ be a polynomial to be factored and suppose that $\mathbf{P}_{\leqq 2t^6}$ holds. Then we can find the irreducible factors of f over $GF(p)$ in time $(t + n + \log p)^{O(1)}$.*

Proof. By the availability of Berlekamp's reduction [B2], it suffices to give an algorithm for the case where $f \mid x^p - x$. Also, we can assume that f is monic. Let $\alpha_1, \alpha_2, \cdots, \alpha_n \in GF(p)$ be the roots of f. The idea of the algorithm is to interpret the elements α_i as x-coordinates of affine points of $\mathscr{E}$. For $1 \leqq i \leqq n$, let P_i be a point of $\mathscr{E}$ such that $x(P_i) = \alpha_i$. In general, there are two possible choices using the fact that, for two affine points $P \neq R \in \mathscr{E}$, we have that $x(P) = x(R)$ if and only if $P = -R$. In any case, we know that $P_i \in \mathscr{C}$.

For $l = e_1 + e_2 + \cdots + e_r$, let $s_1, s_2, \cdots, s_l$ be a sequence consisting of the prime factors of $k = \#\mathscr{C}$ with the appropriate multiplicities (i.e., the sequence contains p_i exactly e_i times). For $1 \leqq j \leqq l$, we put $m_j = \prod_{1 \leqq i \leqq j} s_i$. Note that $m_l = k$.

PROPOSITION 7.2. *Let $P_0, R_0 \in \mathscr{C}$, be points such that $P_0, R_0 \neq P_\infty$ and $x(P_0) \neq x(R_0)$. For $1 \leqq j \leqq l$, we let $P_j = m_j P_0$ and $R_j = m_j R_0$. Then there exists an integer $0 \leqq i < l$ such that $x(P_i) \neq x(R_i)$, and either at least one of the points P_{i+1}, R_{i+1} is P_∞, or both of them are affine points and $x(P_{i+1}) = x(R_{i+1})$.*

Proof. Let i be the largest integer j such that P_j and R_j are both affine points and $x(P_j) \neq x(R_j)$. Clearly, such i exists and $i < l$. If neither of the points P_{i+1}, R_{i+1} is P_∞, then we must have that $x(P_{i+1}) = x(R_{i+1})$. □

Recall that for a point $P = (u, v) \in \mathscr{E} \setminus \mathscr{E}[m]$, we have the multiplication formulas

$$x(mP) = u - \frac{f_{m-1}(u) f_{m+1}(u) 4(u^3 + au + b)}{f_m(u)^2} \tag{7.2}$$

for m odd, and

$$x(2P) = \frac{u^4 - 2au^2 - 8bu + b^2}{4(u^3 + au + b)}. \tag{7.3}$$

Let $g_m(u) \in GF(p)(u)$ denote the rational functions on the right-hand sides of (7.2) and (7.3). We can compute $g_m(C)$ for a matrix $C \in M_n(GF(p))$, which is similar to a diagonal matrix, provided that the "denominator" is invertible; i.e., if $f_m(C)^2$ for m odd and $4(C^3 + aC + b)$ for $m = 2$ is an invertible matrix. In this case, if γ_i are the eigenvalues of C, then the eigenvalues of $g_m(C)$ are $g_m(\gamma_i)$.

After these preliminaries, we consider the problem of factoring f. First, we form the companion matrix $C = A_f$ of f. We try to successively compute the matrices $M_i = g_{m_i}(A_f)$, $(M_0 = A_f)$. Given M_i, we attempt to compute M_{i+1} only if M_i has no multiple eigenvalues (this holds for M_0). If M_i has multiple eigenvalues, then we take a step toward factoring f as follows. If M_i has at least two different eigenvalues, then, for the minimal polynomial g of M_i, we have that $1 < \deg g < n$, $g(x) | x^p - x$. In this case, the problem of factoring f is reduced to the same problem regarding g. Indeed, if h is a proper factor of g, then $\ker h(M_i)$ is a nontrivial invariant subspace of A_f (M_i is a polynomial of A_f and thus $A_f M_i = M_i A_f$). The amount of computation necessary to find g from M_i and a nontrivial factor of f from h is $(n + \log p)^{O(1)}$. If M_i has no two different eigenvalues, then $i > 0$ and $M_i = \operatorname{diag}(\alpha, \alpha, \cdots, \alpha)$ for some $\alpha \in GF(p)$. In this subcase, we find a point $R \in \mathscr{E}$ such that $x(R) = \alpha$. This can be done by solving in y the quadratic equation $y^2 = \alpha^3 + a\alpha + b$ in $GF(p^2)$. With this point R, we consider the equation $s_i P = R$, where P is an unknown point of $\mathscr{E}$. The solutions of this equation can be found in time $(t + \log p)^{O(1)}$ by Theorem 6.4. By our inductive hypothesis, M_{i-1} has no multiple eigenvalues. If γ is an eigenvalue of M_{i-1}, then we have that $g_{s_i}(\gamma) = \alpha$. From this, we infer that there exists a point $P \in \mathscr{E}$ such that $x(P) = \gamma$ and $s_i P = R$. We have, however, all of the points P satisfying the latter equation. Consequently, we can find the characteristic roots of M_{i-1} by substituting the x-coordinates of the points P into the characteristic polynomial h_1 of M_{i-1}. As this matrix has no multiple eigenvalues, we obtain the complete factorization of h_1 and then the complete factorization of f.

We can thus assume that M_i has no multiple eigenvalues. For $m = s_{i+1}$, we compute $N = f_m(M_i)$ if m is odd and $N = M_i^3 + aM_i + b$ if $m = 2$ (i.e., the "denominator" of $g_m(M_i)$). This computation requires time $(n + t + \log p)^{O(1)}$. If N is singular, then we distinguish two subcases: the first of them being $N = 0$. Then the eigenvalues of M_i are all roots of f_m if m odd, or $x^3 + ax + b$ if $m = 2$. By Theorem 6.3, we can find the roots of these polynomials in time $(t + \log p)^{O(1)}$ and thus find the eigenvalues of M_i. This, in turn, leads to the complete factorization of f at additional cost $(n + \log p)^{O(1)}$.

In the remaining subcase, N is singular, but $N \neq 0$. Then we compute $\operatorname{Ker} N$. This is a proper invariant subspace of A_f, and we obtain a proper factor of f by computing the characteristic polynomial of A_f in this subspace. The time required is $(n + \log p)^{O(1)}$.

Finally, we are left with the case when M_i has no multiple eigenvalues and N is nonsingular. We compute M_{i+1} by the formula $M_{i+1} = g_m(M_i)$. This takes time $(n + t + \log p)^{O(1)}$.

We have finished the description of the general step of our iteration for generating the matrix M_{i+1} from M_i. This step requires time $(n + t + \log p)^{O(1)}$ and has

three possible outcomes: (a) we compute M_{i+1}; (b) we obtain a proper (but possibly partial) factorization of f; (c) we obtain a polynomial $g \in GF(p)[x]$, $1 < \deg g < n$, $g(x) | x^p - x$ such that, from a nontrivial factor h of g, we can find a nontrivial factor of f in time $(n + \log p)^{O(1)}$.

Proposition 7.2 ensures that, after less than $l = O(\log p)$ iteration steps, we terminate (i.e., we cannot compute M_{next}) at either (b) or (c). If type (c) termination occurs, then we repeat the procedure with g in the place of f, and so on. Observe that type (c) termination can happen in a row only at most $n - 2$ times. When we arrive to a type (b) termination, then, working backward, we obtain a proper factorization of f in time $(n + \log p)^{O(1)}$, as in [R2, § 3]. Summing up, in time $(n + t + \log p)^{O(1)}$, we find a proper factorization of f. This suffices to prove the theorem. □

A concluding remark. Using a recent result of Lenstra [Len], the preconditions used throughout the paper can be weakened. We define the precondition $\mathbf{L}_d$ for a positive integer d as follows: we have polynomials $f_r \in GF(p)[x]$ for every prime $r | d$ such that $\deg f_r = r$, and f_r is irreducible over $GF(p)$. The statements of the paper remain valid if we replace $\mathbf{P}_d$ by $\mathbf{L}_d$.

Acknowledgments. I thank Gábor Ivanyos for his comments on the manuscript. I am pleased to acknowledge insightful criticism by the referee.

REFERENCES

[AMM] L. ADLEMAN, G. MILLER, AND K. MANDERS, *On taking roots in finite fields*, in Proc. 18th IEEE Symp. on Foundations of Computer Science, Providence, RI, 1977, pp. 175–178.

[B] M. BEN-OR, *Probabilistic algorithms in finite fields*, in Proc. 22nd IEEE Symp. on Foundations of Computer Science, Nashville, TN, 1981, pp. 394–398.

[B1] E. R. BERLEKAMP, *Algebraic Coding Theory*, McGraw–Hill, New York, 1968.

[B2] ———, *Factoring polynomials over large finite fields*, Math. Comp., 24 (1970), pp. 713–735.

[CG] A. L. CHISTOV AND D. YU. GRIGORYEV, *Polynomial-time factoring of the multivariable polynomials over a global field*, LOMI preprint E-5-82, Leningrad, 1982.

[CZ] D. G. CANTOR AND H. ZASSENHAUS, *On algorithms for factoring polynomials over finite fields*, Math. Comp., 36 (1981), pp. 587–592.

[E] S. A. EVDOKIMOV, *Efficient factorization of polynomials over finite fields and Generalized Riemann Hypothesis*, unpublished manuscript, 1986.

[G] J. VON ZUR GATHEN, *Factoring polynomials and primitive elements for special primes*, Theoret. Comput. Sci., 52 (1987), pp. 77–89.

[GR] B. GROSS AND D. ROHRLICH, *Some results on the Mordell–Weil group of the Jacobian of the Fermat curve*, Invent. Math., 44 (1978), pp. 201–224.

[He] I. N. HERSTEIN, *Noncummutative Rings*, Math. Assn., Washington, DC, 1968.

[H1] M. A. HUANG, *Factorization of polynomials over finite fields and factorization of primes in algebraic number fields*, in Proc. 16th ACM Symp. on Theory of Computing, Washington, DC, 1984, pp. 175–182.

[H2] ———, *Riemann hypothesis and finding roots over finite fields*, in Proc. 17th ACM Symp. on Theory of Computing, Providence, RI, 1985, pp. 121–130.

[KLL] R. KANNAN, A. K. LENSTRA, AND L. LOVÁSZ, *Polynomial factorization and nonrandomness of bits of algebraic and some transcendental numbers*, in Proc. 16th ACM Symp. on Theory of Computing, Washington, DC, 1984, pp, 191–200.

[Ke] A. KERTÉSZ, *Lectures on Artinian Rings*, Akadémiai Kiadó, Budapest, 1987.

[La] S. LANDAU, *Factoring polynomials over algebraic number fields*, SIAM J. Comput., 14 (1985), pp. 184–195.

[L1] S. LANG, *Algebraic Number Theory*, GTM 110, Springer-Verlag, Berlin, New York, 1986.

[L2] ———, *Elliptic curves Diophantine analysis*, Grundlehren der Mathematischen Wissenschaften 231, Springer-Verlag, Berlin, New York, 1978.

[Le] A. K. LENSTRA, *Factoring polynomials over algebraic number fields*, in Proc. EUROCAL, London, 1983, LNCS 162, Springer-Verlag, Berlin, New York, 1983, pp. 245–254.

[Len] H. W. LENSTRA, *Finding isomorphisms between finite fields*, preprint, 1989.
[LLL] A. K. LENSTRA, H. W. LENSTRA, AND L. LOVÁSZ, *Factoring polynomials with rational coefficients*, Math. Ann., 261 (1982), pp. 515–534.
[M] M. MIGNOTTE, *An inequality about factors of polynomials*, Math. Comput., 28 (1974), pp. 1153–1157.
[MS] M. MIGNOTTE AND C. P. SCHNORR, *Calcul deterministe des racines d'un polynome dans un corps fini*, C. R. Acad. Sci. Paris. Ser. I. Math., 306 (1988), pp. 467–472.
[Mo] R. T. MOENCK, *On the efficiency of algorithms for polynomial factoring*, Math. Comp., 31 (1977), pp. 235–250.
[P] R. S. PIERCE, *Associative Algebras*, Springer-Verlag, Berlin, New York, 1982.
[Ra] M. O. RABIN, *Probabilistic algorithms in finite fields*, SIAM J. Comput., 9 (1980), pp. 273–280.
[R1] L. RÓNYAI, *Factoring polynomials over finite fields*, J. Algorithms, 9 (1988), pp. 391–400.
[R2] ———, *Factoring polynomials modulo special primes*, Combinatorica, 9 (1989), pp. 199–206.
[R3] ———, *Computing the structure of finite algebras*, J. Symbolic Comput., 9 (1990), pp. 355–373.
[S] R. J. SCHOOF, *Elliptic curves over finite fields and the computation of square roots* mod p, Math. Comp., 44 (1985), pp. 483–494.
[Sh] D. SHANKS, *Five number-theoretic algorithms*, in Proc. 1972 Number Theory Conference, University of Colorado, Boulder, CO, 1972, pp. 217–224.
[Si] J. H. SILVERMAN, *The Arithmetic of Elliptic Curves*, GTM 106, Springer-Verlag, Berlin, New York, 1986.
[T] A. TONELLI, *Göttinger Nachrichten* (1891), pp. 344–346; *History of the Theory of Numbers*, Vol. I, L. E. Dickson, ed., Chelsea, New York, p. 215.
[W] B. L. VAN DER WAERDEN, *Algebra* I–II, Springer-Verlag, Berlin, New York, 1967, 1971.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 366–376, August 1992
© 1992 Society for Industrial and Applied Mathematics
005

A POLYNOMIAL ALGORITHM FOR THE 2-PATH PROBLEM FOR SEMICOMPLETE DIGRAPHS*

JØRGEN BANG-JENSEN† AND CARSTEN THOMASSEN‡

Abstract. This paper presents polynomially bounded algorithms for finding a cycle through any two prescribed arcs in a semicomplete digraph and for finding a cycle through any two prescribed vertices in a complete k-partite oriented graph. It is also shown that the problem of finding a maximum transitive subtournament of a tournament and the problem of finding a cycle through a prescribed arc set in a tournament are both NP-complete.

Key words. tournaments, semicomplete digraphs, k-partite tournaments, polynomial algorithms, NP-completeness, feedback vertex set, feedback arcset, 2-path problem

AMS(MOS) subject classifications. 05C20, 05C38, 05C40, 68R10

1. Introduction.

For general digraphs, it is easy to see, using standard transformations, that the following three problems are equivalent from an algorithmic point of view: (i) Given four distinct vertices u_1, u_2, v_1, v_2 in a digraph D, decide whether D has disjoint paths connecting u_1 to v_1 and u_2 to v_2; (ii) Given two arcs e_1, e_2 in a digraph D, decide whether D has a cycle through e_1 and e_2; and (iii) given two vertices u and v in a digraph D, decide whether D has a cycle through u and v.

The problem of finding a cycle through two prescribed vertices or arcs in a digraph is NP-complete, as shown by Fortune, Hopcroft, and Wyllie [5]. Hence all three of the above problems are NP-complete for general digraphs. However, if we restrict ourselves to a certain class of digraphs, then the complexity (as a measure of difficulty) of these problems can vary considerably. For example, the third problem is trivial for tournaments, while the other two, as we see below, are not quite trivial, even though they prove to be polynomially decidable. Note also that, for semicomplete digraphs (that is, digraphs with no two nonadjacent vertices) the first two problems are equivalent from an algorithmic point of view.

If $e_1 = y_1x_2$ and $e_2 = y_2x_1$ are given arcs in a digraph D, then a necessary condition for D to have a cycle through e_1 and e_2 is that for each vertex z of D, $D - z$ has a path from x_i to y_i for $i = 1$ or 2. This condition is also sufficient when D has no two disjoint cycles, as shown in [11]. In general, however, it is far from sufficient even for tournaments. Indeed, there are examples of 2-connected tournaments and of 4-connected semicomplete digraphs that contain two independent arcs not contained in a cycle [3]. Thus there seems to be no simple structural characterization of the semicomplete digraphs with no cycle through two given arcs. However, we conjecture that there exists a polynomially bounded algorithm for the k-path problem in semicomplete digraphs. The *k-path problem* is the following: Given distinct vertices $u_1, u_2, \cdots, u_k, v_1, v_2, \cdots, v_k$ in a digraph D, decide whether D has k disjoint paths $P_1, P_2, \cdots, P_k$ such that P_i is a (u_i, v_i)-path for $i = 1, \cdots, k$. We verify this here for $k = 2$. In the last section, we show that, if k is not fixed, then the problem is NP-complete.

In [5] it is shown that the k-path problem is in P for acyclic digraphs. In [10] the 2-path problem is completely solved in the case of acyclic digraphs. We also present a polynomially bounded algorithm for finding a cycle through two given vertices in

* Received by the editors April 27, 1988; accepted for publication (in revised form) June 10, 1991.
† Department of Mathematics and Computer Science, Odense University, DK-5230 Denmark.
‡ Mathematical Institute Technical University of Denmark 2800 Lyngby, Denmark.

a digraph that have the same neighbours. This shows that the third of the above problems has an easy solution in terms of a polynomial algorithm, for complete k-partite digraphs.

2. Terminology and preliminaries. Most of the notation is the same as in [2], [9], but, for completeness, we repeat most of it here, except for the very standard notation, for which we refer to [4].

A *digraph* D consists of a pair $V(D), E(D)$, where $V(D)$ is a finite set of *vertices* and $E(D)$ is a set of ordered pairs xy of vertices called *arcs.* In our definition of a digraph, we do not allow multiple arcs in the same direction between two vertices. An *oriented* graph is a digraph with no cycle of length 2. A *semicomplete* digraph is a digraph with no nonadjacent vertices. A *tournament* is an oriented graph with no nonadjacent vertices. Thus tournaments are a special subclass of the semicomplete digraphs.

If there is an arc from x to y in the digraph D, then we say that x *dominates* y, and we use the notation $x \to y$ to denote this. We also sometimes denote the arc xy by the symbol $x \to y$. For any subset A of $V(D) \cup E(D)$, $D - A$ denotes the subgraph obtained by deleting all vertices of A and their incident arcs and then deleting the arcs of A still present. We write $D - x$ instead of $D - \{x\}$ when $x \in V(D) \cup E(D)$. For a given vertex x of a digraph D, $d^+(x)$ (respectively, $d^-(x)$) denotes the number of vertices dominated by x in D (respectively, dominating x in D). We also call $d^+(x)$ (respectively, $d^-(x)$) the *outdegree* (respectively, the *indegree*) of x.

The subgraph *induced* by a vertex set A of D is defined as $D - (V(D) \setminus A)$ and is denoted by $D(A)$. We often write $x \in D$ instead of $x \in V(D)$ or $x \in E(D)$, but the meaning is always clear. A *path* is a digraph with vertex set $x_1, x_2, \cdots, x_n$ and arc set $x_1 \to x_2, x_2 \to x_3, \cdots, x_{n-1} \to x_n$ such that all the vertices and arcs shown are distinct. We call such a path an (x_1, x_n)-path and denote it by $x_1 \to x_2 \to \cdots \to x_n$. If P is a path containing a subpath from x to y, then we let $P[x, y]$ denote the part of P from x to y. A *cycle* is defined analogously. A *k-cycle* is a cycle of length k.

A *component* D' of a digraph D is a maximal subdigraph, such that, for any two vertices $x, y \in D'$, D' contains an (x, y)-path and a (y, x)-path. A digraph D is *strong* if it has only one component. D is *k-connected* if, for any set A of at most $k-1$ vertices, $D - A$ is strong.

The *local connectivity* from x to y in a digraph D is the maximum number of internally disjoint paths in D from x to y.

Let x and y be vertices of a digraph D such that there is no arc from x to y. An *(x, y)-separator of size k* is a set S of k vertices of $D - \{x, y\}$ that separates x from y in D; that is, there is no (x, y)-path in $D - S$. We also sometimes call S a *k-separator* of x and y. A k-separator S of x and y is called *trivial* if either x has outdegree zero or y has indegree zero in $D - S$.

The following theorem gives a sufficient condition, in terms of local connectivities, for the existence of disjoint (x_1, y_1)-, (x_2, y_2)-paths in a semicomplete digraph T.

THEOREM 2.1 (see [3]). *Let T be a semicomplete digraph and let x_1, x_2, y_1, y_2 be distinct vertices of T. If $T - \{x_i, y_i\}$ has three internally disjoint (x_{3-i}, y_{3-i})-paths and $T - \{x_{3-i}, y_{3-i}\}$ has two internally disjoint (x_i, y_i)-paths, for $i = 1$ or 2, then T has a pair of disjoint (x_1, y_1)-, (x_2, y_2)-paths.*

In [3] this was shown to be the best possible in the sense that "three" cannot be replaced by "two," and "two" cannot be replaced by "one."

3. Cycles through two vertices with the same neighbours. Before we describe the main algorithm, we point out that in some special cases it is easy to

decide if a digraph D has a cycle through two prescribed vertices x, y. If $x \to y$, then we just check if D has a (y,x)-path. The next result deals with the case where x and y are nonadjacent, but have the same neighbours.

THEOREM 3.1. *There exists a polynomially bounded algorithm for the following problem: Given two nonadjacent vertices x, y in a digraph D such that a vertex z in D is adjacent to x if and only if z is adjacent to y, find a cycle through x and y or show that such a cycle does not exist.*

Proof. We first assume that there are two internally disjoint (x,y)-paths P_1, P_2 and two internally disjoint (y,x)-paths Q_1, Q_2 in D. We claim that D then has a cycle through x and y. If one of P_1 and P_2 has length 2, then the union of that path and one of Q_1, Q_2 is a cycle through x and y. So assume that P_1, P_2, Q_1, Q_2 all have length at least 3. Let x' (respectively, y') be the successor of x (respectively, predecessor of y) on P_1. Then, by the assumption of the theorem, x is adjacent to y', and y is adjacent to x'. If $x \to y'$, or $x' \to y$, then we obtain a cycle through x and y as above. On the other hand, if $y' \to x$ and $y \to x'$ are arcs, then they are contained in a path P_1' from y to x such that $V(P_1') = V(P_1)$. Then $P_1' \cup P_2$ is a cycle through x and y. If the paths P_1, P_2, Q_1, Q_2 do not exist, then we can assume that P_1, P_2 do not exist, and so there is a vertex z such that $D - z$ has no path from x to y. Let $V(D) - z = A \cup B$ such that $x \in B, y \in A, A \cap B = \emptyset$ and there is no arc from B to A. Now D has a cycle through x and y if and only the following conditions are satisfied:

(i) There is a vertex $a \in A \setminus \{y\}$ such that $D - a$ has a path from z to y and $y \to a$;

(ii) There is a vertex $b \in B \setminus \{x\}$ such that $D - b$ has a path from x to z and $b \to x$.

Note that if a exists in (i), then a dominates x, and if b exists in (ii), then y dominates b. Since all the steps in the argument can be checked in polynomial time, the proof is complete. □

Note that Theorem 3.1 and the remark preceeding it imply the following result.

COROLLARY 3.2. *There exists a polynomial algorithm for deciding if two vertices in a complete k-partite oriented graph are on a common cycle (regardless of the value of k).*

The restriction of this result to the case where $k = 2$ was found by Manoussakis and Tuza [7].

4. Two technical results. We now turn to the main algorithm, which is based on Theorem 2.1. In this section, we prove a theorem that deals with a special case of the 2-path problem for semicomplete digraphs that do not satisfy the condition of Theorem 2.1. We also prove a lemma that allows us to reduce the problem to a smaller one in certain cases.

THEOREM 4.1. *Let T be a semicomplete digraph, and let x_1, x_2, y_1, y_2 be distinct vertices of T such that, for each $i = 1, 2$, there are two, but not three, internally disjoint (x_i, y_i)- paths in $T - \{x_{3-i}, y_{3-i}\}$. Suppose that all (x_i, y_i)-separators of size 2 in $T - \{x_{3-i}, y_{3-i}\}$ are trivial, for $i = 1, 2$. Then T has a pair of disjoint $(x_1, y_1)-, (x_2, y_2)$-paths.*

Proof. When we refer to indegrees and outdegrees, below, it is to be understood that, for instance, $d^-(y_1) = 2$ means that y_1 has indegree 2 in $T - \{x_2, y_2\}$. Also, when we refer to a 2-separator of x_i and y_i, it is to be understood that this is in $T - \{x_{3-i}, y_{3-i}\}$, for $i = 1, 2$.

We may assume without loss of generality that every (x_i, y_i)-path has length at least 3, since otherwise it follows easily, from the assumption of the theorem, that T

has the desired paths. Also we may assume, without loss of generality, that there is no arc $x_i \to y_{3-i}$ for $i = 1,2$.

We can assume that all (x_i, y_i)-separators of size 2 induce a 2-cycle, for $i = 1, 2$. For if, without loss of generality, $\{x, y\}$ is an (x_1, y_1)-separator of size 2, such that there is only one arc between x and y, say $x \to y$, then we add the arc $y \to x$. Now, if P_1 and P_2 are disjoint (x_1, y_1)-, (x_2, y_2)-paths in T with the arc $y \to x$ added, then P_2 does not use the arc $y \to x$, and, if P_1 uses that arc, then we can replace part of P_1 by either the arc $x_1 \to x$ or $y \to y_1$, one of which exists by the assumption that all 2-separators of x_1, y_1 are trivial. We now distinguish between four cases, depending on the half degrees of $x_i, y_i, i = 1, 2$.

Case 1. We have that $d^-(y_1), d^-(y_2) \geq 3$.

Then the assumption of the theorem implies that $d^+(x_1) = d^+(x_2) = 2$. Let r, r', a, a' be chosen such that $x_1 \to r, r'$ and $x_2 \to a, a'$. By the assumption that all (x_2, y_2)-paths have length at least 3, $y_2 \to a, a'$. Hence it follows from the fact that all the 2-separators are trivial and induce directed 2-cycles that there are three internally disjoint (a, y_2)-paths and three internally disjoint (a', y_2)-paths in $T - \{x_1, y_1, x_2\}$. Now Theorem 2.1 implies the existence of disjoint (x_1, y_1)-, (a, y_2)-paths (respectively, disjoint (x_1, y_1)-, (a', y_2)-paths) in $T - \{x_2\}$, ensuring the existence of the desired paths in T, unless a (respectively, a') is contained in an (x_1, y_1)-separator of size 2. Thus we may assume that $\{a, a'\} = \{r, r'\}$, and now the existence of the desired paths follows from Menger's theorem. Now we assume that $\min\{d^-(y_1), d^-(y_2)\} = 2$ and, by directional symmetry, $\min\{d^+(x_1), d^+(x_2)\} = 2$.

Case 2. We have that $d^-(y_i), d^+(x_{3-i}) \geq 3, i = 1$ or 2

We may assume, without loss of generality, that i = 2. Hence $d^+(x_2) = d^-(y_1) = 2$. Let a, a', s, s' be chosen such that $x_2 \to a, a'$ and $s, s' \to y_1$. As in Case 1, we can assume that $\{a, a'\} = \{s, s'\}$. $T - \{x_2, y_2, y_1\}$ has a pair of disjoint (x_1, s)-, (x_1, s')-paths P_1, P_2, by the assumption in the theorem. Let P_1 and be P_2 be chosen such that they are minimal; i.e., no proper subset of the vertices of P_i induces a semicomplete digraph containing a path with the same endvertices, $i = 1, 2$. Let P be any (x_2, y_2)-path in $T - \{x_1, y_1\}$. Go backward on P from y_2, and let u be the first vertex in $P_1 \cup P_2$ that we encounter. If $u \in \{s, s'\}$, then it is easy to see that T has the desired paths. Suppose, without loss of generality, that u is on P_1. If P_1 is the path $x_1 \to u \to s$, then T has the desired paths, since $\{u, s\}$ is not a 2-separator of x_2 and y_2. Thus we may assume that P_1 has length at least 3. Now it follows from the minimality of P_1 that $P_1 \cup \{x_2\} \cup P[u, y_2] - \{x_1\}$ contains an (x_2, y_2)-path that is disjoint from the (x_1, y_1)-path $P_2 \cup \{s' \to y_1\}$ (s dominates the successor of x_1 on P_1 and $x_2 \to s$). Now we assume that $\min\{d^+(x_1), d^-(y_2)\} = \min\{d^+(x_2), d^-(y_1)\} = 2$

Case 3. We have that $d^-(y_i) \geq 3, d^-(y_{3-i}) = d^+(x_{3-i}) = 2, i = 1$ or 2.

We may assume, without loss of generality that i = 2. By the assumption of the theorem $d^+(x_2) = 2$.Let a, a', r, r', s, s' be chosen such that $x_1 \to r, r'$, and $s, s' \to y_1$ and $x_2 \to a, a'$. As in Case 1, we can assume that $\{a, a'\} \subset \{r, r', s, s'\}$. If there is an arc from $u \in \{r, r'\}$ to $v \in \{s, s'\}$, then either the desired paths exist or $\{a, a'\} = \{u, v\}$. Hence we may assume that there is at most one such arc. Thus we may assume, without loss of generality that there are no such arcs ending in s. Suppose first that $s \notin \{a, a'\}$. Then, by the assumption of the theorem, $T - \{x_1, y_1, s\}$ contains two internally disjoint (x_2, y_2)-paths and $T - \{x_1, y_1, x_2, y_2\}$ contains three internally disjoint (r, s)-paths. Thus we conclude, by Theorem 2.1, that the desired paths exist, unless $r \in \{a, a'\}$. Similarly, $r' \in \{a, a'\}$. If $\{r, r'\} = \{a, a'\}$, however, then any two disjoint paths from $\{x_1, x_2\}$ to $\{y_1, y_2\}$ (which exist by Menger's theorem) can be

modified into the desired paths. Hence we may assume that $s \in \{a, a'\}$. Now either the desired paths exist, or there can be no arc $\alpha \to \beta$ from $\{r, r'\}$ to $\{s, s'\}$ ending in s' either, since that would imply a 2-separator of x_2, y_2 (namely, $\{\alpha, \beta\}$) that is different from $\{a, a'\}$, a contradiction. Thus, by the same argument as above, $s' \in \{a, a'\}$; i.e., $\{a, a'\} = \{s, s'\}$. Now we can argue as in Case 2 to show that the desired paths exist in this case. Now, by directional symmetry, there only remains one case.

Case 4. We have that $d^+(x_1) = d^-(y_1) = d^+(x_2) = d^-(y_2) = 2$

Let $a, a', b, b', r, r', s, s'$ be chosen such that $x_1 \to r, r'$, and $s, s' \to y_1$ and $x_2 \to a, a'$, and $b, b' \to y_2$.

Suppose first that $\{a, a', b, b'\} = \{r, r', s, s'\}$. If $\{a, a'\}$ equals $\{r, r'\}$ (respectively, $\{s, s'\}$), then we conclude that T has the desired paths using the same arguments as we did in in Case 1 (respectively, Case 2). Otherwise, we obtain an (x_i, y_i)-path of length 3 for $i = 1$ or 2, and it follows from the assumption of the theorem that this path does not separate x_{3-i} from y_{3-i}.

Thus we may assume that $\{a, a', b, b'\} \neq \{r, r', s, s'\}$. By symmetry and directional symmetry, we may assume that $s \notin \{a, a', b, b'\}$.

Then the assumption of the theorem implies that there are two internally disjoint (x_2, y_2)-paths in $T - \{x_1, y_1, s\}$, since s is not in any 2-separator of x_2, y_2. Also, we may assume that $s \to r, r'$, since otherwise the existence of the desired paths follows from the fact that s is not in a 2 separator of x_2, y_2. By assumption of the theorem and the fact that all 2-separators induce a 2-cycle, there are three internally disjoint (r, s)-paths (respectively, (r', s)-paths) in $T - \{x_1, x_2, y_1, y_2\}$. We now distinguish between two further subcases.

Case A. We have that $\{r, r'\} \not\subset \{a, a', b, b'\}$

Say, without loss of generality, that $r \notin \{a, a', b, b'\}$. Let $T' = T - \{x_1, y_1\}$. As we have seen, $T' - \{x_2, y_2\}$ has three internally disjoint (r, s)-paths. If $T' - \{r, s\}$ has two internally disjoint (x_2, y_2)-paths, then Theorem 2.1 implies the existence of disjoint (r, s)-, (x_2, y_2)-paths in T', and hence T has the desired paths. Otherwise, there exists a vertex z such that $T' - \{r, s, z\}$ has no (x_2, y_2)-path. Now $T' - r$ has an (x_2, s)-path L_1 and an (x_2, z)-path L_2 such that $L_1 \cap L_2 = \{x_2\}$ by Menger's theorem and the fact that r is not in any 2-separator of x_2, y_2. Similarly, since s is not in any 2-separator of x_2, y_2, $T' - s$ has an (r, y_2)-path L_3 and a (z, y_2)-path L_4 such that $L_3 \cap L_4 = \{y_2\}$. Since the last vertex of $L_3 - y_2$ dominates the first vertex of $L_1 - x_2$, T has an (x_1, y_1)-path disjoint from $L_2 \cup L_4$, showing that T has the desired paths (L_3 is not just an arc, since r does not dominate y_2. Similarly, L_1 is not just an arc, since x_2 does not dominate s).

Case B. We have that $\{r, r'\} \subset \{a, a', b, b'\}$.

If $\{r, r'\} = \{a, a'\}$ (respectively, $\{r, r'\} = \{b, b'\}$) then we conclude as in Case 1 (respectively, Case 2) that T has the desired paths. Hence we may assume, without loss of generality, that $\{r, r'\} = \{a, b\}$. Since $s \notin \{a, a', b, b'\}$, there is one of a, a', b, b' not in $\{r, r', s, s'\}$. If $b' \notin \{r, r', s, s'\}$, then we conclude, as above, that $\{a, a'\} = \{r, s'\}$ (or $\{a, a'\} = \{r', s'\}$, in which case the proof is analogous to the proof given below). Now, however, the arc between $r' = b$ and $s' = a'$ implies an (x_i, y_i)-path of length 3 for $i = 1$ or 2, implying the existence of the desired paths, by the assumption of the theorem (r', s' is not a 2-separator of any of the pairs x_i, y_i, $i = 1, 2$). Thus we may assume that $b' \in \{r, r', s, s'\}$, which implies that $b' = s'$. (We choose the notation such that $r = a, r' = b$.)

If $b \to s$, then T has the desired paths, since $r' = b$ and $\{r', s\}$ is not a 2-seperator of x_2, y_2. So we may assume that $s \to b$.

Suppose first that $a' \to s$. Then T has the desired paths with $x_2 \to a' \to s \to b \to y_2$ being one of them, unless $\{a', s, b\}$ is an (x_1, y_1)-separator. Suppose that $\{a', s, b\}$ is an (x_1, y_1)-separator. Then x_1 and a (respectively, b' and y_2) are not separated by $\{a', s, b\}$. Let R_1 be an (a, s)-path in $T - \{x_2, y_2, a', b\}$, and let R_2' be an (x_1, y_1)-path in $T - \{x_2, y_2, b, s\}$. Then R_2' contains a' and an (a', b')-path R_2. Since $R_1 \cap R_2 = \emptyset$, they can be extended to the desired paths in T (since $x_1 \to a, s \to y_1, x_2 \to a', b' \to y_2$).

Suppose now that $s \to a'$. Let $T'' = T - \{x_1, y_1, x_2, y_2\}$. By Menger's theorem and the assumption of the theorem, $T'' - b$ has two internally disjoint (a, b')-, (a', b')-paths and two internally disjoint (r, s)-, (r, s')-paths. Also, $b \to s'$, since $\{b, s'\}$ is a 2-separator of x_2, y_2, and we have assumed that all these form 2-cycles. Now it is easy to see that T'' has three paths from $\{a, b, a'\}$ to $\{s, s'\}$ that are disjoint, except that two of them contain s'. Two of these can be extended to the desired paths. (If the path that ends in s starts in a', then T'' has disjoint (a', b)-, (r, s')-paths, because $s \to b$.) Hence, if $s \notin \{a, a', b, b'\}$, then T has the desired paths. This completes the proof of the theorem. □

LEMMA 4.2. *Let T be a semicomplete digraph, and let x_1, x_2, y_1, y_2 be distinct vertices such that there are two internally disjoint (x_2, y_2)-paths in $T - \{x_1, y_1\}$. Suppose that there exists a nontrivial 2-separator $\{x, y\}$ of x_2 and y_2 in $T - \{x_1, y_1\}$ such that there is no arc from $B - x_2$ to y_1, where A and B form any patition of $T - \{x_1, y_1, x, y\}$ such that $x_2 \in B$, $y_2 \in A$, and all arcs between A and B go from A to B.*

Transform T into a new semicomplete digraph T' as follows:

1. *If x_1 dominates some vertex in $A - y_2$ then*
   - *If there exists a vertex $b \in B - x_2$ such that $b \to x$ and there is an (x_2, y)-path in $T(B \cup \{y\} \setminus \{b\})$, then add all arcs from $A - y_2$ to x that are not present already.*
   - *If there exists a vertex $b \in B - x_2$ such that $b \to y$ and there is an (x_2, x)-path in $T(B \cup \{x\} \setminus \{b\})$, then add all arcs from $A - y_2$ to y that are not present already.*
2. *Add the arcs $x_2 \to x, x_2 \to y$ if they are not present already;*
3. *Add the arc $x_1 \to z$ for $\{z, w\} = \{x, y\}$ if $T(B \cup \{z, w, x_1\})$ has a pair of disjoint (x_1, z)-, (x_2, w)-paths, and the arc $x_1 \to z$ is not present already;*
4. *Delete the vertices of $B - x_2$.*

Call the added *arcs* special arcs. *Then the resulting semicomplete digraph T' has disjoint (x_1, y_1)-, (x_2, y_2)-paths if and only if T also has them.*

Proof. Suppose that P and Q are disjoint $(x_1, y_1)-, (x_2, y_2)$-paths in T chosen such that they are minimal; i.e., no proper subset of P or Q is a path from x_i to y_i for $i = 1$ or 2. If $x, y \in V(Q)$, then P is entirely in $T(A \cup \{x_1, y_1\})$ since there is no arc from B to y_1. Let z be that of x, y that is closest to y_2 on Q. Then P is in T', and $Q' = \{x_2 \to z\} \cup Q[z, y_2]$ is in T'. Now suppose that Q contains only one of x, y, and let z denote that one, and w the other. Then $Q[z, y_2] \cap B = \emptyset$. If P does not intersect B, then P and $Q' = \{x_2 \to z\} \cup Q[z, y_2]$ are in T'. Thus we can assume that P intersects B. Then $w \in V(P)$ since there are no arcs from B to y_1. If $P[x_1, w]$ intersects A, then the minimality of P implies that P contains the sequence $a \to b \to w$ for some $a \in A, b \in B$. Let $P' = P[x_1, a] \cup \{a \to w\} \cup P[w, y_1]$ and $Q' = \{x_2 \to z\} \cup Q[z, y_2]$. Then P', Q' are the desired paths in T'. If $P[x_1, w]$ does not intersect A, then we let $P' = \{x_1 \to w\} \cup P[w, y_1]$ and take Q' as above. Thus if T has the desired paths, then so has T'.

Conversely, suppose that P', Q' are disjoint (x_1, y_1)-, (x_2, y_2)-paths in T', chosen such that they are minimal. We may assume, without loss of generality, that $P' \cup Q'$ contains at least one special arc. By the minimality of Q', exactly one of x, y belongs to Q'. Let z be that one, and w the other. If P' contains the special arc $x_1 \to w$, then, by the definition of the special arcs, this is the only special arc in P', and $T(B \cup \{x_1, w, z\})$ has a pair of disjoint (x_1, w)-, (x_2, z)-paths P^*, Q^*. Then $Q^* \cup Q'[z, y_2]$ and $P^* \cup P'[w, y_1]$ are the desired paths in T. If P' contains a special arc of the form $a \to w$ for some $a \in A$, then we know, from the definition of the special arcs, that there exists a vertex $b \in B$, dominating w such that $T((B \cup \{z\}) \setminus \{b\})$ has an (x_2, z)-path Q^*. Let $Q = Q^* \cup Q'[z, y_2]$ and $P = P'[x_1, a] \cup \{a \to b \to w\} \cup P'[w, y_1]$. Then P, Q are the desired paths in T. Finally, if P' contains no special arcs, then P' and $Q'[z, y_2]$, together with some (x_2, z)-path in $T(B \cup \{z\})$, are the desired paths. This proves the lemma. □

5. A polynomial algorithm for the 2-path problem for semicomplete digraphs. The idea in the algorithm is either to settle the problem, or else to reduce the problem to a smaller one (i.e., construct a semicomplete digraph S such that the desired paths exist in T if and only if there exist some corresponding paths in S), and then let the algorithm call itself recursively. The crucial step in the algorithm is when the local connectivity from x_i to y_i is precisely 2 for $i = 1, 2$. If all 2-separators of x_i, y_i are trivial for $i = 1, 2$, then no reduction is possible. Fortunately, this is precisely the situation in Theorem 4.1, and we can conclude that the desired paths exist. In the case when a nontrivial 2-separator exists, say, for x_2, y_2, we show how to use the structure of the nontrivial 2-separators of x_2, y_2 to decide the existence of the desired paths, or to reduce the problem to a smaller one.

THEOREM 5.1. *There exists a polynomial algorithm for the following problem for semicomplete digraphs: Let T be a semicomplete digraph and x_1, x_2, y_1, y_2 be four different vertices of T. Decide whether T has a pair of disjoint (x_1, y_1)-, (x_2, y_2)-paths.*

Proof. Clearly, if there is no (x_i, y_i)-path in $T - \{x_{3-i}, y_{3-i}\}$, $i = 1$ or 2, then the desired paths do not exist. This is easy to check, and we may thus assume that there is an (x_i, y_i)-path in $T - \{x_{3-i}, y_{3-i}\}$ for $i = 1, 2$. If T is not strong, then it is easy to see that the desired paths exist, except possibly when x_1, x_2, y_1, y_2 all belong to the same strong component of T. Then we can reduce the problem to that component. Since it is easy to find the strong components of T, we can thus assume that T is strong. We can also assume that T contains none of the arcs $x_1 \to y_1, x_2 \to y_2$, since otherwise the desired paths exist.

From Theorem 2.1, we know that, if $T - \{x_i, y_i\}$ has three internally disjoint (x_{3-i}, y_{3-i})-paths and $T - \{x_{3-i}, y_{3-i}\}$ has two internally disjoint (x_i, y_i)-paths, for i=1 or 2, then T has the desired paths. Thus the algorithm first checks this and stops if this condition is met. We now distinguish between two cases.

Case 1. The local connectivity from x_i to y_i in $T - \{x_{3-i}, y_{3-i}\}$ is 1 for i=1 or 2.

Without loss of generality, we may assume that i=2. By Menger's theorem, there exists an $x \in T - \{x_1, y_1\}$ such that there is no (x_2, y_2)-path in $T - \{x, x_1, y_1\}$. We can assume that there is an (x_1, y_1)-path in $T - \{x, x_2, y_2\}$, since otherwise the desired paths do not exist in T. For a given x, which separates x_2 from y_2 in $T - \{x_1, y_1\}$, we define A and B as follows:

$$A = \{v \in T - \{x\} \mid v \text{ can reach } y_2 \text{ by a path in } T - \{x, x_1, y_1\}\},$$

$$B = V(T) - A - \{x, x_1, y_1\}.$$

Then all the arcs between A and B are leaving A.

If A contains a vertex a dominated by x_1 such that $T-\{x_1,y_1,a\}$ has an (x,y_2)-path P_1, and B contains a vertex b dominating y_1 such that $T-\{x_1,y_1,b\}$ has an (x_2,x)-path P_2, then $P_1\cup P_2$ and $x_1\to a\to b\to y_1$ are the desired paths. Suppose that a does not exist. Now the desired paths exist if and only if $T(B\cup\{x,x_1,y_1\})$ contains a pair of disjoint $(x_1,y_1)-,(x_2,x)$-paths. Thus we have reduced the problem to a smaller one. The case where b does not exist is analogous.

Case 2. There exist two internally disjoint (x_i,y_i)-paths in $T-\{x_{3-i},y_{3-i}\}$ for i=1, 2.

If all 2-separators of x_1 and y_1 in $T-\{x_2,y_2\}$ and all 2-separators of x_2 and y_2 in $T-\{x_1,y_1\}$ are trivial, then it follows from Theorem 4.1 that T has the desired paths. Hence we may assume, by renaming if necessary, that there exists a nontrivial 2-separator $\{x,y\}$ of x_2 and y_2 in $T-\{x_1,y_1\}$. Define A,B as follows:

$$A=\{v\in T-\{x_1,y_1\}\mid \text{v can reach } y_2 \text{ by a path in } T-\{x,y,x_1,y_1\}\},$$

$$B=T-A-\{x_1,y_1,x,y\}.$$

Then all arcs between A and B go from A to B, and neither x_1 nor y_1 belong to $A\cup B$. Since $\{x,y\}$ is a nontrivial 2-separator, we have that $|A|,|B|\geq 2$. We can assume, without loss of generality, that $x\to y$. There are two subcases to consider.

Subcase 2.1. There are no arcs from x_1 to $A-y_2$, or there are no arcs from $B-x_2$ to y_1.

We may assume that the latter case holds, since the former case can be treated similarly (using an analogous version of Lemma 4.2 to contract A into one vertex, just as shown below for B in the latter case). Our next step is to contract B into one vertex x_2. First, we check, using the flow version of Menger's theorem (see, e.g., [4]), whether there exist disjoint (x_1,z)-, (x_2,w)-paths in $T(B\cup\{x_1,x,y\})$, where $\{z,w\}=\{x,y\}$ (using the flow version, we can actually find these paths if they exist). If T has such paths, then we apply our algorithm to $T(B\cup\{x_1,x,y\})$ to decide whether $T(B\cup\{x_1,x,y\})$ also has disjoint (x_1,w)-, (x_2,z)-paths. If $T(B\cup\{x_1,x,y\})$ also has these paths, then we conclude that the desired paths exist. This follows, since there is no arc from $B-x_2$ to y_1 and we know that there are two internally disjoint paths P_1,P_2 from $\{x,y\}$ to y_2 in $T(A\cup\{x,y\})$. Also, $T-y_2$ has a path P from x_1 to y_1. Going backward on P until we meet a vertex in one of the previously mentioned paths, P_1,P_2, or x_1 gives a configuration that contains the desired paths: Suppose that the first vertex of $P_1\cup P_2$ that we encounter when going backward from y_1 on P is the vertex u on P_1. Let Q_1,Q_2 be disjoint (x_1,x)-, (x_2,y)-paths in $T(B\cup\{x_1,x,y\})$. Then $Q_1\cup P_1[x,u]\cup P[u,y_1]$ and $Q_2\cup P_2$ are the desired paths. Thus we may assume that $T(B\cup\{x_1,x,y\})$ does not have both pairs of paths.

Now we contract B to $\{x_2\}$ giving a new semicomplete digraph T', as described in Lemma 4.2. By Lemma 4.2, the resulting semicomplete digraph T' has disjoint (x_1,y_1)-, (x_2,y_2)-paths if and only T also has them. Thus we have reduced the problem to a smaller one.

Subcase 2.2. $x_1\to r$ for some $r\in A-y_2$, and $s\to y_1$ for some $s\in B-x_2$.

Then T contains the path $x_1\to r\to s\to y_1$. If $T-\{x_1,r,s,y_1\}$ has an (x_2,y_2)-path, then T has the desired paths. Hence we may assume that this is not the case. Then $\{r,s\}$ must separate x_2 from y_2 in $T-\{x_1,y_1\}$. Since the local connectivity from x_2 to y_2 in $T-\{x_1,y_1\}$ is 2, r does not destroy all paths from $\{x,y\}$ to y_2 in $T(A\cup\{x,y\})$, and s does not destroy all paths from x_2 to $\{x,y\}$ in $T(\{x,y\}\cup B)$. Thus, since $x\to y$

and $T - \{x_1, r, s, y_1\}$ has no (x_2, y_2)-path, we conclude that in $T(A \cup \{x, y\} - \{r\})$ there is an (x, y_2)-path but no (y, y_2)-path, and in $T(B \cup \{x, y\} - \{s\})$ there is an (x_2, y)-path, but no (x_2, x)-path. Thus $\{r, s\}$ is also a nontrivial 2-separator of x_2 and y_2 in $T - \{x_1, y_1\}$ (x is included in the new A and y in the new B).

Now look at the sets A' and B' that correspond to $\{r, s\}$ as A and B correspond to $\{x, y\}$ (the remainder of the discussion deals only with A', B'; hence the reader may disregard that x, y, A, B). If there is no arc from x_1 to $A' - y_2$ or there is no arc from $B' - x_2$ to y_1, then we are in Case 2.1, where we can settle the problem or reduce. Hence we may assume that we are in Case 2.2 with A' and B' instead of A, B. Thus we have some $r' \in A' - y_2$ and some $s' \in B' - x_2$ such that $x_1 \to r' \to s' \to y_1$ is a path in T. Again, we may assume that there is no (x_2, y_2)-path in $T - \{x_1, r', s', y_1\}$. As above, $T(A' \cup \{r, s\} - \{r'\})$ has an (r, y_2)-path but no (s, y_2)-path, and in $T(B' \cup \{r, s\} - \{s'\})$ there is an (x_2, s)-path, but no (x_2, r)-path. If $r \to s'$ or $r' \to s$, then it is easy to see that T contains the desired paths (for example, if $r \to s'$, then there is an (x_2, y_2)-path in $T - \{x_1, r, s', y_1\}$ since s' does not separate x_2 from s in $T(B' \cup \{r, s\})$, and r does not separate s from y_2 in $T(A' \cup \{r, s\})$). So assume that these arcs do not exist. Also, we must have that $x_2 \to s'$ and $r' \to y_2$: Suppose that there is no arc from x_2 to s'. Let P_1 and P_2 be internally disjoint (x_2, r)- , (x_2, s)-paths in $T(B' \cup \{r, s\})$. By the above argument, P_1 contains s', and, by the assumption, $P_1[x_2, s']$ has length at least 2. Let u be the first vertex after x_2 on $P_1[x_2, s']$. We cannot have that $u \to r$, since then s' would not separate x_2 from r in $T(B' \cup \{r, s\})$ as assumed above. Thus $r \to u$, and now we easily find an (x_2, y_2)-path that avoids the (x_1, y_1)-path $x_1 \to r \to u \cup P_1[u, s'] \cup \{s' \to y_1\}$ (when the vertices of this path are removed, there is still a path from x_2 to s and then to y_2, namely, P_2 and any (s, y_2)-path in $T(A' \cup \{r, s\})$). Similarly, we get that $r' \to y_2$. Now if we look at the nontrivial 2-separator $\{r', s'\}$, we can argue similarly that either T has the desired paths, or $x_2 \to s$ and $r \to y_2$. Hence we may assume that we have the following paths in T:

$$x_1 \to r \to s \to y_1, x_1 \to r' \to s' \to y_1, x_2 \to s \to r' \to y_2, x_2 \to s' \to r \to y_2.$$

Now any (x_1, y_1)-path in $T - \{x_2, y_2\}$ must contain at least two of the vertices r, r', s, s', to destroy both of the above (x_2, y_2)-paths. On the other hand, no minimal (x_1, y_1)-path that does not start with $x_1 \to r$ or $x_1 \to r'$ contains more than one of these four vertices, since $\{s, s'\} \to y_1$. Thus, if $\{r, r'\}$ is not a 2-separator of x_1 and y_1 in $T - \{x_2, y_2\}$, then T contains the desired paths. Hence we may assume that $\{r, r'\}$, and, similarly, $\{s, s'\}$ are 2-separators of x_1 and y_1 in $T - \{x_2, y_2\}$. That is, all minimal (x_1, y_1)-paths start with one of the arcs $x_1 \to r$, $x_1 \to r'$ and end with one of the arcs $s \to y_1, s' \to y_1$. Now it is easy to see that T contains the desired paths if and only if at least one of $T - \{x_2, s, r', y_2\}$, $T - \{x_2, s', r, y_2\}$ contains an (x_1, y_1)-path. (Remember that we have assumed that there is no (x_2, y_2)-path in $T - \{x_1, r, s, y_1\}$ or $T - \{x_1, r', s', y_1\}$).

This completes the proof of the theorem, since it is easy to see that all our actions can be done in a polynomially bounded number of steps, and, for each application of the algorithm, the number of vertices decreases before the next time the algorithm is applied. In fact, the complexity of the algorithm is of order $O(n^5)$: The analysis of all separating sets of size 2 takes $O(n^4)$ time, the flow calculation and all other actions at most $O(n^3)$ time. Hence, between two consecutive calls of the algorithm we spend at most $O(n^4)$ time. Thus, if $T(n)$ denotes the time complexity of our algorithm, then $T(n) \leq T(n - x + 1) + T(x + 3) + O(n^4)$, where x denotes the size of B in Subcase 2.1. This implies that $T(n) = O(n^5)$. Above n denotes the number of vertices in the semicomplete digraph. □

We make no claim as to the optimality of this algorithm. Our sole objective has been to demonstrate that the problem is polynomially solvable for semicomplete digraphs. However, we feel that an algorithm with a significantly lower time complexity is considerably more complicated to describe. This is supported by the fact that there exist highly nontrivial families of 4-connected semicomplete digraphs that are not 2-linked (i.e., in such a digraph, there exist four vertices u, v, x, y such that there do not exist disjoint $(u, v)-, (x, y)-$paths). Hence it is very unlikely that a simple algorithm exists, since this would imply a simple characterization of those semicomplete digraphs that are not 2-linked (see also §1, p. 366).

Note that it is easy to use this algorithm to obtain a polynomial algorithm that finds the desired paths, given that they exist: Successively, reverse all arcs out of x_1 and, each time, call the algorithm to check for the existence of the desired paths. If an arc is identified, whose reversal destroys the property of having the desired paths, then this arc must be part of such a pair of paths in the current semicomplete digraph. Suppose this is the arc $x_1 \to u$. Now remove x_1, rename u by x_1, and repeat the above step for the new x_1. Eventually, we get that the current x_1 dominates y_1 and that the current semicomplete digraph T contains an (x_2, y_2)-path that does not contain any of x_1 and y_1. Hence the (x_1, y_1)-path given by the sequence of successively identified vertices and any (x_2, y_2)-path in the final semicomplete digraph with x_1, y_1 removed is a pair of disjoint paths as desired.

COROLLARY 5.2. *There exists a polynomal algorithm for the following problem for semicomplete digraphs. Let T be a semicomplete digraph and let x, y, z be distinct vertices of T. Decide whether T has an (x, z)-path through y.*

Proof. This correponds to the situation $y_1 = x_2$ in the two path problem. In the above proof, we required that x_1, x_2, y_1, y_2 are distinct vertices, but it is easy to see that this special case of the 2-path problem reduces to the general case in polynomial time within the class of semicomplete digraphs. Hence the result follows. Details are given in [1]. □

6. Two NP-complete problems on tournaments. There are many NP-complete problems on graphs. One example is the problem of finding a Hamiltonian cycle. That problem has an easy solution for tournaments, and not many "natural" NP-complete problems on tournaments are known. (Since every oriented graph is a subgraph of a tournament, it is, of course, possible to formulate a number of artificial NP-complete problems involving (subgraphs of) tournaments.) We mention here two basic NP-complete tournament problems.

THEOREM 6.1. *The problem q of finding a cycle through a prescribed arc set in a tournament is* NP*-complete.*

Proof. To see this, we reduce the problem of finding a Hamiltonian cycle in a directed graph to q. Let D be a directed graph. First, we split every vertex v of D into two vertices v_1 and v_2 such that all arcs entering v (respectively, leaving v) now enter v_1 (respectively, leave v_2). We also add the "distinguished" arc v_1v_2. This transforms D into a bipartite digraph D'. We add arcs from vertices of index 1 to vertices of index 2 in D (whenever the arcs in the opposite direction are not already present). We add all arcs between vertices of the same index and orient them at random. Then the resulting tournament has a cycle through the distinguished arcs if and only if D has a Hamiltonian cycle. □

This proves that, if k is not fixed, then the k-path problem NP-complete for tournaments.

THEOREM 6.2. *The problem p_1 of finding a largest transitive subtournament*

(or, equivalently, the problem of finding a smallest vertex set meeting all cycles) is NP*-complete.*

Proof. It is well known that the problem p_2 of finding a largest set of independent (i.e., pairwise nonadjacent) vertices in an undirected graph is NP-complete. Now we reduce p_2 to p_1 by a polynomial time reduction. Let G be an undirected graph with vertex set $v_{1,0}, v_{2,0}, \cdots, v_{n,0}$. We form a tournament T as follows : We add, for each $i = 1, 2, \cdots, n$, a set of n+1 new vertices $v_{i,1}, v_{i,2}, \cdots, v_{i,n+1}$. Now T contains the directed arcs $v_{i,k}v_{j,m}$ whenever $i > j$ or $i = j$ and $k > m$, unless $k = m = 0$ and $v_{i,0}, v_{j,0}$ are adjacent in G. In that case, T contains the arc $v_{j,0}v_{i,0}$. Now a vertex set S in G is a largest independent set if and only if $T - (V(G) \setminus S)$ is a largest transitive subtournament of T. □

Note that problem of finding a feedback vertex set, i.e., a minimum set of vertices such that every directed cycle is incident with this set, corresponds to p_1 for tournaments; hence Feedback vertex set is NP-complete for tournaments.

CONJECTURE 6.3. *The problem of finding a minimum set of arcs in a tournament whose reversal results in a transitive tournament is* NP*-complete.*

Such a set of arcs is also called a feedback arc set. This problem is known to be NP-complete for general digraphs [6, p. 192].

Note added in proof. After the submission of this manuscript, the second author [12] has shown that for general digraphs there is no degree of connectivity that will ensure that a digraph is 2-linked, i.e., has disjoint (u, v)-, (x, y)-paths for any choice of distinct vertices u, v, x, y.

We also want to point out that the result of Theorem 6.2 was found independently by Speckenmeyer; see [8].

REFERENCES

[1] J. BANG-JENSEN, *A note on a special case of the 2-path problem for semicomplete digraphs*, in Graph Theory, Combinatorics and Applications, in Proc. of the Sixth Quadrennial International Conference on the Theory and Applications of Graphs, Alavi, Chartrand, Oellermann, and Schwenk, eds., John Wiley, New York, 1991.

[2] ———, *Edge-disjoint in- and out-branchings in tournaments and related path problems*, J. Combin. Theory Ser. B, 51 (1991), pp. 1–23.

[3] ———, *On the 2-linkage problem for semicomplete digraphs*, in Graph Theory in the Memory of G. A. Dirac, Ann. Discrete Math., 41 (1989), pp. 23–37

[4] J. A. BONDY AND U. S. R. MURTY, *Graph Theory with Applications*, MacMillan, New York, 1976.

[5] S. FORTUNE, J. HOPCROFT, AND J. WYLLIE, *The directed subgraph homeomorphism problem*, Theoret. Comput. Sci., 10 (1980), pp. 111–121.

[6] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability*, W. H. Freeman, San Francisco, 1979.

[7] Y. MANOUSSAKIS AND Z. TUZA, *Polynomial algorithms for finding cycles and paths in bipartite tournaments*, SIAM J. Discrete Math., (1990) pp. 537–543.

[8] E. SPECKENMEYER, *On feedback problems in digraphs*, in Proc. 15 WG 89, Springer-Verlag, Lecture Notes in Comput. Sci., 411 (1989), pp. 218–231.

[9] C. THOMASSEN, *Connectivity in Tournaments*, in Graph Theory and Combinatorics, Academic Press, New York, 1984, pp. 305–313.

[10] ———, *The 2-linkage problem for acyclic digraphs*, Discrete Math., 55 (1985), pp. 73–87.

[11] ———, *On directed graphs with no two disjoint directed cycles*, Combinatorica, 7 (1987), pp. 145–150.

[12] ———, *Highly connected non 2-linked digraphs*, Combinatorica, 11 (1991), pp. 393–395.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 377–379, August 1992
© 1992 Society for Industrial and Applied Mathematics
006

ANOTHER WAY OF COUNTING N^N*

ARTHUR T. BENJAMIN† AND FRITZ JUHNKE‡

Abstract. An original combinatorial proof of a combinatorial identity is presented.

Key words. combinatorial enumeration, graph theory

AMS(MOS) subject classification. 05A19

1. Introduction. We provide a combinatorial proof that

$$N^N = \sum_{0 \le k_1 \le 1} \cdots \sum_{0 \le k_1 + \cdots + k_{N-1} \le N-1} \frac{N!}{k_1! k_2! \cdots k_{N-1}!}.$$

2. Sequence graphs and cycle sets. We count the number of sequences of N integers between 1 and N, which we call N-sequences, in two ways, which give us the left- and right-hand sides of our equation, respectively. Counting in the obvious way, there are N choices for each integer, yielding N^N total N-sequences. Next, after setting up some apparatus, we answer the same question in a way that yields the summation.

As is done in [1] and [2], we define an *N-sequence graph* (*N-graph*) as a directed graph of N nodes, where each node has out-degree 1. We can set up a one-to-one correspondence between N-sequences and N-graphs as follows: Let the (directed) edge originating at node i of a N-graph terminate at node j, where j is the ith integer of the corresponding N-sequence. For example, the 12-sequence 433744657575 corresponds to the 12-graph of Fig. 1. This one-to-one correspondence allows us to count N-graphs instead of N-sequences.

To attain the right-hand side of our identity, we associate with each N-graph a unique sequence $k_1, k_2, \cdots, k_{N-1}$, which meets the conditions of the summation. We then show that the number of N-graphs associated with $k_1, k_2, \cdots, k_{N-1}$ is $N!/k_1!k_2! \cdots k_{N-1}!$.

To that end, we define the *cycle set* $\mathcal{C}$ of an N-graph to be the set of all nodes in the graph which are contained in at least one cycle. (Since our graph is finite, and each node has out-degree 1, it is clear that $\mathcal{C}$ is nonempty.) Note that, by construction, all edges originating at nodes of $\mathcal{C}$ point to nodes of $\mathcal{C}$, and no two of these edges terminate at the same node. In our example, $\mathcal{C} = \{3, 4, 6, 7\}$.

3. Ordered forests. An N-graph can be reduced to a forest by removing the edges originating at nodes contained in $\mathcal{C}$, and letting the nodes in $\mathcal{C}$ serve as roots. The forest generated by doing this to the 12-graph of Fig. 1 is shown in Fig. 2. To remove ambiguity, we then rearrange the forest into what we will call an *ordered forest* by placing the roots in ascending order from left to right, and doing the same for all *sibling sets* (i.e., children of the same node). (See Fig. 3.)

* Received by the editors September 10, 1990; accepted for publication (in revised form) July 10, 1991. This research was supported by the Western Cluster of the Pew Science Program.

† Harvey Mudd College, Claremont, California 91711.

‡ Reed College, Portland, Oregon 97202.

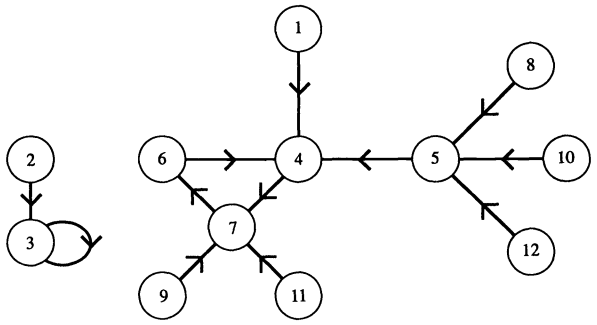


FIG. 1. *The* 12*-graph of the* 12*-sequence* 433744657575.

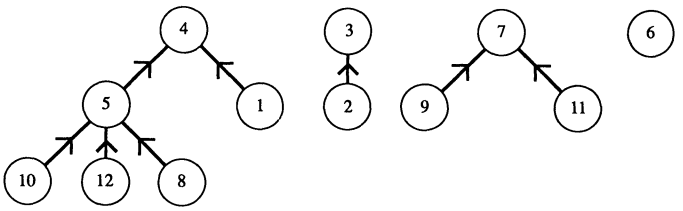


FIG. 2. *A forest from* 433744657575.

Next, we label each tree from left to right in *postorder*. Starting with the root node of a tree, the postorder labeling process may be defined recursively as follows: postorder the subtrees whose roots are the nodes that point to the current root, going from left to right among these subtrees, then label the current root. (See [3].) We let the first postorder label be zero. In Fig. 3, we have written the postorder label alongside each node. Note that, by construction, the label assigned to each node must be greater than the labels of its descendant nodes.

We can now uniquely determine the sequence $k_1, k_2, \cdots, k_{N-1}$ from our ordered forest by letting k_j be the number of children of the node with postorder label j. Note that k_0 must always be zero. In Fig. 3, k_1 through k_{11} are equal to 1, 0, 0, 0, 0, 3, 2, 0, 0, 0, 2, respectively. Note that the condition $0 \leq k_1 + k_2 + \cdots + k_j \leq j$ is met for all j between 1 and 11. This is true in general: $k_1 + k_2 + \cdots + k_j$ is equal to the number of children possessed by nodes 0 through j, all of which must have postorder labels less than j.

It remains to show that there are $N!/k_1!k_2!\cdots k_{N-1}!$ N-graphs with k-sequence $k_1, k_2, \ldots, k_{N-1}$. In [3] it is proved that every *unlabeled* rooted forest is characterized by its postorder degree sequence. Hence our k-sequence completely determines the shape of the ordered forest. For example, any unlabeled forest with k-sequence 10000320002 must look like Fig. 4.

Since the elements of $\mathcal{C}$ and the sibling sets (circled in Fig. 4) must be written in ascending order, we can convert the unlabeled forest of Fig. 4 into a labeled ordered

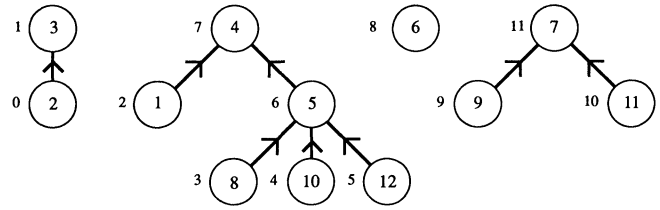


FIG. 3. *The ordered forest of* 433744657575 *with postorder labels.*

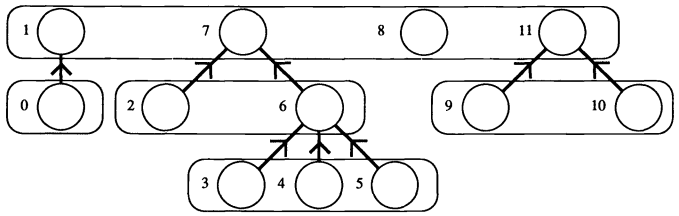


FIG. 4. *Unlabeled graph for k-sequence* 10000320002.

forest in 12!/1!3!2!2!4! ways.

In general, any unlabeled forest with k-sequence $k_1, k_2, \cdots, k_{N-1}$ can be labeled $N!/k_1!k_2!\cdots k_{N-1}!|\mathcal{C}|!$ ways. Finally, each labeled ordered forest with root set $\mathcal{C}$ gives rise to $|\mathcal{C}|!$ N-graphs because the edges leaving the nodes of $\mathcal{C}$ must enter different nodes of $\mathcal{C}$.

Therefore each k-sequence corresponds to $N!/k_1!k_2!\cdots k_{N-1}!$ N-graphs, as asserted. Since each sum counts the correct number of N-graphs, we have finished our second count and have established the identity.

Acknowledgments. The authors thank Professor Bernard Harris for bringing this problem to their attention and an anonymous referee for many valuable suggestions. Bernhard Harris attributes the question to Tsvetan Ignatov.

REFERENCES

[1] G. LABELLE, *Une nouvelle demonstration combinatoire des formules d'inversion de Lagrange*, Adv. Math., 42 (1981), pp. 217–247.

[2] D. STANTON AND D. WHITE, *Constructive Combinatorics*, Springer–Verlag Undergraduate Texts in Mathematics, Berlin, New York, 1986, pp. 105–106.

[3] D. E. KNUTH, *The Art of Computer Programming—Volume* 1: *Fundamental Algorithms*, Exercise 2.3.2.10, Addison–Wesley, Reading, MA, 1969, p. 345.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 380–385, August 1992
©1992 Society for Industrial and Applied Mathematics
007

INFINITE GRAPHS WITH NONCONSTANT DIRICHLET FINITE HARMONIC FUNCTIONS*

DONALD I. CARTWRIGHT† AND WOLFGANG WOESS‡

Abstract. A class of infinite graphs that can be embedded uniformly in the hyperbolic plane and carry nonconstant harmonic functions with finite Dirichlet sum is exhibited. In fact, a general method of constructing such harmonic functions "with prescribed boundary values" is provided.

Key words. harmonic functions on graphs, uniqueness of currents, Dirichlet sum

AMS(MOS) subject classifications. primary 31C20, 05C10; secondary 20H10, 51M20, 94C15

1. Introduction. Given a locally finite connected graph G, with vertex set V and edge set E, without loops or multiple edges, we consider it as an electrical network where each edge is a resistor of 1 ohm. An *electric current* is a flow along the edges with respect to given sources and sinks in V, which satisfies Kirchhoff's node and loop laws and has finite dissipative energy. We refer to Flanders [8] and Zemanian [16] for the fundamentals concerning electric currents in infinite networks. If input and output, given at finitely many sources and sinks, are balanced, then such an electric current always exists [8], while uniqueness holds only under additional conditions [8], [16]. In this paper, we are interested in the question of (non)uniqueness when no additional conditions are imposed.

If f is a real-valued function on V, then its *Dirichlet sum* is

$$D(f) = \sum_{[u,v]\in E} \big(f(u) - f(v)\big)^2.$$

A function $f : V \to \mathbf{R}$ is *harmonic* if $Pf = f$, where the operator P is given by

$$Pf(u) = \frac{1}{\deg(u)} \sum_{v\sim u} f(v), \qquad u \in V.$$

Here $v \sim u$ means that v is a neighbour of u, and $\deg(u)$ is the number of such v. We write $\mathbf{D}(G)$ for $\{f : V \to \mathbf{R} \mid D(f) < \infty\}$ and $\mathbf{HD}(G)$ for $\{f \in \mathbf{D}(G) \mid f \text{ harmonic}\}$. In this setting, proving the uniqueness of electric currents amounts to showing that $\mathbf{HD}(G)$ contains only the constant functions. This problem has been studied, for example, by Thomassen [13], [14], Soardi and Woess [12], and Doyle [7].

If U is a finite subset of V, then let $e(U)$ be the number of infinite components in $G \setminus U$. The *number of ends* of G is then defined as

$$e(G) = \sup\{e(U) : U \subset V \text{ finite}\}.$$

Until now, the graphs shown to carry nonconstant **HD**-functions all have more than one end [12]. Typical one-ended graphs are square grids, vertex transitive graphs with

*Received by the editors November 26, 1990; accepted for publication (in revised form) June 11, 1991. This research was partially supported by an Australian Research Council grant.

†School of Mathematics and Statistics, University of Sydney, N.S.W. 2006, Australia.

‡Dipartimento di Matematica, Università di Milano, Via C. Saldini, 50, Milano 20133, Italy.

nonlinear polynomial growth, or Cartesian products of two or more infinite graphs. All these have no nonconstant **HD**-functions [12], [14]. We exhibit a class of graphs, each of which carries nonconstant **HD**-functions. The class contains many vertex-transitive graphs with one end. Our graphs are embedded in a uniform way in the open unit disc with the hyperbolic metric, they have at least two accumulation points on the unit circle, and they satisfy a strong isoperimetric inequality. The most typical examples are the dual graphs of tesselations of the disc by cocompact Fuchsian groups. For these graphs, $\mathbf{HD}(G)$ is a discrete anologue of the space of (classical) harmonic functions u on the unit disc for which

$$\iint_{x^2+y^2<1} \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 dxdy < \infty.$$

Such spaces have also been studied more generally in the context of potential theory on Riemann surfaces (see, e.g., Ahlfors and Sario [1]).

2. Results. Let Δ denote the unit disc $\{z \in \mathbf{C} : |z| < 1\}$ equipped with the hyperbolic metric ρ. All properties and formulas from hyperbolic geometry needed below can be found in Beardon [3, §7.1, especially pp. 131–132]. A graph G with vertex set V and edge set E is called *uniformly embedded* in Δ if

(i) $V \subset \Delta$, and

(ii) There is a constant $c \geq 1$ such that, for all $u, v \in V$, $c^{-1}\rho(u,v) \leq d(u,v) \leq c\,\rho(u,v)$.

Here $d(u,v)$ is the discrete distance between vertices u and v, i.e., the smallest number of edges on a path in G connecting the two.

The *limit set* of G is the set ∂V of accumulation points of V on the unit circle $\partial\Delta$ (in the Euclidean topology). Note that $\partial V \neq \emptyset$ if G is infinite. By *continuity* of a function on $\bar{V} = V \cup \partial V$, we mean continuity with respect to the relative (Euclidean) topology.

For a finite subset U of V, we denote by dU the set of vertices in U that have a neighbour in $V \setminus U$. We say that G satisfies a *strong isoperimetric inequality* (see, e.g., Dodziuk [6], Gerl [9]) if there is a number $\alpha > 0$ such that $|dU| \geq \alpha|U|$ for every finite $U \subset V$. Such graphs are sometimes called *coercive* (see Ancona [2]), *infinite expanders*, or *enlargers* (see Bien [4]).

THEOREM. *Suppose that G is uniformly embedded in Δ and satisfies a strong isoperimetric inequality. Then, for every $\varphi : \bar{\Delta} \to \mathbf{R}$ satisfying a Lipschitz property $|\varphi(z) - \varphi(w)| \leq M|z - w|$, there is a unique continuous function h on $\bar{V}$ such that*

(a) *h coincides with φ on ∂V, and*

(b) *$h_{|V} \in \mathbf{HD}(G)$.*

In particular, if ∂V has more than one point, then $\mathbf{HD}(G)$ contains nonconstant functions.

We now give examples of graphs satisfying the hypotheses of the theorem.

Example 1 (Fuchsian groups (see, e.g., [3, §§8–9] and also [2, p. 13])). Let Γ be a finitely generated Fuchsian group acting on Δ, and let $V = \{\gamma 0 : \gamma \in \Gamma\}$, where it is assumed that $\gamma 0 \neq 0$ for all $\gamma \in \Gamma$, $\gamma \neq id$. Assume that the limit set ∂V of Γ has more than two (and thus infinitely many) points (i.e., Γ is nonelementary) and that there are no parabolic elements in Γ. Then there is a convex fundamental domain F of Γ in Δ which has finitely many sides and no proper vertices [3, pp. 227, 254]. For $\beta, \gamma \in \Gamma$, we join the vertices $u = \beta 0$ and $v = \gamma 0$ by an edge if and only if βF and γF have a common side. We thus obtain a graph G that is just the dual graph of the tesselation of Δ given by the Γ images of F. It is a Cayley graph of Γ [3,

p. 220]. It is known [2, p. 14] that G is uniformly embedded in Δ. Furthermore, any nonelementary Fuchsian group is *nonamenable* (see [11]), which is equivalent to the property that some (equivalently, all) of its Cayley graphs with respect to finitely many generators satisfy a strong isoperimetric inequality (see [9] and [2]). Hence G carries nonconstant **HD**-functions.

Because there are no proper vertices, it is easy to see that G has one end if and only if F is bounded in the hyperbolic metric, which happens exactly when the limit set ∂V is all of $\partial\Delta$. Otherwise, G has infinitely many ends. (Indeed, we can show that Γ has a free subgroup of finite index in the latter case.) For a detailed description of the metric structure of Cayley graphs of one-ended (equivalently, cocompact) Fuchsian groups, see Cannon [5].

A typical example of this type is abstractly given by the presentation

$$\Gamma = \langle \alpha, \beta, \gamma, \delta \mid \alpha^{-1}\beta^{-1}\alpha\beta\gamma^{-1}\delta^{-1}\gamma\delta = id \rangle.$$

With respect to these generators, the Cayley graph of Γ is made rather like a square grid, but instead of four squares, eight octagons meet at each vertex. Further examples are the dual graphs of several of the tesselations of Δ shown in Magnus [10].

Finally, we remark that as an abstract group, every finitely generated nonelementary Fuchsian group is isomorphic with a group having the above properties (see, e.g., [11, p. 226]). Furthermore, any two Cayley graphs of a given group with respect to finite sets of generators are metrically equivalent. Hence every Cayley graph of a finitely generated nonelementary Fuchsian group carries nonconstant **HD**-functions.

Example 2. Suppose that G is a graph, with vertex set V and edge set E, which is uniformly embedded in Δ and, in addition, satisfies

(iii) $\rho(z, V) \leq k < \infty$ for all $z \in \Delta$.

Then G has one end, $\partial V = \partial\Delta$, and it is shown in [2] that G satisfies a strong isoperimetric inequality. Thus our theorem applies.

A typical example is the following graph, which we describe in the hyperbolic upper half-plane $\{z \in \mathbf{C} : \Im(z) > 0\}$ rather than in the disc. Let $V = \{m2^{n-1} + i2^n : m, n \in \mathbf{Z}\}$; the edges are between $m2^{n-1} + i2^n$ and $(m+1)2^{n-1} + i2^n$ and between $m2^{n-1} + i2^n$ and $m2^{n-1} + i2^{n-1}$ for each $m, n \in \mathbf{Z}$ (see Fig. 1). A similar graph is described in [2, p. 13]. (Read $\log((3+\sqrt{5})/2)$ for "Log(2)" there.)

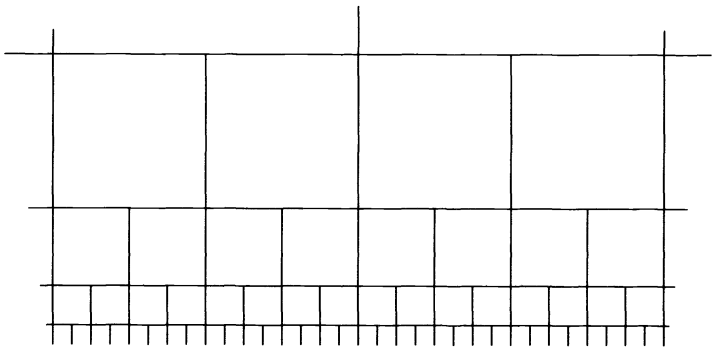


FIG. 1

3. Proofs. Recall that, for $z, w \in \Delta$, the hyperbolic metric ρ satisfies

$$\tanh(\rho(z,w)/2) = |z - w|/|1 - z\bar{w}|. \tag{1}$$

The open hyperbolic disc with centre $u \in \Delta$ and radius $r > 0$ is denoted by $B(u,r)$. Its hyperbolic area is independent of u, and, writing $z = x + iy$, is given by

$$\sigma(r) = \iint_{B(u,r)} \frac{4}{(1-|z|^2)^2}\, dxdy = 4\pi \sinh^2(r/2). \tag{2}$$

We always assume that G is uniformly embedded in Δ, with constant $c > 0$ as in (i).

LEMMA 1. *For $u \in V$, $\deg(u) \le \sigma(c + \frac{1}{2c})/\sigma(\frac{1}{2c})$.*

Proof. If $u, v \in V$, $u \ne v$, then $d(u,v) \ge 1$ and so $\rho(u,v) \ge 1/c$. Thus the discs $B(v, \frac{1}{2c})$, where $v \in V$ and $v \sim u$, are mutually disjoint and contained in $B(u, c + \frac{1}{2c})$.

A similar argument shows that every compact subset of Δ must have finite intersection with V, so that indeed $\partial V \ne \emptyset$ if V is infinite. In the next lemma, $\lambda(B)$ denotes the usual Euclidean area of $B \subset \Delta$.

LEMMA 2. *For $u \in \Delta$, $\lambda\big(B(u,r)\big) \ge \big(\sigma(r)/4e^{2r}\big)\big(1-|u|^2\big)^2$.*

Proof. The hyperbolic disc $B(u,r)$ is a Euclidean disc with centre on the ray from 0 through u (e.g., calculate directly from (1)). Thus the minimum of $1 - |z|$ for z in the closure of $B(u,r)$ is attained at the point z_1 on this ray for which $|z_1| \ge |u|$ and $\rho(u, z_1) = r$. From (1) we have $\tanh(r/2) = (|z_1| - |u|)/(1 - |z_1||u|)$, whence

$$1 - |z_1| \ge (1 - |u|)e^{-r}. \tag{3}$$

Thus, by (2) and (3),

$$\begin{aligned}\sigma(r) &\le \lambda\big(B(u,r)\big)\frac{4}{(1-|z_1|^2)^2} \\ &= \lambda\big(B(u,r)\big)\frac{4}{(1+|z_1|)^2(1-|z_1|)^2} \le \lambda\big(B(u,r)\big)\frac{4e^{2r}}{(1-|u|^2)^2}.\end{aligned}$$

LEMMA 3. *It holds that $\sum_{u\in V}(1-|u|^2)^2 \le 4\pi e^{1/c}/\sigma(\frac{1}{2c})$.*

Proof. Once more using the disjointness of the balls $B(u, \frac{1}{2c})$, $u \in V$, we obtain that

$$\pi = \lambda(\Delta) \ge \sum_{u\in V} \lambda\big(B(u, 1/2c)\big),$$

and Lemma 2 yields the result.

PROPOSITION 1. *If $\varphi : \Delta \to \mathbf{R}$ satisfies a Lipschitz property $|\varphi(z) - \varphi(w)| \le M|z-w|$ on Δ, then $f = \varphi_{|V} \in \mathbf{D}(G)$.*

Proof. The hypothesis on φ implies that

$$D(f) \le (M^2/2) \sum_{u\in V} \sum_{v\sim u} |u-v|^2.$$

If $u \sim v$, then $\rho(u,v) \le c$, and [3, (7.2.4)] gives $|u-v|^2 \le \sinh^2(c/2)(1-|u|^2)(1-|v|^2)$. Hence the Cauchy–Schwarz inequality implies that

$$\begin{aligned}\sum_{u\in V}\sum_{v\sim u}|u-v|^2 &\le \sinh^2(c/2)\Big(\sum_{u\in V}\sum_{v\sim u}(1-|u|^2)^2\Big)^{1/2}\Big(\sum_{u\in V}\sum_{v\sim u}(1-|v|^2)^2\Big)^{1/2} \\ &= \sinh^2(c/2)\sum_{u\in V}\deg(u)(1-|u|^2)^2,\end{aligned}$$

which is finite by Lemmas 1 and 3.

We now use the strong isoperimetric inequality. Let $l_0(V)$ denote the linear space of finitely supported functions $f : V \to \mathbf{R}$. Furthermore, let $l^2(V)$ be the Hilbert space of square summable functions $f : V \to \mathbf{R}$, with the usual inner product and corresponding norm $\|f\|_2 = \left(\sum_{u \in V} f(u)^2\right)^{1/2}$. Observe that, for $f \in l^2(V)$,

$$D(f) \leq \sum_{u \in V} \sum_{v \sim u} \bigl(f(u)^2 + f(v)^2\bigr) \leq 2K\|f\|_2^2, \tag{4}$$

if $\deg(u) \leq K$ on V, so that $f \in \mathbf{D}(G)$.

PROPOSITION 2. *Assume that G satisfies a strong isoperimetric inequality and has bounded vertex degrees. Then every $f \in \mathbf{D}(G)$ has a unique decomposition $f = g + h$, where $g \in l^2(V)$ and $h \in \mathbf{HD}(G)$.*

Proof. Under our assumptions, it is known from [9] or [2] that there is a constant $C > 0$ such that

$$D(f) \geq C\|f\|_2^2 \quad \text{for every } f \in l_0(V). \tag{5}$$

Consider the quotient space of $\mathbf{D}(G)$ with respect to the subspace of constant functions, and let $[\cdot]$ denote the corresponding quotient map. Then $[\mathbf{D}(G)]$ becomes a Hilbert space with inner product

$$\langle [f], [g] \rangle = \sum_{[u,v] \in E} \bigl(f(u) - f(v)\bigr)\bigl(g(u) - g(v)\bigr).$$

It is easy to check that $h \in \mathbf{D}(G)$ is harmonic if and only if

$$\langle [g], [h] \rangle = 0 \quad \text{for every } g \in l_0(V).$$

Hence $[\mathbf{HD}(G)]$ is the orthogonal complement of $[l_0(V)]$, and thus closed, while its orthogonal complement is the closure of $[l_0(V)]$. Let (f_n) be a sequence in $l_0(V)$ such that $[f_n]$ converges in $[\mathbf{D}(G)]$. Then, by (5),

$$\langle [f_n - f_m], [f_n - f_m] \rangle \geq C\|f_n - f_m\|_2^2,$$

and so (f_n) is a Cauchy sequence in $l^2(V)$. If $f \in l^2(V)$ is its limit, then by (4), $f \in \mathbf{D}(G)$ and $[f]$ is the limit of $[f_n]$ in $[\mathbf{D}(G)]$. Thus the closure of $[l_0(V)]$ is $[l^2(V)]$. Observing that nonzero constant functions are in $\mathbf{HD}(G)$, but not in $l^2(V)$, the orthogonal decomposition theorem now yields the result.

We remark that Proposition 2 is a discrete analogue of Royden's decomposition. Another version of it is stated by Yamasaki [15]. We have now collected all the ingredients necessary to prove our main result.

Proof of the theorem. Let $\varphi : \bar{\Delta} \to \mathbf{R}$ satisfy $|\varphi(z) - \varphi(w)| \leq M|z - w|$. Then $f = \varphi_{|V} \in \mathbf{D}(G)$ by Proposition 1. By Proposition 2, there is a unique $h \in \mathbf{HD}(G)$ such that $h - f \in l^2(V)$. In particular, $h - f$ vanishes uniformly at infinity, so that h extends continuously to $\bar{V}$ with the same boundary values on ∂V as f. This proves the existence of the required h. If h' is another harmonic function with the same boundary values, then $h - h'$ is harmonic and vanishes at infinity. The maximum principle implies that $h - h' = 0$.

Finally, if ∂V has more than one point, then there are functions on $\bar{\Delta}$ satisfying a Lipschitz condition that are nonconstant on ∂V. The corresponding harmonic

functions have nonconstant values on ∂V and hence, by continuity, are nonconstant on V.

REFERENCES

[1] L.V. AHLFORS AND L. SARIO, *Riemann Surfaces*, Princeton University Press, Princeton, NJ, 1960.

[2] A. ANCONA, *Positive harmonic functions and hyperbolicity*, in Potential Theory, Surveys and Problems, J. Král et al., eds., Springer Lecture Notes in Mathematics, Berlin, New York, 1344 (1988), pp. 1–23.

[3] A.F. BEARDON, *The Geometry of Discrete Groups*, Springer-Verlag, New York, Heidelberg, Berlin, 1983.

[4] F. BIEN, *Constructions of telephone networks by group representations*, Notices Amer. Math. Soc., 36 (1989), pp. 5–22.

[5] J.W. CANNON, *The combinatorial structure of cocompact discrete hyperbolic groups*, Geom. Dedicata, 16 (1984), pp. 123–148.

[6] J. DODZIUK, *Difference equations, isoperimetric inequality and transience of certain random walks*, Trans. Amer. Math. Soc., 294 (1984), pp. 787–792.

[7] P.G. DOYLE, *Electric currents in infinite networks*, preprint, Princeton University, Princeton, NJ, 1988.

[8] H. FLANDERS, *Infinite networks* I—*Resistive networks*, IEEE Trans. Circuit Theory, 18 (1971), pp. 326–331.

[9] P. GERL, *Random walks on graphs with a strong isoperimetric inequality*, J. Theoret. Probab., 1 (1988), pp. 171–187.

[10] J. M. MAGNUS, *Non-Euclidean Tesselations and Their Groups*, Academic Press, New York, 1974.

[11] C. SERIES, *Martin boundaries of random walks on Fuchsian groups*, Israel J. Math., 44 (1983), pp. 221–242.

[12] P. M. SOARDI AND W. WOESS, *Uniqueness of currents in infinite resistive networks*, Discrete Appl. Math., 31 (1991), pp. 37–49.

[13] C. THOMASSEN, *Resistances and currents in infinite electrical networks*, J. Combin. Theory Ser. B, 49 (1990), pp. 87–102.

[14] ———, *Transient random walks, harmonic functions, and electric currents in infinite resistive networks*, preprint, Technical University of Denmark, DK-2800 Lyngby, Denmark, 1989.

[15] M. YAMASAKI, *Discrete potentials on an infinite network*, Mem. Fac. Sci. Shimane Univ., 13 (1979), pp. 31-44.

[16] A. H. ZEMANIAN, *Infinite electrical networks*, Proc. IEEE, 64 (1976), pp. 6-17.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 386–397, August 1992
© 1992 Society for Industrial and Applied Mathematics
008

MEAN PASSAGE TIMES AND NEARLY UNCOUPLED MARKOV CHAINS*

REFAEL HASSIN† AND MOSHE HAVIV‡

Abstract. Let $P(0) \in R^{n\times n}$ be a stochastic matrix representing transition probabilities in a Markov chain. Also, for a matrix $A \in R^{n\times n}$ whose row-sums are zero, let $P(\varepsilon) \equiv P(0) + \varepsilon A$ be stochastic and irreducible for all $0 < \varepsilon \leq \varepsilon_{\max}$, for some $\varepsilon_{\max}$. Finally, let $M(\varepsilon)$ be a matrix whose (i,j) entry is the mean passage time from state i to state j when transitions are governed by $P(\varepsilon)$. When the Markov chain associated with $P(0)$ is decomposable into a number of independent chains plus a set of transient states, some of the entries of $M(\varepsilon)$ have singularities at zero. The orders of these poles define timescales associated with the process when ε is small. An algorithm is developed for computing these orders. The only input required is the supports of $P(0)$ and A, making the problem a combinatorial one. Finally, it is shown how the orders of the poles of $M(\varepsilon)$ at zero play a role in developing series expansions for $\pi(\varepsilon)$, the stationary distribution of $P(\varepsilon)$.

Key words. Markov chains, mean passage time, nearly uncoupled

AMS(MOS) subject classifications. 60J10, 68C05

1. Introduction.

Let $P(0) \in R^{n\times n}$ be a stochastic matrix representing transition probabilities in a Markov chain. Let $A \in R^{n\times n}$ have zero row-sums. For all real ε, $0 < \varepsilon \leq \varepsilon_{\max}$, assume that $P(\varepsilon) \equiv P(0) + \varepsilon A$ are stochastic matrices representing transition probabilities in irreducible Markov chains. (The irreducibility assumption of $P(\varepsilon)$ is without loss of generality. However, we still require that the structure of the chains is the same for all ε, $0 < \varepsilon \leq \varepsilon_{\max}$. Of course, such an $\varepsilon_{\max}$ exists.) Note that we do not assume irreducibility of $P(0)$, and, in fact, we are interested in the case where the Markov chain associated with it is decomposable into a number of independent recurrent classes plus a set of transient states. In that case and for small values of ε, the matrices $P(\varepsilon)$ and the associated Markov chains are called *nearly uncoupled* or *nearly completely decomposable.*

For $0 < \varepsilon \leq \varepsilon_{\max}$, let $M(\varepsilon) \in R^{n\times n}$ be such that $M_{ij}(\varepsilon)$ is the mean passage time from state i to state j when transitions are governed by $P(\varepsilon)$. It is clear that, if $P(0)$ is decomposable, then there exist pairs (i,j) such that $\lim_{\varepsilon\to 0} M_{ij}(\varepsilon) = \infty$. As $M(\varepsilon)$ admits a Laurent series expansion (see §7), this implies that some entries of $M(\varepsilon)$ have singularities at zero; namely, they have poles there. The orders of these poles represent various timescales in the nearly uncoupled Markov chain. For example, if the order of the pole of $M_{ij}(\varepsilon)$ at zero is two, then, for a process that initiates at i, the event of hitting state j for the first time occurs after an expected time, which is of the same order of magnitude as $1/\varepsilon^2$.

The main purpose of this paper is to show that the problem of finding the orders of the above-mentioned poles is combinatorial and to develop an algorithm for computing them. Specifically, we suggest an algorithm whose input is the two binary matrices describing the supports of $P(0)$ and A. The output of the algorithm is the orders of the poles.

Finally, we discuss an application of the orders of the poles. Let $\pi(\varepsilon)$ be the

* Received by the editors September 6, 1990; accepted for publication (in revised form) July 25, 1991.

† Department of Statistics, Tel Aviv University, Tel Aviv 69978, Israel.

‡ Department of Statistics, Hebrew University of Jerusalem. Present address, Department of Econometrics, University of Sydney, N.S.W. 2006, Australia.

stationary distribution of $P(\varepsilon)$. If $\pi(\varepsilon) = \sum_{i=0}^{\infty} \pi^{(i)}\varepsilon^i$, then the series $\{\pi^{(i)}\}_{i=0}^{\infty}$ solves

$$\pi^{(0)}(I - P(0)) = 0,$$

and for $j = 0, 1, \ldots,$

$$\pi^{(j+1)}(I - P(0)) = \pi^{(j)}A.$$

If $P(0)$ is decomposable, then the first system of above equations does not determine $\pi^{(0)}$ uniquely. We show that the minimal number of consecutive sets of equations that must be considered to determine $\pi^{(0)}$ uniquely is $\max_{i,j}(u_{ij} - u_{jj}) + 1$, where u_{ij} is the order of the pole of $M_{ij}(\varepsilon)$ at zero.

Nearly uncoupled systems were introduced by Simon and Ando [18]. These were applied to some Markovian models by Courtois [3]. Also, they were considered in the Russian literature, probably begining with Gaitsgory and Pervozvansky [6]. The question of first passage times is dealt with in [10] and [11], but only for the case where $P(0)$ does not possess transient states. For this case, Latouch and Louchard (see [10] and [11]) show that the orders of the poles of $M(\varepsilon)$ at zero are zero or one. The more general case, where two independent recurrent chains at $P(0)$ are coupled at $P(\varepsilon)$ through states that are transient at $P(0)$, is considered in Delebecque [4], Coderch et al. [1], [2], and Rohlicek and Willsky [14], who developed algorithms for approximating the transient and the long-run behavior of the Markov chains associated with $P(\varepsilon)$ for $0 < \varepsilon \leq \varepsilon_{\max}$. The analysis in the above-mentioned papers is based on Kato's [8] classical perturbation results. The combinatorial version of the algorithm in [14] is given in Rohlicek and Willsky [13]. They solved a problem different from that addressed here. Specifically, let $F_{ij}^{T}(\varepsilon)$ be the probability that a process governed by $P(\varepsilon)$ and initiating at state i hits state j for the first time prior to time T.[1] Then let

$$d_{ij} = \arg\max\{d \mid \lim_{\varepsilon \to 0} F_{ij}^{T/\varepsilon^d}(\varepsilon) > 0 \text{ for some finite } T\}.$$

Next, we present an example showing that u_{ij} and d_{ij} do not necessarily coincide. This example was communicated to us by a referee.

Let

$$P(\varepsilon) = \begin{pmatrix} 0 & \varepsilon & 1-\varepsilon \\ 0 & 1-\varepsilon^2 & \varepsilon^2 \\ 0 & 0 & 1 \end{pmatrix}.$$

Here $u_{13} = 1$: although $P_{13}(\varepsilon) = \Theta(1)$, the transition probability ε from state 1 into state 2, and then the additional expected time of $1/\varepsilon^2$ until hitting state 3, leads to this value of u_{13}. On the other hand, $d_{13} = 0$ as for any $T \geq 1$, $F_{13}^{T}(\varepsilon) \geq 1 - \varepsilon$.[2]

The question of series expansions for nearly uncoupled Markov chains is analyzed by Schweitzer [15]–[17] and Haviv and Ritov [7]. Schweitzer showed that the deviation matrix of $P(\varepsilon)$ has a Laurent series expansion around zero, and he gave some implicit forms for it. He also solved for the series expansion of $\pi(\varepsilon)$ and related it to the

[1] In order to ease the exposition, here we allow T to be any nonnegative number and not necessarily an integer.

[2] Note that $P(\varepsilon)$ here was not defined by a linear perturbation. However, the same phenomenon can be seen with a corresponding linear perturbation, but in a Markov chain, where state 2, above, is replaced with three other states.

expansion for the deviation matrix. His method, or, alternatively, the method given in [7], can be used to obtain the order of the poles as a by-product. Since these methods are based on solving n^2, equations they have a larger complexity $O(n^6)$, and they do not explore the combinatorial aspects of the problem. Finally, Langenhop [9] suggested a general method for inverting near singular matrices. As $M(\varepsilon)$ uniquely solves a system of equations, this method can be utilized for the problem discussed here, and then the orders of the poles are obtained as a by-product. However, for computing the orders of the poles, our direct approach is much simpler.

After introducing some terminology in §2, we present the algorithm in §3. Section 4 is devoted to proving the correctness of the algorithm. In §5 we mention a possible generalization of our algorithm to nonlinear perturbations. In §6 we state some background on Markov chains, and we use it in §7 for developing a series expansion for $\pi(\varepsilon)$ around $\varepsilon = 0$. Section 8 concludes the paper with a numerical example.

2. Terminology. A function $f : R_+ \to R_+$ is called $\Theta(\varepsilon^k)$ for some integer k (positive, zero, or negative) if there exist two positive real numbers M_1 and M_2 such that, for all $\varepsilon > 0$ small enough,

$$M_1\varepsilon^k \leq f(\varepsilon) \leq M_2\varepsilon^k.$$

If $f(\varepsilon) = \Theta(\varepsilon^{-k})$, then we say that $f(\varepsilon)$ is of *order of magnitude* k.

The following are immediate:

$$\Theta(\varepsilon^{k_1}) + \Theta(\varepsilon^{k_2}) = \Theta(\varepsilon^{min\{k_1,k_2\}}), \qquad \Theta(\varepsilon^{k_1}) * \Theta(\varepsilon^{k_2}) = \Theta(\varepsilon^{k_1+k_2}).$$

Consider $G = (V, E_e, E_r)$, a finite directed graph with a vertex set V, an edge set $E = E_e \cup E_r$ that may contain loops, and a distinguished vertex $s \in V$. G is *strongly connected*; that is, it contains, for every $i, j \in V$ a directed $i - j$ path. We call edges of E_e "*e-edges*" (epsilon edges) and those of E_r "*r-edges*" (regular edges). These sets are disjoint. A path in G is called an "*r-path*" if it consists of r-edges only (similarly for an "*r-cycle*"). An "*r-component*" is defined to be a maximal strongly connected subgraph of (V, E_r). Note that an r-component may also be a single vertex. For a subset $C \subset V$, we let $\delta(C) = \{(i,j) \in E | i \in C, j \notin C\}$ be the set of its (outward-oriented) *boundary edges*.

In terms of $P(\varepsilon)$, which was defined in the Introduction, we define the following graph G. Each vertex of V corresponds to a state. Each edge $(i,j) \in E$ is associated with a *transition probability* $p_{ij}(\varepsilon)$. If $p_{ij}(\varepsilon) = \Theta(1)$, then $(i,j) \in E_r$, while, if $p_{ij}(\varepsilon) = \Theta(\varepsilon)$, then $(i,j) \in E_e$. If $P_{ij}(\varepsilon) = 0$, then $(i,j) \notin E$. Obviously, every $i \in V$ satisfies $\sum_{j|(i,j)\in E} p_{ij}(\varepsilon) = 1$.

Fix a state $s \in S$. Let $m_i(\varepsilon)$ denote the expected time until s is first reached when the initial state is i ($m_s(\varepsilon)$ is the expected return time to s, given that it is also the initial state). It is shown that $m_i(\varepsilon) = \Theta(\varepsilon^{-u(i)})$ for some integer $u(i)$ that is zero or positive. Our problem is to compute $u(i)$ for all $i \in V$.

3. The algorithm. The algorithm computes $u(i)$ for $i \neq s$ by first assigning and revising temporary values and, finally, changing them to permanent. The set of vertices with temporary values is denoted by T. In a final step, $u(s)$ is computed from the other values. In the course of the algorithm, a multigraph $G' = (V', E'_r, E'_e)$ is maintained. Then sets $C \subset V'$ are *condensed* into a single vertex c as follows: Edges $(i,j) \in \delta(C)$ are replaced by (possibly parallel) edges (c,j), and, similarly, edges in $\delta(V \setminus C)$ are replaced by (possibly parallel) edges (i,c). Edges (i,j) such that $i, j \in C$ are simply deleted. The correspondence of each edge in G' to the original edge in G is

maintained. The algorithm is stated next. A numerical example is given in §8. The reader may wish to consider this example prior to reading the rest of the paper.

Input: $G = (V, E_e, E_r)$.

Output: $u(i)$, $i \in V$.

Step 1 (Initialization).

Construct a graph $G' = (V', E'_e, E'_r)$ from G by first setting $G' \leftarrow G$, and then deleting all loops $(i,i) \in E_e$ and all edges going out of s. Set $u(i) \leftarrow 0$ and $S(i) \leftarrow \{i\}$ for all $i \in V$.

Step 2 (Elimination of loops).[3]

If G' contains no loops, go to Step 3. Otherwise, let $(i,i) \in E'_r$. Set $E'_r \leftarrow E'_r \setminus \{(i,i)\}$. If $\delta(\{i\}) \cap E'_r = \phi$, set $E'_r \leftarrow E'_r \cup \delta(\{i\})$, $E'_e \leftarrow E'_e \setminus \delta(\{i\})$ and $u(i) \leftarrow 1$. Repeat Step 2.

Step 3 (Condensation of cycles).

If G' contains no directed r-cycles, go to Step 4.

Let C be (the vertex set in V' of) such a cycle.[4] Condense C into a single vertex c.

Case (i). $\delta(C) \cap E'_r \neq \phi$.

Set $u(c) \leftarrow \max\{u(i) | i \in C\}$.

Case (ii). $\delta(C) \subseteq E'_e$.

Set $u(c) \leftarrow 1 + \max\{u(i) | i \in C\}$.

Set $E'_r \leftarrow E'_r \cup \delta(C)$, $E'_e \leftarrow E'_e \setminus \delta(C)$.

Set $S(c) \leftarrow \cup_{i \in C} S(i)$.

Repeat Step 3.

Step 4 (Solution of the problem for r-acyclic graphs).

Set $T \leftarrow V'$. Let $u(j) = \max\{u(i) | i \in T\}$. (Break ties arbitrarily.) Delete j from T (thus turning $u(j)$ into permanent for j). For r-edges (i,j) where $i \in T$, set $u(i) \leftarrow u(j)$. For e-edges (i,j) where $i \in T$, set $u(i) \leftarrow \max\{u(i), u(j) - 1\}$. If $T = \phi$, go to Step 5. Else, repeat Step 4.

Step 5 (Computation of $u(i)$ $i \in V \setminus \{s\}$).

$\{S(v') | v' \in V'\}$ is a partition of V. For each $v \in V$ find $v' \in V'$ such that $v \in S(v')$ and set $u(v) \leftarrow u(v')$.

Step 6 (Computation of $u(s)$).

Set $u(s) \leftarrow \max\{\max\{u(i) | (s,i) \in E_r\}, \max\{u(i) - 1 | (s,i) \in E_e\}\}$.

4. Validation of the algorithm. In this section, we use the following notation; we distinguish between *states*, which correspond to the initial Markov process, and *vertices* $v \in V'$, which correspond to the sets of states $S(v)$; we also distinguish between *transitions*, which correspond to the initial Markov process, and *edges* $(i,j) \in E'_r \cup E'_e$, which correspond to moves of the process between vertices. It should be emphasized that each edge $(i,j) \in E'$ is associated with a transition from a state in $S(i)$ to a state in $S(j)$. It is possible, however, that an r-edge in V' is associated with a transition of a $\Theta(\varepsilon)$ probability, since such changes are performed at Step 3, Case (ii) in the above algorithm.

Let C denote an r-cycle (or an r-component) of G'; We say for a state j that $j \in C$ whenever $j \in S(v)$ for some $v \in C$. Also, let $P^*_i(\varepsilon) = \sum_{j|(i,j) \in \delta(C)} p_{ij}(\varepsilon)$ be the probability of an immediate exit from C, given that the current state i is in C. For

[3] This step can be deleted; instead, loops will be considered as directed cycles in Step 3. It is included here only to simplify the proofs below and, in particular, to guarantee that the inductive assumption in Theorem 4.5 (i), below, holds.

[4] Instead of r-cycles, we may choose r-components.

a vertex $v \in V'$ and for states i and j in $S(v)$, let $Q_{ij}(\varepsilon)$ be the expected number of visits in state $j \in S(v)$ before first exiting $S(v)$, given the current state is $i \in S(v)$. A similar notation is used for cycles instead of vertices. Note that, for simplicity, we omit from the notation the cycle or the vertex. However, from the context, it is clear which is the cycle or the vertex under consideration.

The following two lemmas give the orders of magnitude of the expected number of visits in a state prior to an exit from a cycle. Their proofs are similar, and thus we supply a detailed proof only to one of them.

LEMMA 4.1. *Let C be an r-cycle in the original graph G. Suppose that $\delta(C) \cap E_r \neq \phi$. Then $Q_{ij}(\varepsilon) = \Theta(1)$ for all $i, j \in C$.*

LEMMA 4.2. *Let C be an r-cycle in the original graph G. Suppose that $\delta(C) \subseteq E_e$. Then $Q_{ij}(\varepsilon) = \Theta(\varepsilon^{-1})$ for all $i, j \in C$.*

Proof. Let $T(\varepsilon)$ be the submatrix of $P(\varepsilon)$ representing transition probabilities in an r-cycle C. It is well known that $Q(\varepsilon) = (I - T(\varepsilon))^{-1}$. Hence, $Q_{ij}(\varepsilon)$ is a ratio between two polynomials in ε and therefore it has an integer order of magnitude. Since for all $i, j \in C$ there exists an $i-j$ r-path, the orders of magnitude of all the entries of $Q(\varepsilon)$ are identical. Consequently, it is sufficient to prove that $\sum_{j \in C} Q_{ij}(\varepsilon) = \Theta(\varepsilon^{-1})$. Indeed, since the exit probabilities from all states are at most $\Theta(\varepsilon)$, the time to the first exit from C (regardless of the current state) is stochastically dominated by a geometric random variable whose expectation is $\Theta(\varepsilon^{-1})$. Hence, $\sum_{j \in C} Q_{ij}(\varepsilon)$ is at least $\Theta(\varepsilon^{-1})$. On the other hand, in order for $\sum_{j \in C} Q_{ij}(\varepsilon)$ to be $\Theta(\varepsilon^{-k})$ for $k \geq 2$ there should be at least one $j \in C$ with this property. An r-path from j to an exit state $e \in C$ exists, however, and hence the number of visits to state e is also of the order of magnitude k. This contradicts the fact that the number of visits to state e is at most $1/P_e^*(\varepsilon) = \Theta(\varepsilon^{-1})$. □

The following two lemmas correspond to exit probabilities from a cycle C in the original graph G. Basically, they show that the order of exit probabilities from various states $i \in C$ and via various pairs $(i, j) \in \delta(C)$ does not depend on the entering state.

LEMMA 4.3. *For an r-cycle C in the original graph G, suppose that $\delta(C) \cap E_r \neq \phi$. Then, for every current state in C, an exit from C occurs with probability $\Theta(1)$ for every $(i, j) \in \delta(C) \cap E_r$ and with probability $\Theta(\varepsilon)$ for every $(i, j) \in \delta(C) \cap E_e$.*

Proof. For a current state i, with $(i, j) \in \delta(C) \cap E_r$, the exit probability is, of course, $\Theta(1)$. For all other states k in C, there exists an r-path from k to i, making the probability of an exit through (i, j) also $\Theta(1)$ for any current state k. For $(i, j) \in \delta(C) \cap E_e$, pick an arbitrary $(g, h) \in \delta(C) \cap E_r$. As there is an r-path from g to i, it is easy to see that, conditional on an exit from (i, j) or (g, h), the exit is from (i, j) with probability $\Theta(\varepsilon)$. Extending that to conditioning on an exit completes the proof. □

LEMMA 4.4. *For a cycle C in the original graph G, suppose that $\delta(C) \subseteq E_e$. Let $(i, j) \in \delta(C)$. Then, regardless of the current state in C, exit occurs via (i, j) with probability $\Theta(1)$.*

Proof. We prove that i is the exit state with probability $\Theta(1)$. The claim then follows immediately by considering conditional probabilities. For $i, j \in C$, let $G_{ij}(\varepsilon)$ be the probability of exiting through j, given that i is the entering (or current) state. If $i \neq j$, then

$$G_{ij}(\varepsilon) = \sum_{k \in C} T_{ik}(\varepsilon) G_{kj}(\varepsilon),$$

and otherwise

$$G_{ii}(\varepsilon) = \sum_{k \in C} T_{ik}(\varepsilon) G_{ki}(\varepsilon) + P_i^\star(\varepsilon).$$

Hence $G(\varepsilon) = Q(\varepsilon)\mathrm{diag}(P^\star(\varepsilon))$ where $\mathrm{diag}(P^\star(\varepsilon))$ is a diagonal matrix whose ith diagonal entry equals $P_i^\star(\varepsilon)$. Solving for $G(\varepsilon)$, we obtain that $G_{ij}(\varepsilon) = Q_{ij}(\varepsilon)P_j^\star(\varepsilon)$, which is $\Theta(\varepsilon^{-1})\Theta(\varepsilon) = \Theta(1)$ in the case where j is an exit state. □

THEOREM 4.5. *After each execution of Step* 3,

(i) $u(c)$ *is the order of the expected sojourn time at* $S(c)$, *regardless of the current state in* $S(c)$;

(ii) *if* $\delta(C) \cap E_r' \neq \phi$ (*case* (i)), *then the conditional exit probability from* $S(c)$ *is* $\Theta(1)$ *for each transition associated with r-edges in* $\delta(C)$, *and* $\Theta(\varepsilon)$ *for those associated with e-edges of* $\delta(C)$, *regardless of the current state in* $S(c)$;

(iii) *if* $\delta(C) \subseteq E_e'$ (*case* (ii)), *then the conditional exit probability from* $S(c)$ *is* $\Theta(1)$ *for each transition associated with edges in* $\delta(C)$, *regardless of the current state in* $S(c)$.

Proof. The proof is by induction on the number of executions of Step 3. Lemmas 4.1–4.4 establish the claims of the theorem for the first execution. Next, we consider an arbitrary execution assuming the theorem holds for the previous executions.

1. Let $v \in C$ be a vertex and let $i \in S(v)$ be a state. Also, let $u_i(v)$ be the order of the expected sojourn time at $S(v)$ given i as the current state. By the induction assumption, $u_i(v) = u(v)$ is independent of i for every $i \in S(v)$. We wish to next show that $u_i(c)$ is independent of i for all $i \in C$ and that $u_i(c) = u(c)$ (the value generated by the algorithm). Suppose that this is not the case. Following the r-edges defining C, at least one corresponds to $v, w \in C$, with $u_g(c) < u_h(c)$, where $g \in S(v)$ and $h \in S(w)$. Two possibilities exist.

(a) We have that $P_{gh}(\varepsilon) = \Theta(1)$. Considering this transition probability, we obtain that $\Theta(\varepsilon^{-u_g(c)}) \geq \Theta(1) * \Theta(\varepsilon^{-u_h(c)})$ so $u_g(c) \geq u_h(c)$, a contradiction.

(b) We have that $P_{gh}(\varepsilon) = \Theta(\varepsilon)$. Since now the transition $g \to h$ is associated with an r-edge, g belonged to some cycle C' that was condensed at some earlier execution of Step 3, Case (ii). By the induction hypothesis on part (iii) of Theorem 4.5, the probability of an exit from C' by an $g \to h$ conditional on an exit from C', is $\Theta(1)$. Again, it follows that $\Theta(\varepsilon^{-u_g(c)})$ is at least $\Theta(1) * \Theta(\varepsilon^{-u_h(c)}) = \Theta(\varepsilon^{-u_h(c)})$, so that $u_g(c) \geq u_h(c)$, a contradiction.

2. Suppose that the Markov process initiates at C, and consider the clock that runs only while transitions between vertices in C or outward of C occur (and thus transitions between states belonging to the same vertex do not progress the clock). This is not a Markov process, as the transition out of a vertex may depend on the entering state. However, by the induction assumptions on parts (ii) and (iii) of the theorem, the orders of these transition probabilities are independent of the entering state. Thus, if C satisfies the condition of Case (i) (respectively, Case (ii)) in Step 3, then, as in Lemma 4.1 (respectively, Lemma 4.2), the expected number of visits at each of the vertices of C (in the new clock) until an exit from C is $\Theta(1)$ (respectively, $\Theta(\varepsilon^{-1})$). Thus, returning to the original clock, the expected time until an exit from C (regardless of the current state in C) is $\Theta(1) * \Theta(\varepsilon^{-\max_{v \in C} u(v)})$ (respectively, $\Theta(\varepsilon^{-1}) * \Theta(\varepsilon^{-\max_{v \in C} u(v)})$), which is of the order of magnitude $\max_{v \in C} u(v)$ (respectively, $1 + \max_{v \in C} u(v)$). Indeed, this is the value computed by algorithm.

3. Parts (ii) and (iii) of the theorem are the counterparts of Lemmas 4.3 and 4.4., respectively, when considering the above-mentioned new clock. For a formal proof, the

argument of these lemmas may be repeated, in addition to the need to condition on the initial state in C. However, by the induction hypothesis, the orders of magnitude are not a function of the entering state and hence the result follows. □

LEMMA 4.6. *Step* 4 *of the algorithm determines the order of magnitude of the time until absorption at* s *for all vertices in the final graph.*

Proof. Step 4 of the algorithm is reached when G' contains no r-cycles. Also, note that s can be reached from all vertices via an r-path. Since G' has no r-cycles, then regardless of the initial state, the expected number of moves between vertices until the process reaches s is $\Theta(1)$. To see this, note that the vertices can be numbered in such a way that an r-edge from vertex i to vertex j exists only if $i < j$. In particular, s obtains the maximum index. Hence the probability of moving backward is either zero or $\Theta(\varepsilon)$.

Let $m = \max\{u(i) | i \in V'\}$. Starting at a vertex j with $u(j) = m$, each move to a new vertex contributes expected time until absorption of the order of magnitude of at most m, with this value being achieved at least once. Since, in expectation, there are $\Theta(1)$ such moves, we conclude by a simple renewal argument that the expected time until absorption in state s is $\Theta(\varepsilon^{-m})$. Hence the algorithm determines the correct values for those vertices that were first to be deleted from T. To see that, the order m applies for those vertices from which j can be reached by an r-edge is now immediate.

Consider the case when an e-edge exists from vertex i to j (but no r-path from i to j exists). There is an $\Theta(\varepsilon)$ probability to enter j, and this contributes to the expected time until absorption at state s, while initiating at state i a value of $\Theta(\varepsilon) * \Theta(\varepsilon^{-m}) = \Theta(\varepsilon^{-(m-1)})$. As a larger order cannot be achieved, this is the order of time until absorption at s that such a vertex has. These arguments extend inductively to other vertices to prove the lemma. □

COROLLARY 4.7. *Steps* 5 *and* 6 *correctly determine* $u(i)$ *for all states* $i \in V$.

Proof. The proof for $i \neq s$ is immediate from Lemma 4.6 and part (i) of Theorem 4.5. Then the claim follows for $i = s$ by a simple expectation argument. □

Complexity of the algorithm. The complexity of the algorithm is dominated by Step 3. This step can be executed in $O(n^2)$ time (cf. Fox and Landi [5]), so that an $O(n^3)$-time complexity obtains for a fixed target state and $O(n^4)$ when all target states are required. We suspect, however, that sophisticated dynamic data structures can be used to reduce this bound.

5. Generalization to a nonlinear perturbation. The algorithm can be extended to solve a more general problem. Let $P(0) \in R^{n \times n}$ be a stochastic matrix representing transition probabilities in a Markov chain. Let $A(x) \in R^{n \times n}$ have zero row-sums for $x \in X$. X is a set of positive real numbers that are not necessarily integers. For all real ε, $0 < \varepsilon \leq \varepsilon_{\max}$, assume that $P(\varepsilon) \equiv P(0) + \sum_{x \in X} \varepsilon^x A(x)$ are stochastic matrices representing transition probabilities in irreducible Markov chains. As before, we are interested in computing $u(i)$, the order of magnitude of the expected time to reach a given state s given an initial state i.

The edge-set of the graph G is now associated with the number k_{ij} denoting the order of the transition probability from i to j. The value of k_{ij} is set to zero if $P_{ij}(0) > 0$, and in this case where $(i, j) \in E_r$. Else, k_{ij} is set to the lowest index x for which $A_{ij}(x) > 0$, if such an index exists. Since the sum of the probabilities on the outgoing transitions from a state is 1, there is at least one r-edge (i, j) for every $i \in V$. After deleting the edges, leaving s, it is true, as was obvious in the special case above, that if G' contains no r-cycle then there is an r-path from every $i \neq s$ to s.

The following modifications are needed in the algorithm.

Replace cases (i) *and* (ii) *of Step* 3 *by the following lines.* Set $k \leftarrow \min\{k_{ij} | (i,j) \in \delta(C)\}$. Set $u(c) \leftarrow k + \max\{u(i) | i \in C\}$. Set $k_{ij} \leftarrow k_{ij} - k$ for all $(i,j) \in \delta(C)$. Modify E_r by adding to it edges $(i,j) \in \delta(C)$ that now have $k_{ij} = 0$.

Replace Step 4. Let $u(j) = \max\{u(i) | i \in T\}$. Delete j from T (thus turning $u(j)$ into permanent for j). For edges (i,j) where $i \in T$, set $u(i) \leftarrow \max\{u(i), u(j) - k_{ij}\}$. If $T = \phi$, go to Step 5. Else, repeat Step 4.

Replace Step 6. Set $u(s) \leftarrow \max\{u(i) - k_{si} | (s,i) \in E\}$.

It is obvious that the generalization of the problem is a natural one and that it leads to a simpler presentation of the algorithm. We have chosen to present the special case of linear perturbation because of its more obvious applicability and also because the proofs are somewhat simpler.

Another obvious extension of the problem is that in which the time from entrance to a state to the next transition is state-dependent. In this case, numbers $t(i)$ $i \in V$ are given, representing the order of the expected time from entrance to i till the next transition. Hardly any modification is needed in the algorithm to account for this case. The only change is at the initialization step, where, instead of starting with $u(i) \leftarrow 0$ for all $i \in V$, we start with $u(i) \leftarrow t(i)$ for all $i \in V$.

6. Background on Markov chains. Next, we briefly introduce some preliminaries concerning Markov chains (see Meyer [12]). Let $P \in R^{n \times n}$ be a stochastic matrix representing transition probabilities in the Markov chain. If P is irreducible (or ergodic), then a unique positive probability vector π with $\pi = \pi P$ exists. It is called the *stationary distribution* associated with P. The stochastic matrix P is also associated a matrix M, which is the matrix of mean passage times; namely, M_{ij} is the expected time until the process first hits state j when it initiates at i. In particular, $M_{ii} = 1/\pi_i$. Denote by Y the *deviation matrix* of P. This is the unique matrix Y satisfying the following three requirements:

$$(I - P)Y(I - P) = I - P,$$

$$Y(I - P)Y = Y,$$

$$Y(I - P) = (I - P)Y.$$

Furthermore, for each pair of states i and j,

$$\frac{M_{ij}}{M_{jj}} = \delta_{ij} + Y_{jj} - Y_{ij}, \tag{6.1}$$

where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise. This relation between the matrices M and Y implies also that $Y_{jj} \geq Y_{ij}$ for all i and j. If P is aperiodic (or regular), then

$$Y = \lim_{N \to \infty} \left[\sum_{t=0}^{N} P^t - (N+1)E \right],$$

where E is matrix whose rows coincide with π. This representation of the deviation matrix gives its probabilistic interpretation. Also, $Y = [I - (P - E)]^{-1} - E$. Finally, zero is an eigenvalue of Y with $\pi Y = \underline{0}$ and with $Y\underline{1} = \underline{0}$, where $\underline{0}$ (respectively, $\underline{1}$) is a vector all its entries are 0 (respectively, 1).

7. Series expansion for the stationary distribution. We now return to nearly uncoupled stochastic matrices with linear perturbation as defined in §§1–4. Let $\pi(\varepsilon)$ be the stationary distribution of $P(\varepsilon)$. Schweitzer [16] showed that, for all $\varepsilon > 0$ small enough,

$$\pi(\varepsilon) = \sum_{i=0}^{\infty} \pi^{(i)} \varepsilon^i$$

for some sequence $\{\pi^{(i)}\}_{i=0}^{\infty}$. Also, $\pi^{(i)} = \pi^{(0)} U^i$ for some matrix U. Hence finding $\pi^{(0)}$ is a key for a complete description of the series expansion of $\pi(\varepsilon)$.

To determine the terms of such a series expansion and, in particular, to find the leading term, consider the identity $\pi(\varepsilon) = \pi(\varepsilon)(P(0) + \varepsilon A)$. It is clear then that $\{\pi^{(i)}\}_{i=0}^{\infty}$ solves the following systems of difference equations:

$$\pi^{(0)}(I - P(0)) = 0,$$

and, for $j = 0, 1, \ldots$

$$\pi^{(j+1)}(I - P(0)) - \pi^{(j)} A = 0.$$

We refer to the above as the *systems of fundamental equations.* If $P(0)$ is irreducible, then the first set of equations $\pi^{(0)}(I - P(0) = 0$ (plus the needed normalization) is sufficient to determine $\pi^{(0)}$ uniquely. However, if $P(0)$ is decomposable, then a larger number of sets of fundamental equations is needed to determine $\pi^{(0)}$ uniquely, and the question of how many sets are needed to determine $\pi^{(0)}$ uniquely arises. By having this value in advance, we can improve the naive approach of adding a set of fundamental equations one at a time and solving the resulting system until a unique (up to a normalization constant) $\pi^{(0)}$ emerges.

We argue that, for a decomposable $P(0)$, the above question is related to the question of mean passage times at $P(\varepsilon)$ for small values of ε and their orders of magnitude. Recall that $M(\varepsilon)$ and $Y(\varepsilon)$ are the mean passage time matrix and the deviation matrix, respectively, of $P(\varepsilon)$. The following result is taken from Schweitzer [16].

LEMMA 7.1. *$M(\varepsilon)$ and $Y(\varepsilon)$ admit Laurent series expansion.*

Proof. It is well known (see, e.g., Meyer [12]) that $M(\varepsilon)$ is the unique matrix in $R^{n \times n}$ satisfying $(I - P(\varepsilon))M(\varepsilon) = J - P(\varepsilon)M_{dg}(\varepsilon)$, where J is a matrix, all its entries are one, and $M_{dg}(\varepsilon)$ is a diagonal matrix whose diagonal coincides with the diagonal of $M(\varepsilon)$. Solving for $M(\varepsilon)$, say by Cramer's rule, then, for each i, j, $M_{ij}(\varepsilon)$ is the ratio between two polynomials in ε and hence it admits a Laurent expansion.[5] This completes the proof for $M(\varepsilon)$. For the matrix $Y(\varepsilon)$ the result follows from the above proof coupled with (6.1). □

It is clear that $M_{ij}(\varepsilon) = \Theta(\varepsilon^{-u_{ij}})$ where u_{ij} is the order of the pole of $M_{ij}(\varepsilon)$. Thus this order can be computed via the algorithm presented in §3.

THEOREM 7.2. *Let u_{ij} be the order of the pole of $M_{ij}(\varepsilon)$ at zero. Similarly, let v_{ij} be the corresponding order of $Y_{ij}(\varepsilon)$.*[6] *Then,*

$$\max_{ij}(u_{ij} - u_{jj}) = \max_{ij} v_{ij}.$$

[5] Moreover, the order of the pole is bounded by n since the polynomial in the numerator cannot have a larger degree.

[6] Note that, in the case of an analytic function, the order is zero.

Proof. As for any irreducible P, the corresponding π and Y satisfy $\pi Y = 0$, and, as $Y_{jj} \geq Y_{ij}$, the maximal order of the poles over $Y_{jj}(\varepsilon) - Y_{ij}(\varepsilon)$ coincides with the corresponding maximization over $Y_{ij}(\varepsilon)$. The proof is now completed by relation (6.1). □

It was shown in Haviv and Ritov [7] that the maximal value over the order of the poles of the entries of $Y(\varepsilon)$ at zero equals the minimal number of sets of fundamental equations needed to determine $\pi^{(0)}$ uniquely, minus one.[7] By Theorem 7.2, this value can be computed from the orders of the poles at zero of the entries of $M(\varepsilon)$. Thus we have the following theorem.

THEOREM 7.3. *The minimal number of sets of fundamental equations needed to construct a system of equations whose corresponding solution to* $\pi^{(0)}$ *is unique (up to a normalizing constant) is* $\max_{ij}(u_{ij} - u_{jj}) + 1$.

8. An example. Let

$$P(\varepsilon) = P(0) + \varepsilon A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} + \varepsilon \begin{pmatrix} 0 & -1 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

Recall that

$$E_r = \{(i,j) | P_{ij}(0) > 0\}, \quad \text{and} \quad E_e = \{(i,j) | A_{ij} > 0\}.$$

This information is summarized by the following graph where r-edges are represented by bold arrows and e-edges by dashed arrows. See Fig. 1.

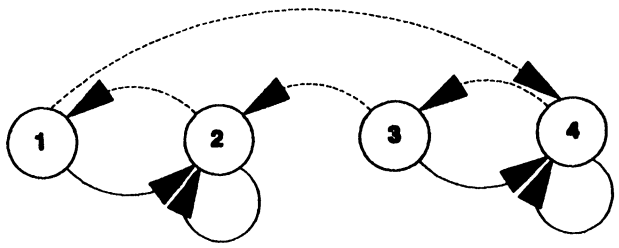


FIG. 1

The Markov chain associated with $P(0)$ is has two independent chains (states 2 and 4, which are absorbing states) and two transient states (states 1 and 3). We proceed through the steps of the algorithm with the target state $s = 4$. Edges emanating form vertex 4 are ignored by the first part of the algorithm, so that we consider Fig. 2, below.

The algorithm initiates with the values $u(1) = u(2) = u(3) = 0$. There is only one r-cycle: (2). All edges emanating from it are e-edges, so that Case (ii) occurs in Step 3. Hence $u(2)$ is set to 1, and the next multigraph to be considered is Fig. 3.

Now there is an r-cycle (2,1). It is condensed to a vertex c. As all edges emanating from it are e-edges, Case (ii) of Step 3 occurs again, and $u(c)2$ is set to 2. See Fig. 4.

[7] In Schweitzer [17] it is stated that the maximal order of the poles is an upper bound for the number of fundamental equations needed to determine $\pi^{(0)}$.

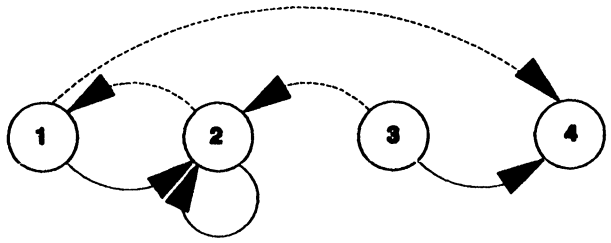


FIG. 2

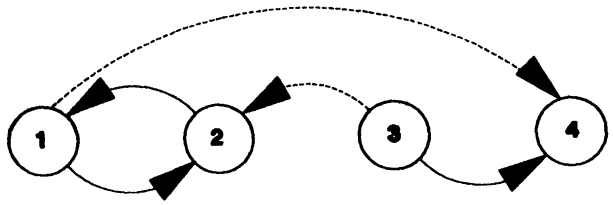


FIG. 3

At this stage, no more r-cycles exist, and the algorithm moves to Step 4. Then the algorithm seeks the set of vertices currently having the largest value and makes this order permanent. Thus $u(c) = 2$ and $T = \{3, 4\}$. Next, the algorithm seeks vertices in T from which vertex c can be reached via paths that are constructed only of r-edges. There are no such vertices in our example. Then it seeks those vertices from which c can be reached in one step via an e-edge. Vertex 3 is such a vertex. It gives it the value of 1, which becomes permanent in the next execution of Step 4. Then $T = \phi$, and the algorithm moves to Step 5. There, it sets $u(1) = u(2) = 2$ and $u(3) = 1$. Finally, in Step 6, Fig. 5, below, is considered.

Then $u(4) = 0$ is computed. Thus $u_{14} = 2$, $u_{24} = 2$, $u_{34} = 1$ and $u_{44} = 0$.

For state 3 as the target state, the algorithm obtains that $u_{13} = 2$, $u_{23} = 2$, $u_{33} = 1$, and $u_{43} = 1$. The other values can be found by symmetry. Hence $\max_{ij}(u_{ij} - u_{jj}) = 2$, and three sets of fundamental equations, $\pi^{(0)}(I - P(0)) = 0$, $\pi^{(1)}(I - P(0)) = \pi^{(0)}A$, and $\pi^{(2)}(I - P(0)) = \pi^{(1)}A$, are required to obtain a system of equations in which the $\pi^{(0)}$ component of a solution $(\pi^{(0)}, \pi^{(1)}, \pi^{(2)})$ is unique (up to a normalizing constant). As is solved in Haviv and Ritov [7], $\pi^{(0)} = (0, 1/2, 0, 1/2)$.

Acknowledgments. The authors thank Y. Ritov and P. J. Schweitzer for their helpful observations.

REFERENCES

[1] M. CODERCH, A. S. WILLSKY, S. S. SASTRY, AND D.A. CASTANON, *Hierarchical aggregation of singularity perturbed finite state Markov processes*, Stochastics, 8 (1983), pp. 259–289.

[2] ———, *Hierarchical aggregation of linear systems with multiple time scales*, IEEE Trans. Automat. Control, AC–28 (1983), pp. 1017–1030.

[3] P. J. COURTOIS, *Decomposability: Queueing and Computing Systems*, Academic Press, New York, 1977.

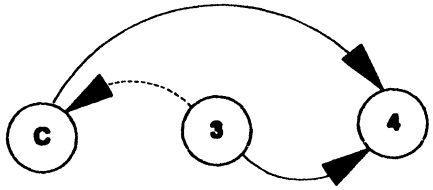


FIG. 4

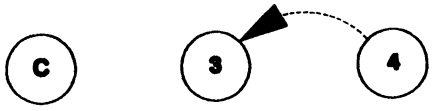


FIG. 5

[4] F. DELEBECQUE, *A reduction process for perturbed Markov chains*, SIAM J. Appl. Math., 43 (1983), pp. 325–350.
[5] B. L. FOX AND D. M. LANDI, *An algorithm for identifying the ergodic subchains and transient states for stochastic matrices*, Comm. Assoc. Comput. Mach., 11 (1968), pp. 619–621
[6] V. G. GAITSGORY AND A. A. PERVOZVANSKY, *Aggregation of systems in Markov chains with weak interactions*, Kibernetika, 11 (1975), pp. 91–98. (In Russian.) (pp. 441–450 in the English translation.)
[7] M. HAVIV AND J. RITOV, *Series expansion for stochastic matrices*, unpublished.
[8] T. KATO, *Perturbation Theory for Linear Operators*, Springer-Verlag, Berlin, New York, 1966.
[9] C. E. LANGENHOP, *The Laurent expansion for a nearly singular matrix*, Linear Algebra Appl., 4 (1971), pp. 329–340.
[10] G. LATOUCH AND G. LOUCHARD, *Return times in nearly decomposable stochastic processes*, J. Appl. Prob., 15 (1978), pp. 251–267.
[11] G. LOUCHARD AND G. LATOUCH, *Random times in nearly completely decomposable transient Markov chains*, Cahiers du C.E.R.D., 24 (1982), pp. 321–352.
[12] C. D. MEYER, *The role of the group generalized inverse in the theory of finite Markov chains*, SIAM Rev., 17 (1975), pp. 443–464.
[13] J. R. ROHLICEK AND A. S. WILLSKY, *Structural decomposition of multiple time scale Markov processes*, in Proc. of the 25th Allerton Conference on Communications, Control and Computing, pp. 674–683, University of Illinois, Urbanna, IL, pp. 674–683.
[14] ———, *The reduction of Markov generators: An algorithm exposing the role of transient states*, J. Assoc. Comput. Mach., 35 (1988), pp. 675–696.
[15] P. J. SCHWEITZER, *Perturbation theory and finite Markov chains*, J. Appl. Probab., 5 (1968), pp. 401–413.
[16] ———, *Perturbation series expansions of nearly completely decomposable Markov chains*, Working Papers Series No. 8122, The Graduate School of Management, The University of Rochester, Rochester, NY, 1981.
[17] ———, *Perturbation series expansions of nearly completely decomposable Markov chains*, in Teletraffic Analysis and Computer Performance Evaluation, O. J. Boxma, J. W. Cohen and H. C. Tijm, eds., Elsevier, North–Holland, Amsterdam, 1986.
[18] H. A. SIMON AND A. ANDO, *Aggregation of variables in dynamic systems*, Econometrica, 29 (1961), pp. 111–138.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 398–412, August 1992
© 1992 Society for Industrial and Applied Mathematics
009

COMPARING QUEUES AND STACKS AS MECHANISMS FOR LAYING OUT GRAPHS*

LENWOOD S. HEATH†, FRANK THOMSON LEIGHTON‡, AND ARNOLD L. ROSENBERG§

Abstract. The relative powers of queues and stacks are compared as mechanisms for laying out the edges of a graph. In a k-queue layout, vertices of the graph are placed in some linear order (also called a linear arrangement), and each edge is assigned to exactly one of the k queues, so that the edges assigned to each queue obey a first-in/first-out (FIFO) discipline. As the vertices are scanned left to right, an edge is enqueued on its assigned queue when its left endpoint is encountered and is dequeued from its queue when its right endpoint is encountered. In a k-stack layout, vertices of the graph are placed in some linear order, and each edge is assigned to exactly one of the k stacks so that the edges assigned to each stack obey a last-in/first-out discipline. As the vertices are scanned left to right, an edge is pushed on its assigned stack when its left endpoint is encountered and is popped from its stack when its right endpoint is encountered.

The paper has three main results. First, a tradeoff between queuenumber and stacknumber is shown for a fixed linear order of the vertices of G. In particular, for a fixed-order layout of a graph G,

$$\text{queuenumber} \times \text{stacknumber} \geq \text{cutwidth}/\text{valence}(G).$$

Second, it is shown that every 1-queue graph has a 2-stack layout and that every 1-stack graph has a 2-queue layout. Third, in a surprising display of the power of queues, it is shown that the ternary hypercube requires exponentially more stacks than queues. More precisely, an N-vertex ternary hypercube has a $(2\log_3 N)$-queue layout but requires $\Omega\left(N^{1/9-\epsilon}\right)$ stacks, $\epsilon > 0$, in any stack layout. Also, some asymptotic bounds for the queuenumber of bounded-valence graphs are derived.

Key words. queue layout, stack layout, book embedding, graph embedding, ternary hypercube

AMS(MOS) subject classifications. 05C99, 68R10, 94C15

1. Introduction.

A recurring theme in the study of computing mechanisms compares and contrasts the relative powers of queues and stacks. In computability theory, it is well known that two stacks endow an off-line finite-state control with the full power of a Turing machine, whereas a single stack does not; in contrast, a single queue suffices to yield the full power of a Turing machine. In quite another domain, Tarjan [15] characterizes the numbers of queues or stacks needed to realize given permutations with a network of parallel queues or stacks. In the case that the entire permutation must be completely loaded into the queues or stacks, then completely unloaded, Tarjan shows, in a sense that he makes precise, that queues and stacks are dual mechanisms for realizing permutations. In a similar vein, Even and Itai [5] use queues and stacks to linearize graphs in much the way that we study here. They relate the problem of realizing a permutation with a network of parallel queues or stacks to the problem of coloring a permutation graph. (Given a permutation $\pi : \{1, \cdots, n\} \rightarrow \{1, \cdots, n\}$, its *permutation graph* has vertex set $\{1, \cdots, n\}$ and

* Received by the editors December 29, 1990; accepted for publication (in revised form) July 18, 1991. This research was supported by National Science Foundation grants DCI-87-96236, CCR-88-12567, and CCR-90-09953.

† Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061. Part of the research of this author was conducted while at the University of North Carolina at Chapel Hill and at the Massachusetts Institute of Technology.

‡ Department of Mathematics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

§ Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003.

an edge (i,j) whenever $i < j$ and $\pi^{-1}(i) < \pi^{-1}(j)$ or $i > j$ and $\pi^{-1}(i) > \pi^{-1}(j)$.) In the case where loading and unloading may proceed simultaneously, the problem of minimizing the number of *queues* required for a given permutation remains the permutation graph coloring problem. In the same case, Even and Itai show that the problem of minimizing the number of *stacks* required is equivalent to coloring a circle graph, hence is NP-complete [7]. (A *circle graph* is the intersection graph of a set of chords of a circle.) The focus of this paper is to contrast queue-based layouts of graphs with stack-based layouts. Stack-based layouts have been studied extensively, under the aegis of the problem of *embedding graphs in books* [1], [3].

Further motivation for the study of queue and stack layouts originates in the DIOGENES design methodology [14], which is a proposal by one of the authors for implementing fault-tolerant arrays of parallel processors. In DIOGENES, an array of communicating processors is implemented in a conceptual line, and some number of hardware queues and/or stacks pass over the entire line. The queues and/or stacks implement the communication links among processors in such a way that faulty processors are ignored, and all good processors are utilized. If the processors and their connections are represented by an undirected graph, then the DIOGENES layout problem is equivalent to a graph layout problem, where edges are assigned to conceptual queues and/or stacks.

A final motivation for this study is a problem arising in the scheduling of parallel processors. Consider the following simple model of scheduling parallel computations in an architecture-independent fashion; cf. [13]. We represent the computation to be scheduled as a directed acyclic graph (dag) whose nodes represent the processes to be executed and whose arcs indicate computational dependencies: a process-node cannot be executed until all of its predecessors in the dag have been executed. Processes are queued up in a first-in/first-out (FIFO) *processor queue* (PQ) as they become eligible for execution; each idle processor "grabs" the process at the head of the PQ. Our study focuses on the management of data in this scenario: where will the inputs to process P be when P is "grabbed" by a processor? Our solution is to have the PQ be coordinated with a *data manager* (DM), which itself is a collection of FIFO queues: When a process terminates, it places its "outputs" on the queues of the DM in such a way that when process P is "grabbed" by a processor, all inputs to P are at the heads of the DM queues. Our queue-based graph linearization problem idealizes this approach to the scheduling problem: The computation dag is the graph to be linearized; the linearization process implicitly specifies the loading of the PQ; the queues that control the linearization comprise the DM. Here we idealize the problem even a step further by replacing the computation dag by an ordinary (undirected) graph.

A *k-queue layout* of an undirected graph $G = (V, E)$ has two aspects. The first aspect is a linear order of V (which we think of as being on a horizontal line). The second aspect is an assignment of each edge in E to one of k queues in such a way that the set of edges assigned to each queue obeys a FIFO discipline. Think of scanning the vertices in order from left to right. When the left endpoint of an edge is encountered, the edge enters its assigned queue (at the back of the queue). When the right endpoint of an edge is encountered, the edge exits its assigned queue (and must therefore be at the front of the queue). If a queue is examined at any instant, the edges in the queue are in the order of their right endpoints, with the leftmost of those right endpoints belonging to edges at the head of the queue.

More formally, a *k-queue layout* of an n-vertex undirected graph $G = (V, E)$

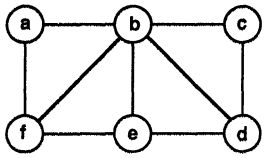


FIG. 1.1. *Example graph G.*

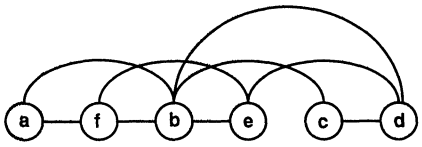


FIG. 1.2. 1-*queue layout.*

consists of a linear order of V, denoted $\sigma = 1, 2, \cdots, n$, and an assignment of each edge in E to exactly one of k queues, $q_1, \cdots, q_k$. Each queue q_j operates as follows. The vertices of V are scanned in left-to-right (ascending) order. When vertex i is encountered, any edges assigned to q_j that have vertex i as their right endpoint must be at the front of that queue; they are removed (dequeued). Any edges assigned to q_j that have vertex i as left endpoint are placed on the back of that queue (enqueued), in ascending order of their right endpoints. k is the *queuenumber* of the layout. The *queuenumber* of G, $\mathrm{QN}(G)$, is the smallest k such that G has a k-queue layout; G is said to be a *k-queue* graph. The freedom to choose the order of V and the assignment of E so as to optimize the queuenumber (or some other measure) of the resulting layout constitutes the essence of the queue layout problem.

As an example of a 1-queue layout, consider the graph G in Fig. 1.1. A 1-queue layout of G is shown in Fig. 1.2. The linear order of V is a, f, b, e, c, d. The order in which edges pass through the single queue is

$$(a, f), (a, b), (f, b), (f, e), (b, e), (b, c), (b, d), (e, d), (c, d).$$

Note that edges having the same left endpoint enter the queue in an order determined by their right endpoints. For example, edge (a, f) must enter the queue before edge (a, b) since f is to the left of b.

Dually, a *k-stack layout* of graph G also has two aspects. The first aspect is again a linear order of V. The second aspect is an assignment of each edge in E to one of k stacks in such a way that the set of edges assigned to each stack obeys a last-in/first-out discipline.

More formally, a *k-stack layout* of an undirected graph consists of a linear order of V and an assignment of each edge in E to exactly one of k stacks, $s_1, \cdots, s_k$. Each stack s_j operates as follows. The vertices of V are scanned in left-to-right (ascending) order. When vertex i is encountered, any edges assigned to s_j that have vertex i as their right endpoint must be on the top of that stack; they are removed (popped). Any edges assigned to s_j that have vertex i as left endpoint are placed on the top of that

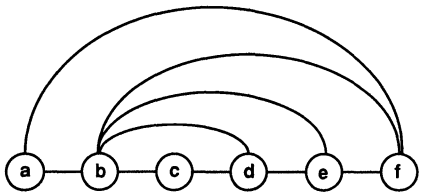


FIG. 1.3. *1-stack layout.*

stack (pushed), in descending order of their right endpoints. k is the *stacknumber* of the layout. The *stacknumber* of G, $\mathrm{SN}(G)$, is the smallest k such that G has a k-stack layout; G is said to be a *k-stack* graph. Unlike a queue layout, edges do not exit a stack in the same order in which they enter it.

As an example, Fig. 1.3 shows a 1-stack layout of the graph G in Fig. 1.1. The linear order of V is a, b, c, d, e, f. The order in which edges enter the stack is

$$(a,f),(a,b),(b,f),(b,e),(b,d),(b,c),(c,d),(d,e),(e,f).$$

The order in which edges exit the stack is

$$(a,b),(b,c),(c,d),(b,d),(d,e),(b,e),(e,f),(b,f),(a,f).$$

One goal of this study is to find graphs for which the queuenumber and stacknumber differ significantly. A second goal is to improve our insight into when queues (stacks) are easier to use than stacks (queues). Some contrasts are found in the work of Heath and Rosenberg [8] on queue layouts. 1-queue graphs, like 1-stack graphs, have a characterization as a class of planar graphs. (We review this characterization in §2.) However, while 1-stack graphs can be recognized in linear time, the recognition problem for 1-queue graphs is NP-complete. On the other hand, when the vertex order of a layout is fixed, the minimum number of queues required for the layout is easily characterized and found in polynomial time, while the problem of finding the minimum number of stacks is NP-complete [5], [7].

The queuenumbers of a number of familiar classes of graphs are determined in [8], with one exception. See Table 1. In all cases save one, the number of queues required is no more than the number of stacks. The sole exception is the open problem of the queuenumber of planar graphs. In the case of the complete bipartite graph $K_{m,n}$, the queuenumber is determined to be exactly $\min(\lceil m/2 \rceil, \lceil n/2 \rceil)$, while the best upper bound known for the stacknumber is significantly higher, $\lceil (m+2n)/4 \rceil$ [11].

In this paper, we obtain the following results comparing queue and stack layouts. Initially, we find instances where the powers of queues and stacks are roughly equal. We prove that every 1-queue graph has a 2-stack layout, and that every 1-stack graph has a 2-queue layout. Moreover, we find that the asymptotic results for graphs of bounded valence (maximum vertex degree) are identical for queues and for stacks. These two results lead us to hope that, in fact, queues and stacks are equally powerful graph-layout mechanisms, in the sense that every graph that admits a k-queue layout admits an $O(k)$-stack layout, and vice versa. We prove that such equivalence results cannot possibly be proved using the same linear ordering of graph vertices for both the queue and stack layouts, by establishing a queuenumber-stacknumber tradeoff for any fixed layout of a graph, as follows:

$$\text{queuenumber} \times \text{stacknumber} \geq \text{cutwidth}/\text{valence}(G).$$

TABLE 1
Queuenumbers of specific graphs.

Graph class	Queuenumber [8]	Stacknumber
Trees	1	1 [3]
X-trees	2	2 [3]
DeBruijn graph	2	≤ 5 [12]
Complete graph K_n	$\lfloor n/2 \rfloor$	$\lfloor n/2 \rfloor$ [3]
Complete bipartite graph $K_{m,n}$	$\min(\lceil m/2 \rceil, \lceil n/2 \rceil)$ (exact)	$\leq \lceil (m+2n)/4 \rceil$ [11]
FFT network	2	3 [6]
Benes network	2	3 [6]
Boolean n-cube	$\leq n-1$	$\leq n-1$ [3]
Planar graphs	Unknown (conjecture bounded)	4 [16]

Moreover, by studying layouts of the ternary hypercube, we find that, in fact, no such equivalence result exists! While an N-vertex ternary hypercube can be laid out with $2\log_3 N$ queues, any stack layout requires $\Omega\left(N^{1/9-\epsilon}\right)$ stacks, $\epsilon > 0$.

The organization of this paper is as follows. The second section reviews the necessary results from [8]. Section 3 contains our queuenumber/stacknumber tradeoff for fixed order layouts. In §4 we show that every 1-queue graph has a 2-stack layout, and that every 1-stack graph has a 2-queue layout. Section 5 dualizes previously-known asymptotic results for stack layouts to queue layouts. Section 6 contains the ternary hypercube as an example of an exponential tradeoff between queuenumber and stacknumber.

2. Basics. This section reviews some needed results from [8].

2.1. Fixed-order layouts. In this section, we fix an order $\sigma = 1, 2, \cdots, n$ of V and examine the difficulty of minimizing the number of queues or the number of stacks required to complete σ to a layout. We concentrate on sets of edges that are obstacles to minimizing the number of stacks or queues. A *k-rainbow* is a set of k edges

$$\{e_i = (a_i, b_i),\ 1 \leq i \leq k\}$$

such that

$$a_1 < a_2 < \cdots < a_{k-1} < a_k < b_k < b_{k-1} < \cdots < b_2 < b_1;$$

in other words, a rainbow is a *nested* matching. A *k-twist* is a set of k edges

$$\{e_i = (a_i, b_i),\ 1 \leq i \leq k\}$$

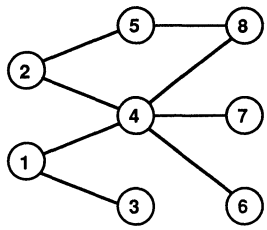


FIG. 2.1. *A leveled-planar graph.*

such that

$$a_1 < a_2 < \cdots < a_{k-1} < a_k < b_1 < b_2 < \cdots < b_{k-1} < b_k;$$

in other words, a twist is a fully intersecting matching.

A rainbow is an obstacle for a queue layout because no two nested edges can be assigned to the same queue.

PROPOSITION 2.1 (see [8]). *Suppose that σ has a k-rainbow. Then there is no queue layout of σ with fewer than k queues. There exists a stack layout of σ in which all edges of the k-rainbow are assigned to the same stack.*

A twist is an obstacle for a stack layout because no two intersecting edges can be assigned to the same stack.

PROPOSITION 2.2 (see [3]). *Suppose that σ has a k-twist. Then there is no stack layout of σ with fewer than k stacks. There exists a queue layout of σ in which all edges of the k-twist are assigned to the same queue.*

The largest rainbow in σ determines the smallest number of queues needed in a queue layout of σ. In fact, a queuenumber-optimal layout of σ can be found in polynomial time.

THEOREM 2.3 (see [8]). *If σ has no rainbow of more than k edges, then there is a k-queue layout for σ. Such a layout can be found in time $O(|E| \log\log |V|)$.*

2.2. A characterization of 1-queue graphs. Bernhart and Kainen [1] give a characterization of 1-stack graphs.

PROPOSITION 2.4 (see [1]). *G is a 1-stack graph if and only if G is outerplanar.*

(An *outerplanar* graph is a planar graph having a planar embedding in which all vertices appear on a common face.)

Heath and Rosenberg [8] show that the 1-queue graphs are also planar graphs that have a particular kind of planar embedding. For completeness, we repeat their characterization here.

Consider the normal Cartesian (x, y) coordinate system for the plane. For i an integer, let ℓ_i be the vertical line defined by $\ell_i = \{(i, y) \mid y \in \mathbf{R}\}$. A graph $G = (V, E)$ is a *leveled-planar graph* if V can be partitioned into levels $V_1, V_2, \cdots, V_m$ and G can be embedded in the plane in such a way that all vertices of V_i are on the line ℓ_i, each edge in E is embedded as a straight-line segment wholly between ℓ_i and ℓ_{i+1} for some i, and the embedding is a valid planar embedding for G (i.e., no edges cross). Figure 2.1 shows a leveled-planar graph having three levels. Note that the leveled-planar embedding of a leveled-planar graph is not unique. Henceforth, we assume that a

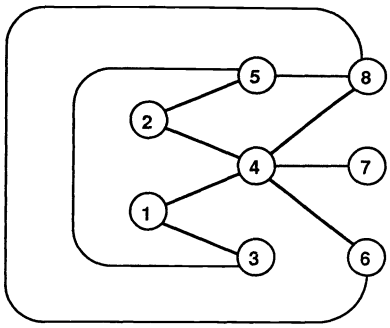


FIG. 2.2. *Drawing arches.*

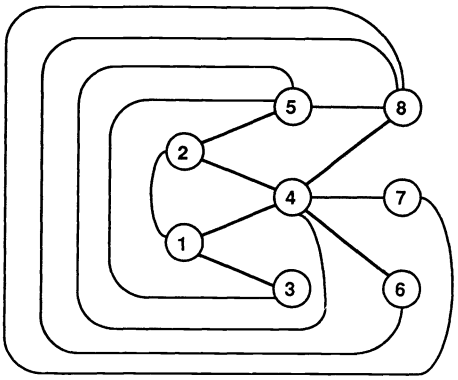


FIG. 2.3. *A maximal arched leveled-planar graph.*

valid (but arbitrary) leveled-planar embedding is given along with a leveled-planar graph.

A leveled-planar embedding induces an order (the *induced order*) on V as follows. As i takes the values $1, 2, \cdots, m$, scan line ℓ_i from bottom to top. Label the vertices $1, 2, \cdots, n$ as they are encountered. For $1 \leq i \leq m$, let b_i be the (bottom) first vertex in level i, and let t_i be the (top) last. Let s_i be the first vertex in level i that is adjacent to some vertex in level $i+1$, or, if there are no edges between levels i and $i+1$, let $s_i = t_i$. Consider augmenting G with new edges. A *level-i arch* for G is an edge connecting vertex t_i with some vertex j, where $b_i \leq j \leq \min(t_i - 1, s_i)$. A leveled-planar graph G, augmented by any number of arches, can be embedded in the plane by drawing the arches around level 1; because of the leveling, the arches do not cross. See Fig. 2.2, where edges $(3, 5)$ and $(6, 8)$ are arches. A leveled-planar graph augmented by (zero or more) arches is called an *arched leveled-planar graph.* The edges that are not arches are called *leveled edges.* An arched leveled-planar graph that cannot be augmented with further arches or leveled edges is *maximal.* See

Fig. 2.3 for an example. The above definitions for b_i, s_i, and t_i are used in §4 to refer to vertices in arched leveled-planar graphs.

We can now state the characterization of 1-queue graphs.

THEOREM 2.5 (see [8]). *A graph G is a 1-queue graph if and only if G is an arched leveled-planar graph.*

3. A queuenumber/stacknumber tradeoff. Let $\sigma = 1, 2, \cdots, n$ be a fixed order of the vertices of G. Intuitively, the *cutwidth* of σ is the maximum number of edges cut by any line perpendicular to σ. Formally, define the *cut at vertex i*, $1 \leq i \leq n-1$, to be the set of edges of G,

$$\mathrm{CUT}(i) = \{(j,k) \mid 1 \leq j \leq i < k \leq n\}.$$

The *cutwidth* of σ is $\mathrm{CW}(\sigma) = \max_i |\mathrm{CUT}(i)|$. The *valence* of G, denoted $\mathrm{valence}(G)$, is the maximum degree of a vertex of G.

We develop a tradeoff between the queuenumber and the stacknumber of σ using the following result of Erdös and Szekeres [4].

PROPOSITION 3.1. *Let P be the sequence $\pi(1), \cdots, \pi(n)$, where π is some permutation of $1, \cdots, n$. Let a be the length of the longest ascending subsequence in P, and let d be the length of the longest descending subsequence in P. Then $ad \geq n$.*

The tradeoff is based on finding an interesting matching in the graph.

THEOREM 3.2. *Let $\sigma = 1, 2, \cdots, n$ be a fixed order of the vertices of G. Then*

$$\mathrm{SN}(\sigma) \times \mathrm{QN}(\sigma) \geq \mathrm{CW}(\sigma)/\mathrm{valence}(G). \tag{1}$$

Proof. Choose a vertex i, $1 \leq i \leq n-1$, such that $|\mathrm{CUT}(i)| = \mathrm{CW}(\sigma)$. $\mathrm{CUT}(i)$ is the edge set of a bipartite graph H with $\mathrm{valence}(H) \leq \mathrm{valence}(G)$. Select a maximum matching $M \subseteq \mathrm{CUT}(i)$ in H. The size of M is at least $\mathrm{CW}(\sigma)/\mathrm{valence}(G)$.

The left vertices of M give an order to the edges in M, and the right vertices give some permutation π of that order. Let a and d be as required for Proposition 3.1. Then a gives the length of a longest similarly ordered sequence between left and right vertices of M; therefore M contains an a-twist. By Proposition 2.2, $\mathrm{SN}(\sigma) \geq a$. Similarly, M contains a d-rainbow. By Proposition 2.1, $\mathrm{QN}(\sigma) \geq d$. Finally, by Proposition 3.1,

$$\mathrm{SN}(\sigma) \times \mathrm{QN}(\sigma) \geq ad \geq |M| \geq \mathrm{CW}(\sigma)/\mathrm{valence}(G). \qquad \square$$

The factor $\mathrm{valence}(G)$ in (1) is necessary. Consider the *star graph* G with vertex set $\{1, 2, \cdots, n\}$ and edge set $\{(1,i) \mid 2 \leq i \leq n\}$. If $\sigma = 1, 2, \cdots, n$, then $\mathrm{SN}(\sigma) = 1$, $\mathrm{QN}(\sigma) = 1$, $\mathrm{CW}(\sigma) = n-1$, and $\mathrm{valence}(G) = n-1$.

4. Small numbers of queues and stacks. A graph $G = (V, E)$ is *subhamiltonian* if it is a subgraph of a planar graph G', and G' has a hamiltonian cycle. Bernhart and Kainen [1] provide a characterization of 2-stack graphs.

PROPOSITION 4.1 (see [1]). *A graph G has a 2-stack layout if and only if G is subhamiltonian.*

We can bound the stacknumber of a 1-queue graph.

THEOREM 4.2. *Every 1-queue graph has a 2-stack layout.*

Proof. Let $G = (V, E)$ be a 1-queue graph having $n \geq 3$ vertices. By Theorem 2.5, G has an arched leveled-planar embedding with some leveling of V, say $V_1, \cdots, V_m$. By Proposition 4.1, it suffices to show that G is subhamiltonian.

Because the stacknumber of a graph equals the maximum stacknumber of any of its biconnected components [3], we may assume that G is biconnected; so, in particular, none of the levels $V_2, \cdots, V_{m-1}$ is a singleton. We may also assume that G is a maximal arched leveled-planar graph. For each level i, add the *vertical* edges $(p, p+1)$, $b_i \leq p \leq t_i - 1$, that is, the edges that go along the line ℓ_i, connecting consecutive vertices of V_i. Let the resulting graph be $G' = (V, E')$. Clearly G' is planar; we claim that it is hamiltonian.

Note that, when $|V_i| > 2$, the (new) vertical edges on level i together with the arch (b_i, t_i) form a cycle on V_i. Call these edges the *level-i cycle edges*. These cycles on levels are nested in the planar embedding. Our strategy is to connect each pair of consecutive cycles by two leveled edges to obtain a hamiltonian cycle for G'.

By an induction on $m - i \geq 1$, we show that there is a particular kind of spanning cycle for levels $V_i, V_{i+1}, \cdots, V_m$. The inductive hypothesis is that there is a cycle C spanning levels $V_i, \cdots, V_m$ such that all but one of the level-i cycle edges are in C; if $|V_i| = 2$, then C contains the edge (b_i, t_i) (which is considered to be both a vertical edge and an arch).

For the base case where $i = m - 1$, there are three subcases. First, if $|V_m| = 1$, then let C be the spanning cycle

$$n, t_i, b_i, b_i + 1, \cdots, t_i - 1, n,$$

which contains all level-i cycle edges except $(t_i - 1, t_i)$. Second, if $|V_m| > 1$ and $|V_i| = 1$, then $i = 1$ (because of our assumption that G is biconnected), and G' is obviously hamiltonian. Third, if $|V_m| > 1$ and $|V_i| > 1$, then choose four vertices $p, p+1, q, q+1$ such that $p, p+1 \in V_i$, $q, q+1 \in V_m$, and $(p, q), (p+1, q+1) \in E$. Because G is maximal and $|V_i| > 1$, $|V_m| > 1$, this choice is always possible. Let C be the spanning cycle

$$p+1, \cdots, t_i, b_i, \cdots, p, q, \cdots, b_m, t_m, \cdots, q+1, p+1.$$

All level-i cycle edges except $(p, p+1)$ are in the spanning cycle; if $|V_i| = 2$, then $p = b_i$, $p+1 = t_i$, and $(p, p+1)$ is in C.

For the purpose of induction, assume that there is a spanning cycle C satisfying the inductive hypothesis for $V_{i+1}, \cdots, V_m$. We extend the spanning cycle to a spanning cycle C' for $V_i, \cdots, V_m$.

If $i = 1$ and $|V_i| = 1$, then choose some level-2 vertical edge $(p, p+1)$ that is in C. Construct C' from C by deleting $(p, p+1)$ and adding $(1, p)$ and $(1, p+1)$. A hamiltonian cycle for G' results.

Otherwise, $|V_i| > 1$. Let (x, y), $x < y$, be the level-$(i+1)$ vertical edge (if any) that is not in C. We wish to choose four vertices $p, p+1, q, q+1$ with the properties $p, p+1 \in V_i$, $q, q+1 \in V_{i+1}$, and $(p, q), (p+1, q+1) \in E$. If such a choice is possible so that $(q, q+1) \neq (x, y)$, then C can be extended to C' by removing edge $(q, q+1)$ and adding the path

$$q, p, \cdots, b_i, t_i, \cdots, p+1, q+1.$$

All level-i cycle edges except $(p, p+1)$ are in the spanning cycle; if $|V_i| = 2$, then $p = b_i$, $p+1 = t_i$, and $(p, p+1)$ is in C.

Assume that the only choices for the four vertices force $(q, q+1) = (x, y)$. (This implies that either x or y is the only level-$(i+1)$ vertex that is adjacent to more than one level-i vertex. Thus either $x = b_{i+1}$ or $y = t_{i+1}$. If $x = b_{i+1}$, then every

level-$(i+1)$ vertex is adjacent to t_i. If $y = t_{i+1}$, then every level-$(i+1)$ vertex is adjacent to s_i.) Fix one such choice. Because G is maximal, either $(p, q+1) \in E$ or $(p+1, q) \in E$. First, assume that $(p+1, q) \in E$. Because the choice of q and $q+1$ was forced, we can conclude that $q = b_{i+1}$ and $p+1 = t_i$. Furthermore, the edges (j, q), $s_i \leq j \leq p+1 = t_i$ and $(p+1, j)$, $b_{i+1} = q \leq j \leq t_{i+1}$ are all the leveled edges between V_i and V_{i+1}. (b_{i+1}, t_{i+1}) is an edge in C. Replace it with the path $b_{i+1}, p, \cdots, b_i, t_i, t_{i+1}$, yielding C'. The result is a spanning cycle for $V_i, \cdots, V_m$ satisfying the inductive hypothesis.

Now assume that $(p, q+1) \in E$. We can conclude that $q+1 = t_{i+1}$ and $p = s_i$. Furthermore, the edges $(j, q+1)$, $s_i = p \leq j \leq t_i$ and (p, j), $b_{i+1} \leq j \leq q+1 = t_{i+1}$ are all the level edges between V_i and V_{i+1}. (b_{i+1}, t_{i+1}) is an edge in C. Replace it with the path $b_{i+1}, s_i, \cdots, b_i, t_i, t_{i+1}$, yielding C'. The result is a spanning cycle for $V_i, \cdots, V_m$ satisfying the inductive hypothesis.

By induction, G' has a hamiltonian cycle. The theorem follows. □

Theorem 4.2 cannot be improved, in the sense that there are 1-queue graphs that require two stacks. For example, the complete bipartite graph $K_{2,3}$ is a leveled-planar, hence 1-queue, graph, but it is not outerplanar, hence is not a 1-stack graph. Dually, 1-stack graphs need not be 1-queue graphs, but they never need more than two queues.

THEOREM 4.3. *Any 1-stack graph has a 2-queue layout.*

Proof. Let $G = (V, E)$ be a 1-stack graph having $n \geq 3$ vertices. Then G is outerplanar. We may assume that G is a maximal outerplanar graph. Then G has a unique outerplanar embedding such that all its vertices are on the exterior face, and the boundary of that face is the unique hamiltonian cycle C for G.

Level V as follows. Choose an arbitrary vertex, and label it vertex 1. Proceed in a breadth-first manner from vertex 1. For each vertex $v \in V$, let $\delta(v)$ be the length of a shortest path from vertex 1 to v. Let $m = 1 + \max_{v \in V} \delta(v)$. For $1 \leq i \leq m$, define

$$V_i = \{v \in V \mid \delta(v) = i - 1\}.$$

Then $V_1, \cdots, V_m$ is a partition of V. In each V_i, order the vertices $b_i, \cdots, t_i$ as they are encountered in a counterclockwise traversal of C, beginning at 1. Ordering V level by level, we obtain a linear order $\sigma = 1, 2, \cdots, n$ for V. We must show that σ accommodates an assignment of E to 2 queues.

Let $E_\ell \subset E$ be the edges between consecutive levels. We claim that no two edges in E_ℓ nest with respect to σ. Suppose, to obtain a contradiction, that $(p_2, q_2) \in E_\ell$ nests inside $(p_1, q_1) \in E_\ell$, $p_1 < p_2 < q_2 < q_1$. Then $\delta(p_1) \leq \delta(p_2)$ and $\delta(q_2) \leq \delta(q_1)$. Since $\delta(q_1) = \delta(p_1) + 1$ and $\delta(q_2) = \delta(p_2) + 1$, we must have that $\delta(p_1) = \delta(p_2)$ and $\delta(q_1) = \delta(q_2)$. Then the vertices occur in one of the counterclockwise orders

$$1, p_1, p_2, q_2, q_1, \tag{2}$$

$$1, p_1, q_2, p_2, q_1, \tag{3}$$

$$1, q_2, p_1, q_1, p_2, \tag{4}$$

$$1, q_2, q_1, p_1, p_2. \tag{5}$$

In orders (2) and (3), any path from 1 to p_1 must pass through either p_1 or q_1. Then, however

$$\delta(p_1) + 1 = \delta(q_1) \leq \delta(p_2) = \delta(p_1),$$

a contradiction. A similar contradiction follows for orders (4) and (5). By Theorem 2.3, then, the order σ allows us to lay out the edges in E_ℓ using one queue.

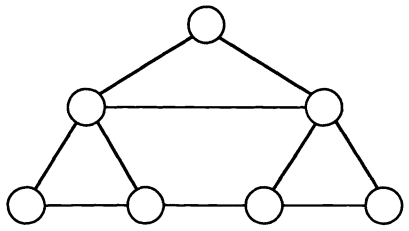


FIG. 4.1. *A 1-stack, 2-queue graph.*

Each edge in $E - E_\ell$ is incident on two vertices in the same level. Clearly, for two edges in $E - E_\ell$ to nest, they must be in the same level. Suppose that there are two edges $(p_1, q_1), (p_2, q_2) \in E - E_\ell$ that nest, so that $p_1, p_2, q_1, q_2 \in V_i$ and $p_1 < p_2 < q_2 < q_1$. Then, however, the counterclockwise order of the vertices is $1, p_1, p_2, q_2, q_1$. Since $\delta(p_1) = \delta(q_1)$ and any path from 1 to p_2 must pass through either p_1 or q_1, we have that $\delta(p_1) + 1 \leq \delta(p_2)$, a contradiction to $\delta(p_1) = \delta(p_2)$. Therefore no two edges of $E - E_\ell$ nest. By Theorem 2.3, then, the order σ allows us to lay out the edges in $E - E_\ell$ using one queue.

We have thus shown that the order σ admits a 2-queue layout of G, as was claimed. □

Theorem 4.3 cannot be improved, in the sense that there are 1-stack graphs that require 2-queues. For example, the graph in Fig. 4.1 is outerplanar, hence 1-stack, but it is not an arched leveled-planar (1-queue) graph [8].

5. Asymptotics of bounded valence graphs. The results in §4 suggest, naturally, the conjecture that every graph admitting a k-queue layout admits an $O(k)$-stack layout, and vice versa. Perhaps the big-O constant is just 2. This conjecture is supported by all the layouts of specific graphs in [8]; refer again to Table 1. Further support for the conjecture comes from considering asymptotic bounds for graph layouts, based on valences. Indeed, the results of Malitz [9], [10] dualize immediately from stack layouts to queue layouts, as we now show.

The observation that twists are obstacles for stack layouts has been exploited to obtain upper and lower bounds on stack number for d-valent graphs [9], [10]. The proofs dualize to queue layouts, with rainbows replacing twists.

The first theorem contains a probabilistic upper bound on the queuenumber of a graph as a function of its valence.

THEOREM 5.1. *Let G be an n-vertex, m-edge graph. Then G has an $O\left(m^{1/2}\right)$-queue layout. In particular, if G has valence d, then G has an $O\left((nd)^{1/2}\right)$-queue layout.*

Proof. Use the same argument as in Theorem 2.2 of [10], except replace completely crossing (twist) with completely nested (rainbow). □

Call G *regular* if all vertices of G have the same degree. The next theorem contains a probabilistic lower bound on the queuenumber of a regular graph of bounded valence that leaves a significant gap with the upper bound of Theorem 5.1. In particular, there are bounded-valence graphs of arbitrarily large queuenumber, though we do not know an example of a sequence of such graphs.

THEOREM 5.2. *Most regular d-valent graphs on n vertices have queuenumber*

$$\Omega\left(\frac{\sqrt{dn}}{n^{1/d}}\right).$$

Proof. Use the same argument as in Theorem 6.1 of [9], except replace completely crossing (twist) with completely nested (rainbow). □

6. An exponential queue/stack tradeoff. The results of the last two sections give scenarios in which queues and stacks are mechanisms of roughly equal power for laying out graphs. The evidence for the conjecture made in §5 is strong. The obvious approach to try to prove the conjecture is to generalize the transformations in Theorems 4.2 and 4.3. However, in both theorems, the transformation from a queue (respectively, stack) layout to a stack (respectively, queue) layout transforms the vertex order in a way that depends on the original queue (respectively, stack) layout. This does not yield the desired generalization, however, because, for a general multiqueue (respectively, multistack) layout, the order transformations are different for each queue (respectively, stack), hence not consistent for all queues (respectively, stacks).

In fact, it is not just the approach that fails. The conjecture is false, at least in one direction! To wit, there is a family of graphs whose stack requirements are exponentially greater than their queue requirements, as we now show.

The vertices of the *ternary n-cube* $TC(n)$ (or, simply, the *ternary hypercube*) comprise all strings of length n over the alphabet $\{0,1,2\}$. The edges of $TC(n)$ connect all triples of vertices of the forms $x0y$, $x1y$, and $x2y$ into a triangle (i.e., a copy of K_3). Hence $TC(n)$ has $N = 3^n$ vertices connected in $n3^{n-1}$ triangles.

The family of ternary n-cubes, $n \geq 1$, requires exponentially more stacks than queues.

THEOREM 6.1. *$TC(n)$ admits a queue layout using $2n$ queues but requires $\Omega(N^{1/9-\epsilon})$ stacks in any stack layout, for any $\epsilon > 0$. $TC(n)$ does admit an $O\left((N \log N)^{1/2}\right)$-stack layout.*

To prove this tradeoff, we need two lemmas.

The first lemma appears in Chung, Leighton, and Rosenberg [3]. The *depth-n sum of triangles graph* $T(n)$ has vertices

$$\{a_i, b_i, c_i | 1 \leq i \leq n\}$$

and edges

$$\{(a_i, b_i), (a_i, c_i), (b_i, c_i) | 1 \leq i \leq n\};$$

i.e., it consists of n disjoint triangles.

LEMMA 6.2 (see [3]). *Let the vertices of $T(n)$ be ordered so that the a_i's all appear in a block to the left of the b_i's, which all appear in a block to the left of the c_i's. Then this layout of $T(n)$ requires at least $n^{1/3}$ stacks.*

The second lemma is the ternary hypercube version of a Boolean hypercube packing lemma due to Chung et al. [2]. The proof in the ternary case is sufficiently different from the binary case that we give it in full.

LEMMA 6.3 (packing lemma for ternary hypercubes). *Let G be an m-node subgraph of the ternary hypercube with average degree $2d$. Then $m = |V(G)| \geq 3^d$. In other words, the number of edges within G is no more than $m \log_3 m$.*

Proof. (All logarithms in this proof are base 3.) We proceed by induction on the dimensionality of the hypercube, the inductive hypothesis being

$$2m \log m \geq \sum_v \deg_G(v),$$

where $\deg_G(v)$ is the degree of the vertex v in the graph G.

The base case being easily verified, let us extend the induction. First, partition the hypercube across some dimension, thereby partitioning G into three subgraphs: G_1 of size $m_1 \geq 1$, G_2 of size $m_2 \geq m_1$, and G_3 of size $m_3 \geq m_2$. (If there is no dimension across which G may be thus partitioned, then G is a subgraph of a *Boolean* hypercube. The result then follows by [2].) These three subgraphs are interconnected as follows. We have $s_{\{1,2\}} \leq m_1$ edges connecting G_1 and G_2, $s_{\{1,3\}} \leq m_1$ edges connecting G_1 and G_3, and $s_{\{2,3\}} \leq m_2$ edges connecting G_2 and G_3.

Our inductive hypothesis allows us to conclude that, for $i = 1, 2, 3$, counting only edges in each G_i,

$$\begin{aligned} 2m_i \log m_i &\geq \sum_v \deg_{G_i}(v) \\ &= \sum_v \deg_G(v) - s_{\{i,j\}} - s_{\{i,k\}}, \end{aligned}$$

where j and k are chosen so that i, j, and k are distinct. Therefore, summing over all nodes of G and applying the bounds on the $s_{\{i,j\}}$'s, we get the inequalities

$$\begin{aligned} \sum_v \deg_G(v) &\leq 2(m_1 \log m_1 + m_2 \log m_2 + m_3 \log m_3) \\ &\quad +2\left(s_{\{1,2\}} + s_{\{1,3\}} + s_{\{2,3\}}\right) \\ &\leq 2(m_1 \log m_1 + m_2 \log m_2 + m_3 \log m_3) \\ &\quad +4m_1 + 2m_2. \end{aligned}$$

Because of these inequalities, our claimed result follows from verifying that the following inequality holds whenever $m_3 \geq m_2 \geq m_1 \geq 1$:

$$\begin{aligned} (m_1 + m_2 + m_3) \log(m_1 + m_2 + m_3) \geq\ & m_1 \log m_1 + m_2 \log m_2 \\ & +m_3 \log m_3 + 2m_1 + m_2. \end{aligned}$$

Let us simplify notation by setting $m_2 = am_1$ and $m_3 = bm_1$, so that $b \geq a \geq 1$. Substituting, we obtain that

$$\begin{aligned} (m_1 + m_2 + m_3) \log(m_1 + m_2 + m_3) = {} & m_1 \log m_1 + m_2 \log m_2 + m_3 \log m_3 \\ & +m_1 \log(1 + a + b) \\ & +am_1 \log\left(\frac{1+a+b}{a}\right) \\ & +bm_1 \log\left(\frac{1+a+b}{b}\right). \end{aligned}$$

Our task thus reduces to verifying that

$$\log(1 + a + b) + a \log\left(\frac{1+a+b}{a}\right) + b \log\left(\frac{1+a+b}{b}\right) \geq 2 + a$$

whenever $b \geq a \geq 1$ This is equivalent (via exponentiation) to verifying that

$$(1+a+b)^{(1+a+b)} \geq 9\,(3a)^a\, b^b.$$

We have equality when $a = b = 1$, so we concentrate on the case where $a = b \geq 1$. The inequality becomes

$$(1+2a)^{1+2a} \geq 9\left(3a^2\right)^a \quad \text{or} \quad (1+2a)\left(\frac{(1+2a)^2}{3a^2}\right)^a \geq 9.$$

Both of the factors on the left-hand side are easily seen to be increasing functions of a. Since there is equality when $a = 1$, the inequality holds for all $a \geq 1$.

It remains to show that, for $a \geq 1$ fixed and $b \geq a$ varying, the inequality holds. Rewrite the inequality as

$$\frac{(1+a+b)^{1+a+b}}{b^b} \geq 9\,(3a)^a\,.$$

The inequality holds when $b = a$ by the preceding paragraph, and the left-hand side is easily seen to be an increasing function of b. This completes the proof. □

We now give the proof of Theorem 6.1.

Proof. We can lay $TC(n)$ out inductively, using $2n$ queues, as follows. Lay out consecutively 3 copies of $TC(n-1)$, using $2(n-1)$ queues, all copies in the same order. Call them copy A, copy B, and copy C, from left to right. Now assign one new queue for the A–C edges and one new queue for the A–B edges and the B–C edges. The easy details are left to the reader.

On the other hand, we claim that any way of linearizing $TC(n)$ must use $\Omega(N^\alpha)$ stacks, for any $\alpha < \frac{1}{9}$. The argument reduces the sum-of-triangles layout from [3] to the current problem. Note that when $TC(n)$ is linearized, there are $nN/3$ *centers* of triangles among the N vertices (if a triangle has vertices u, v, w, and they appear in that order in the linearization, then vertex v is the center). Focus on an integer $K = N^{3\alpha}$, and partition the given arbitrary layout of $TC(n)$ into disjoint windows of size K. Some one window W must contain at least $nK/3$ center vertices (note that, in this count, we allow a single vertex to be the center of many triangles). We want to bound (from above) the number of triangles of $TC(n)$ that have a center in window W and also have another vertex in W. When this happens, say that the "other" vertex (i.e., the noncenter) *spoils* the triangle. We obtain the desired bound by determining the largest possible number of edges that could connect vertices within W (i.e., that do not exit W). By the packing lemma, at most $3\alpha nK$ edges have both endpoints in window W. Each such edge can spoil at most one triangle. Hence W contains at least $(1/3 - 3\alpha)nK$ centers of unspoiled triangles.

Let S be the set of unspoiled triangles with centers in W. Define a graph H with vertex set S in which two triangles (vertices of H) are adjacent exactly when they share a vertex in $TC(n)$. Any triangle shares a vertex with at most $3(n-1)$ other triangles. Therefore H has maximum vertex degree at most $3(n-1)$ and must contain an independent set $S' \subset S$ of cardinality

$$|S'| \geq \frac{|S|}{3(n-1)+1} > \frac{|S|}{3n} \geq \left(\frac{1}{9} - \alpha\right) K.$$

Hence S' consists of at least $(\frac{1}{9} - \alpha)K$ disjoint unspoiled triangles with centers in W. Fix $\alpha < \frac{1}{9}$. The result now follows by Lemma 6.2.

A stack layout of $TC(n)$ using $O((N \log N)^{1/2})$ stacks follows immediately from Theorem 2.2 in [10]. Since the proof of that theorem is nonconstructive, it does not provide an explicit stack layout of $TC(n)$. □

The exponential magnitude of this tradeoff is certainly a surprise. Compare this result to the fact that the queuenumber [8] and stacknumber [3] of the Boolean n-cube are both $\Theta(n)$. A substantial tradeoff in the opposite direction has eluded us. In fact, we do not even have a candidate family of graphs that might be difficult for queues, while easy for stacks, in the same sense that the sum of triangles is difficult for stacks, while easy for queues. We conjecture that there is no such family.

Acknowledgments. The authors thank Sandeep Bhatt, Fan Chung, Sriram Pemmaraju, and Andrew Reibman for helpful conversations. The authors also thank the anonymous referee for a number of suggestions that improved the exposition.

REFERENCES

[1] F. BERNHART AND B. KAINEN, *The book thickness of a graph*, J. Combin. Theory B, 27 (1979), pp. 320–331.

[2] F. R. K. CHUNG, Z. FÜREDI, R. L. GRAHAM, AND P. SEYMOUR, *On induced subgraphs of the cube*, J. Combin. Theory A, 49 (1988), pp. 180–187.

[3] F. R. K. CHUNG, F. T. LEIGHTON, AND A. L. ROSENBERG, *Embedding graphs in books: A layout problem with applications to* VLSI *design*, SIAM J. Alg. Discrete Meth., 8 (1987), pp. 33–58.

[4] P. ERDÖS AND E. SZEKERES, *A combinatorial problem in geometry*, Compositio Math., 2 (1935), pp. 463–470.

[5] S. EVEN AND A. ITAI, *Queues, stacks and graphs*, in Theory of Machines and Computations, Z. Kohavi and A. Paz, eds., Academic Press, New York, 1971, pp. 71–86.

[6] R. GAMES, *Optimal book embeddings of the* FFT, *Benes, and barrel shifter networks*, Algorithmica, 1 (1986), pp. 233–250.

[7] M. R. GAREY, D. S. JOHNSON, G. L. MILLER, AND C. H. PAPADIMITRIOU, *The complexity of coloring circular arcs and chords*, SIAM J. Alg. Discrete Meth., 1 (1980), pp. 216–227.

[8] L. S. HEATH AND A. L. ROSENBERG, *Laying out graphs using queues*, SIAM J. Comput., 21 (1992), to appear.

[9] S. M. MALITZ, *Genus g graphs have pagenumber* $O(\sqrt{g})$, in Proc. 29th Annual IEEE Symposium on Foundations of Computer Science, White Plains, NY, 1988, pp. 458–468.

[10] ———, *Graphs with E edges have pagenumber* $O(\sqrt{E})$, 1989, submitted.

[11] D. J. MUDER, M. L. WEAVER, AND D. B. WEST, *Pagenumber of complete bipartite graphs*, J. Graph Theory, 12 (1988), pp. 469–489.

[12] B. OBRENIĆ, *Embedding de Bruijn and shuffle-exchange graphs in five pages*, in Proc. 3rd Annual ACM Symposium on Parallel Algorithms and Architectures, Hilton Head, SC, 1991, pp. 137–146.

[13] C. H. PAPADIMITRIOU AND M. YANNAKAKIS, *Towards an architecture-independent analysis of parallel algorithms*, in Proc. 20th Annual ACM Symposium on Theory of Computing, Chicago, IL, 1988, pp. 510–513.

[14] A. L. ROSENBERG, *The* DIOGENES *approach to testable fault-tolerant arrays of processors*, IEEE Trans. Comput., C-32 (1983), pp. 902–910.

[15] R. E. TARJAN, *Sorting using networks of queues and stacks*, J. Assoc. Comput. Mach., 19 (1972), pp. 341–346.

[16] M. YANNAKAKIS, *Four pages are necessary and sufficient for planar graphs*, in Proc. 18th Annual ACM Symposium on Theory of Computing, Berkeley, CA, 1986, pp. 104–108.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 413–421, August 1992
©1992 Society for Industrial and Applied Mathematics
010

A COMBINATORIAL APPROACH TO BIORTHOGONAL POLYNOMIALS*

DONGSU KIM†

Abstract. It is proved that biorthogonal polynomials are characterized by the recurrence relations whose coefficients are related in a certain way. On the basis of these recurrence relations for biorthogonal polynomials, biorthogonal polynomials are interpreted as weights of sets of certain paths in the line, and the moments of the linear functional involved as weights of sets of certain paths in the plane. This interpretation is a generalization of Viennot's interpretation of general orthogonal polynomials.

Key words. biorthogonal polynomials, orthogonal polynomials, recurrence relations, weighted paths

AMS(MOS) subject classifications. 05A15, 33A65

1. Introduction.

Let L be a linear functional on the vector space of polynomials in x. Let $w(x)$ be a polynomial in x of degree d, for some positive integer d.

We consider two sets of polynomials, $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$, such that $R_n(x)$ is a polynomial in x of degree n and $S_n(x)$ is a polynomial in $w(x)$ of degree n. (So $S_n(x)$ is a polynomial in x of degree dn.) These two sets of polynomials are said to be biorthogonal with respect to a linear functional L if

$$L(R_m(x)S_n(x))\begin{cases} = 0 & \text{if } m \neq n, \\ \neq 0 & \text{if } m = n. \end{cases} \tag{1.1}$$

DEFINITION 1.1. Polynomials $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$ satisfying (1.1) are called biorthogonal polynomials.

If $w(x) = x$, then biorthogonal polynomials become ordinary orthogonal polynomials.

If the linear functional L is given by the integral with respect to a weight function $p(x)$ over an interval $[a, b]$, the above equation (1.1) can be rewritten as

$$\int_a^b R_m(x)S_n(x)p(x)\,dx\begin{cases} = 0 & \text{if } m \neq n, \\ \neq 0 & \text{if } m = n. \end{cases} \tag{1.2}$$

The biorthogonal polynomials in (1.2) were studied by Didon [Di] and Deruyts [De]. They considered the cases where $w(x) = x^d$ and studied examples for weight functions $x^{\alpha-1}(1-x)^{\beta-1}$, the weight function for the Jacobi polynomials on $(0, 1)$. Deruyts also studied the case in which the weight function was $x^a e^{-x}$ on $(0, \infty)$, the weight function for the Laguerre polynomials. Deruyts showed the existence of $d+2$ term recurrence relations in [De].

More recently, in 1951, Spencer and Fano [SF] introduced the case where $w(x) = x^2$, $p(x) = x^a e^{-x}$ in carrying out calculations involving penetration of gamma rays

*Received by the editors December 30, 1990; accepted for publication June 10, 1991.

†Department of Mathematics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139. Present address, Division of Mathematics, University of Texas at San Antonio, San Antonio, Texas 78249-0601.

through matter. In 1958, Preiser [Pr] asked when biorthogonal polynomials, for $w(x) = x^d$, could be solutions of third-order differential equations. He proved that the case where $w(x) = x^2$ was the only instance.

Konhauser, without knowing the results in [De], studied the general properties of biorthogonal polynomials in [Ko1]. He developed a theory of general biorthogonal polynomials similar to that of orthogonal polynomials. He gave, among other things, necessary and sufficient conditions for the existence of the biorthogonal polynomials with a given weight function, and the existence of recurrence relations. Some of his results were given in [De], for the case where $w(x) = x^d$.

Another direction of development was performed by Rossum [Ro]. Rossum studied (formally) biorthogonal polynomials for the case where $w(x) = x^d$ in (1.1) and gave a sufficient condition for the existence of (formally) biorthogonal polynomials with respect to the linear functional whose moments are given. Using this condition, he showed that the weight functions for the Jacobi polynomials, the Bessel polynomials, and the Laguerre polynomials have a pair of biorthogonal polynomials for every positive integer d.

In the pursuit of a recent trend to combinatorially explain as much as possible, Viennot [Vi1], [Vi2] studied a combinatorial interpretation of general orthogonal polynomials. He gave a model for general orthogonal polynomials on the basis of three-term recurrence relations and proved many properties of general orthogonal polynomials by combinatorial methods.

In this paper, we first prove that biorthogonal polynomials in the sense of (1.1) are characterized by the recurrence relations whose coefficients are related in a certain way. On the basis of these recurrence relations for biorthogonal polynomials, for the case where $w(x) = x^d$, we interpret biorthogonal polynomials as weights of sets of certain paths in the line, and the moments of L as weights of sets of certain paths in the plane. This interpretation is a generalization of Viennot's interpretation of general orthogonal polynomials, since the case where $w(x) = x$ gives Viennot's interpretation of orthogonal polynomials.

2. Recurrence relations. We begin with a known theorem [De], [Ko1], Theorem 2.1, about recurrence relations that are satisfied by any pair of biorthogonal polynomials and prove the characterization theorem (Theorem 2.2) of biorthogonal polynomials in terms of recurrence relations.

THEOREM 2.1. *Let $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$ be biorthogonal polynomials defined as in Definition* 1.1. *Then there exist recurrence relations of the form*

$$w(x)R_n(x) = \sum_{i=n-1}^{n+d} b_{n,i}R_i(x)$$

and

$$w(x)S_n(x) = \sum_{i=n-d}^{n+1} \beta_{n,i}S_i(x),$$

where the coefficients $b_{n,i}$ and $\beta_{n,i}$ are functions of n but not of x.

Proof. Since $w(x)$ is a polynomial of degree d,

$$w(x)R_n(x) = \sum_{i=0}^{n+d} c_{n,i}R_i(x), \quad \text{for some coefficients } c_{n,i}\text{'s.}$$

If we multiply this by $S_{n-i}(x)$, for $i > 1$, and apply L, then biorthogonality gives

$$L(w(x)R_n(x)S_{n-i}(x)) = c_{n,n-i}L(R_{n-i}(x)S_{n-i}(x)).$$

On the other hand, since $w(x)S_{n-i}(x)$ is a polynomial of degree $n-i+1$ in $w(x)$, by biorthogonality, we have that

$$L(w(x)R_n(x)S_{n-i}(x)) = 0.$$

Hence we obtain that $c_{n,n-i} = 0$, for $i > 1$.

By similar arguments, we can get the recurrence relations for $\{S_n(x)\}_{n\geq 0}$. □

In the following theorem, we consider recurrence relations where the coefficients of $R_{n+d}(x)$ and $S_{n+1}(x)$ are 1. In this way, we lose little in generality and gain much in simplicity.

THEOREM 2.2. *Let d be a fixed positive integer and $w(x)$ a fixed polynomial of degree d. Let $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$ be the sequences of polynomials satisfying the following recurrence relations:*

(a) $R_{n+d}(x) = w(x)R_n(x) - \sum_{i=1}^{d+1} a_{i,n+d-i}R_{n+d-i}(x)$, *for all integers $n \geq 0$, and for $0 \leq i < d$, $R_i(x)$ is a polynomial in x of degree i, and $R_{-1}(x) = 0$; and*

(b) $S_{n+1}(x) = w(x)S_n(x) - \sum_{i=1}^{d+1} \alpha_{i,n+1-i}S_{n+1-i}(x)$, *for all integers $n \geq 0$, and $S_{-d}(x) = S_{-d+1}(x) = \cdots = S_{-1}(x) = 0$, and $S_0(x) = 1$.*

Then $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$ are biorthogonal if and only if

$$a_{d+1,n} \neq 0, \quad \text{for all integers } n \geq 0,$$

and, for $k = 1, 2, \cdots, d+1$,

$$\alpha_{k,n} = a_{d-k+1,n+k-1} \prod_{i=n}^{n+k-2} a_{d+1,i}, \quad \text{for all integers } n \geq 0$$

(we assume that $a_{0,n} = 1$). We note that the biorthogonality determines the linear functional uniquely.

Proof. The idea of proof is very simple. We apply recurrence relations and biorthogonality repeatedly to see what conditions are implied and needed.

($\Rightarrow$) Let L be the linear functional with respect to which $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$ are biorthogonal. We may assume that $L(1) = 1$. We want to get relations between a's and α's from the biorthogonality of $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$, using the above recurrence relations. This can be done without much trouble. Refer to [Ki] for details.

($\Leftarrow$) Suppose that

$$\alpha_{k,n} = a_{d-k+1,n+k-1} \prod_{i=n}^{n+k-2} a_{d+1,i}.$$

Define a linear functional L by $L(1) = 1$ and $L(R_n(x)) = 0$ for $n > 0$. We can show that $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$ are biorthogonal with respect to L. Again, refer to [Ki] for details. □

According to this theorem, any sequence $\{R_n(x)\}_{n\geq 0}$ ($\{S_n(x)\}_{n\geq 0}$, respectively) of polynomials satisfying appropriate recurrence relations has a companion sequence $\{S_n(x)\}_{n\geq 0}$ ($\{R_n(x)\}_{n\geq 0}$, respectively) with which it is biorthogonal. The above theorem is the basis of the combinatorial interpretation in the following section. Note that, in Theorem 2.2, if $w(x) = x$, then $R_n(x) = S_n(x)$ and $\{R_n(x)\}_{n\geq 0}$ are orthogonal polynomials [Ch], [Sz].

3. Combinatorial interpretation. In this section, we give a combinatorial interpretation of a pair of biorthogonal polynomials $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$, when $w(x) = x^d$ for a fixed positive integer d. For simplicity of the interpretation, we assume that $R_n(x)$'s, $S_n(x)$'s are monic. So we want to give a combinatorial proof of the following theorem.

THEOREM 3.1. *Let $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$ be the sequences of polynomials satisfying the following recurrence relations:*

(a) $R_{n+d}(x) = x^d R_n(x) - \sum_{i=1}^{d+1} a_{i,n+d-i} R_{n+d-i}(x)$, *for all integers* $n \geq 0$, $R_{-1}(x) = 0$, *and, for* $0 \leq k \leq d-1$,

$$R_k(x) = x^k - \sum_{i=1}^{k} a_{i,k-i} R_{k-i}(x);$$

(b) $S_{n+1}(x) = x^d S_n(x) - \sum_{i=1}^{d+1} \alpha_{i,n+1-i} S_{n+1-i}(x)$, *for all integers* $n \geq 0$, *and*

$$S_{-d}(x) = S_{-d+1}(x) = \cdots = S_{-1}(x) = 0, \text{ and } S_0(x) = 1,$$

where $a_{d+1,n} \neq 0$, *for all integers* $n \geq 0$, *and, for* $1 \leq k \leq d+1$,

$$\alpha_{k,n} = a_{d-k+1,n+k-1} \prod_{i=n}^{n+k-2} a_{d+1,i}, \quad \text{for all integers } n \geq 0$$

(*we assume that* $a_{0,n} = 1$). *Then* $\{R_n(x)\}_{n\geq 0}$ *and* $\{S_n(x)\}_{n\geq 0}$ *are biorthogonal.*

Proof. This is Corollary 3.1, which is proved combinatorially. □

To prove Theorem 3.1 combinatorially, we must have a combinatorial interpretation of $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$ and moments of the linear functional L, which satisfies $L(1) = 1$, and $L(R_n(x)) = 0$ for $n \geq 1$. We define $R_n(x)$ as the weight of the set of certain paths from 0 to n, $S_n(x)$ as the weight of the set of certain paths from 0 to dn and $L(x^n)$ as the weight of the set of certain plane paths from $(0,0)$ to $(n,0)$.

3.1. Interpretation of $R_n(x)$ and $S_n(x)$.

DEFINITION 3.1. Let $\mathcal{R}_n$ be the set of paths from 0 to n with the following weighted segments:

$\overset{-a_{i,k}}{\underset{k \qquad\qquad k+i}{\text{———}}}$ for $k \geq 0, \quad i = 1, 2, \cdots, d+1,$

$\overset{x^d}{\underset{k \qquad\qquad k+d}{\text{———}}}$ for $k \geq 0,$

$\overset{x^i}{\underset{0 \qquad\qquad i}{\text{———}}}$ for $i = 1, 2, \cdots, d-1.$

DEFINITION 3.2. Let $\mathcal{S}_n$ be the set of paths from 0 to dn with the following weighted segments:

$\overset{-\alpha_{i,k}}{\underset{dk \qquad\qquad d(k+i)}{\text{———}}}$ for $k \geq 0, \quad i = 1, 2, \cdots, d+1,$

$\overset{x^d}{\underset{dk \qquad\qquad d(k+1)}{\text{———}}}$ for $k \geq 0.$

DEFINITION 3.3. We define a weight function v by $v(\overset{w}{\text{———}}) = w$, where $\overset{w}{\text{———}}$ is one of the above segments.

Let T be a path in $\mathcal{R}_n$ or $\mathcal{S}_n$. Suppose that T consists of segments $T_1, T_2, \cdots, T_t$ in this order. Since the weight sequence of T, $(v(T_1), v(T_2), \cdots, v(T_t))$, determines T uniquely, we may represent T as a sequence of weights.

DEFINITION 3.4. Let $T = (w_1, w_2, \cdots, w_t)$ be a path in $\mathcal{R}_n$ or $\mathcal{S}_n$. Then $v(T)$, the weight of T, is defined by $v(T) = \prod_{i=1}^{t} w_i$.

We now define $R_n(x)$, $S_n(x)$ as weights of $\mathcal{R}_n$, $\mathcal{S}_n$, respectively.

DEFINITION 3.5. For all $n \geq 0$, $R_n(x) = v(\mathcal{R}_n), S_n(x) = v(\mathcal{S}_n)$.

LEMMA 3.1. *$R_n(x)$, $S_n(x)$ defined in Definition 3.5 satisfy the recurrence relations in Theorem 3.1.*

Proof. The proof is obvious from the definition of $\mathcal{R}_n$ and $\mathcal{S}_n$. □

EXAMPLE 3.1. Let $d = 3$. Then $R = (x^2, -a_{1,2}, x^3, -a_{4,6}, -a_{3,10})$ is a path in $\mathcal{R}_{13}$ and $v(S) = -a_{1,2}a_{4,6}a_{3,10}x^5$. (See Fig. 1.)

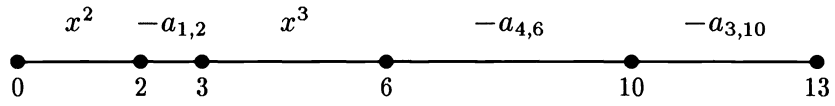


FIG. 1. *A path in $\mathcal{R}_{13}$.*

EXAMPLE 3.2. Let $d = 3$. Then $S = (x^3, x^3, -\alpha_{2,2}, x^3, -\alpha_{4,5}, -\alpha_{1,9}, x^3)$ is a path in $\mathcal{S}_{11}$ and $v(S) = -\alpha_{2,2}\alpha_{4,5}\alpha_{1,9}x^{12}$. (See Fig. 2.)

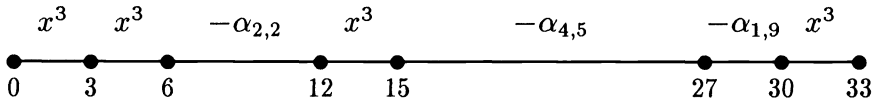


FIG. 2. *A path in $\mathcal{S}_{11}$.*

3.2. Interpretation of moments. We define moments as weights of the sets of moment paths in the plane. Before we define moment paths, we need some preliminary definitions.

Let $\mathcal{P}_{dn}$ be the set of paths in the plane from $(0,0)$ to $(dn,0)$ with the following steps: For nonnegative integers j, k, (see Fig. 3).

(1) Steps from (dj, dk) to $(d(j+1), dk)$ with weight $\alpha_{1,k}$,

(2) Steps from (dj, dk) to $(d(j+1), d(k+1))$ with weight 1,

(3) Steps from $(dj, d(k+1))$ to $(d(j+i-1), dk)$ with weight $\alpha_{i,k}$, for $i = 2, 3, \cdots, d+1$.

Notation. Let $a_{i,n} = 1$ for $n < 0$. Then we can define $\alpha_{k,n}$ for negative integers n as follows:

(a) For $k = 2, 3, \cdots, d$,

$$\alpha_{k,n} = a_{d-k+1,n+k-1} \prod_{i=n}^{n+k-2} a_{d+1,i}, \quad \text{for all integers } n \geq -k+1$$

$$= a_{d-k+1,n+k-1} \prod_{i=0}^{n+k-2} a_{d+1,i}, \quad \text{if } -k+1 \leq n < 0;$$

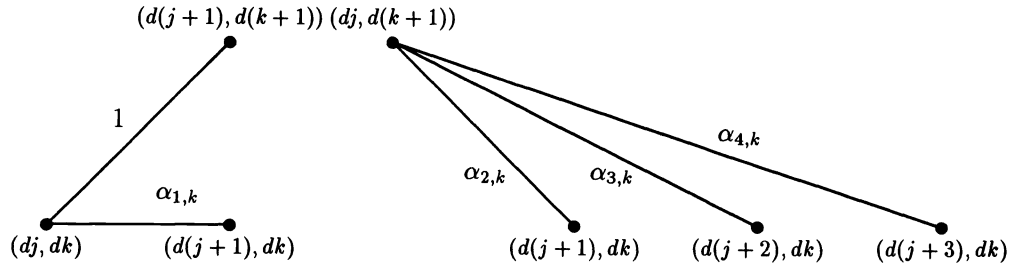


FIG. 3. *Steps for* $\mathcal{P}_{dn}$.

(b) For $k = d + 1$,

$$\alpha_{d+1,n} = \prod_{i=n}^{n+d-1} a_{d+1,i}, \quad \text{for all integers } n \geq -d+1$$
$$= \prod_{i=0}^{n+d-1} a_{d+1,i}, \quad \text{if } -d+1 \leq n < 0.$$

We now want to define $\mathcal{P}_{dn+k}$, the set of paths from $(0,0)$ to $(dn+k,0)$ for $k = 1, 2, \cdots, d-1$. To achieve this goal, we need some steps with length k mod d.

For each ordered pair (i,k) such that $1 < i \leq d$, $-i+1 \leq k < 0$ or $i = d+1$, $-d+1 \leq k < 0$, let the step

$$\underset{(dl,0)}{\overset{\alpha_{i,k}}{\text{———}}}\ \underset{(dl+d(k+i)+k,0)}{}$$

be a horizontal step of length $d(k+i-1)+d+k$. Let P be a path in $\mathcal{P}_{dn}$. Then, by attaching a horizontal $\overset{\alpha_{i,k}}{\text{———}}$ to P at the end, we get a path from $(0,0)$ to $(dn+d(k+i-1)+d+k,0)$. Let $P * \alpha_{i,k}$ denote this new path. We define $\mathcal{P}_{dn+k}$ by

$$\mathcal{P}_{dn+k} = \bigcup_{i=d-k+1}^{d+1} \{P * \alpha_{i,k-d} \mid P \in \mathcal{P}_{dn-d(k-d+i-1)}\}, \quad \text{for } 0 < k < d \text{ and } n \geq 0,$$

where $\mathcal{P}_0$ consists of a point path and $\mathcal{P}_{dn} = \varnothing$, for $n < 0$.

We define a weight function v by $v(\overset{w}{\text{———}}) = w$, where $\overset{w}{\text{———}}$ is one of the steps used in defining paths in $\mathcal{P}_n$.

Let P be a path in $\mathcal{P}_n$. Suppose that P consists of steps $P_1, P_2, \cdots, P_t$ in this order. Since the weight sequence of P, $(v(P_1), v(P_2), \cdots, v(P_t))$ determines P uniquely, we may represent P as a sequence of weights. Let $p_i = v(P_i)$. Then we identify P with $(p_1, p_2, \cdots, p_t)$. It is necessary, however, that we distinguish the underlying path of P from the weight sequence of P.

EXAMPLE 3.3. Let $d = 3$. Then $P = (1, \alpha_{1,1}, 1, \alpha_{1,2}, \alpha_{4,1}, \alpha_{2,0})$ is a path in $\mathcal{P}_{24}$ and $v(P) = \alpha_{1,1}\alpha_{1,2}\alpha_{4,1}\alpha_{2,0}$. $P * \alpha_{3,-1} = (1, \alpha_{1,1}, 1, \alpha_{1,2}, \alpha_{4,1}, \alpha_{2,0}, \alpha_{3,-1})$ is a path in $\mathcal{P}_{29}$ and $v(P * \alpha_{3,-1}) = \alpha_{1,1}\alpha_{1,2}\alpha_{4,1}\alpha_{2,0}\alpha_{3,-1}$. (See Fig. 4.)

Now we want to introduce moment paths. We define the set $\mathcal{M}_n$ of moment paths by $\mathcal{M}_n = \bigcup_{r \geq 0} \mathcal{M}_n^r$, where $\mathcal{M}_n^r$ is defined inductively as follows: We let $\mathcal{M}_n^0 = \mathcal{P}_n$, and assume that $\mathcal{M}_n^{r-1}$ is defined for some $r > 0$. Let $M = (w_1, \cdots, w_i, w_{i+1}, \cdots, w_t)$

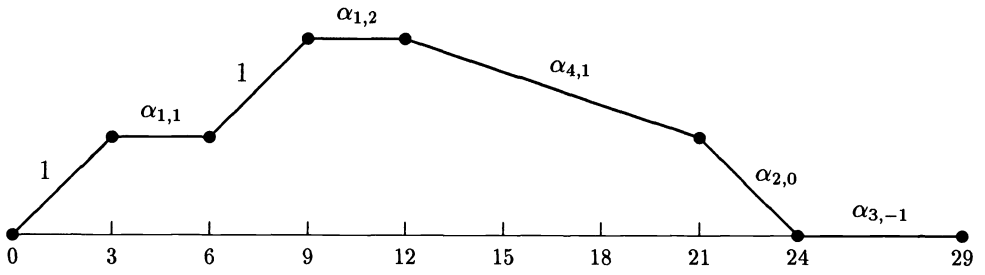


FIG. 4. $P * \alpha_{3,-1}$.

be an element in $\mathcal{M}_n^{r-1}$ such that $w_i \neq 1$, $w_{i+1} \neq 1$. Suppose that $w_i = \alpha_{j_1,k_1}$, $w_{i+1} = \alpha_{j_2,k_2}$ for some j_1, j_2, k_1, k_2 such that $k_1 < k_2 + j_2 - 1$. Then we say that $(w_1, \cdots, w_i^*, w_{i+1}, \cdots, w_t)$ is a variant of M, where $w_i^* = \alpha_{j_1,k_1+1}$. Here we consider this variant as a weighted path whose underlying path is that of M, but the weight of the ith step is changed. We say that a step has an *augmented* weight if its weight is changed. We define $\mathcal{M}_n^r$ to be the set of all the variants of elements in $\mathcal{M}_n^{r-1}$.

Note that (1) for given $M = (w_1, w_2, \cdots, w_t) \in \mathcal{M}_n$, there is a unique path in $\mathcal{P}_n$ that can be the underlying path of M, (2) $\mathcal{M}_n^r = \varnothing$ except for finite number of r's, (3) the step with weight 1 has weight 1 always.

DEFINITION 3.6. Let $M = (w_1, w_2, \cdots, w_t) \in \mathcal{M}_n$. We say that w_i is the weight of the ith step in M. We define $v(M)$, the weight of M, by $v(M) = \prod_{i=1}^t w_i$.

EXAMPLE 3.4. Let $d = 3$. Let $M = (1, \alpha_{1,1}, 1, \alpha_{1,2}, \alpha_{4,1}, \alpha_{2,0}, \alpha_{3,-1})$ be a path in $\mathcal{P}_{29} = \mathcal{M}_{29}^0$. Then there are two variants of M, which belong to $\mathcal{M}_{29}^1$: $M_1 = (1, \alpha_{1,1}, 1, \alpha_{1,3}, \alpha_{4,1}, \alpha_{2,0}, \alpha_{3,-1})$, $M_2 = (1, \alpha_{1,1}, 1, \alpha_{1,2}, \alpha_{4,1}, \alpha_{2,1}, \alpha_{3,-1})$. We find that M_2 has two variants, which belong to $\mathcal{M}_{29}^2$ (see Fig. 5), shown below:

$$(1, \alpha_{1,1}, 1, \alpha_{1,3}, \alpha_{4,1}, \alpha_{2,1}, \alpha_{3,-1}), \qquad (1, \alpha_{1,1}, 1, \alpha_{1,2}, \alpha_{4,2}, \alpha_{2,1}, \alpha_{3,-1}).$$

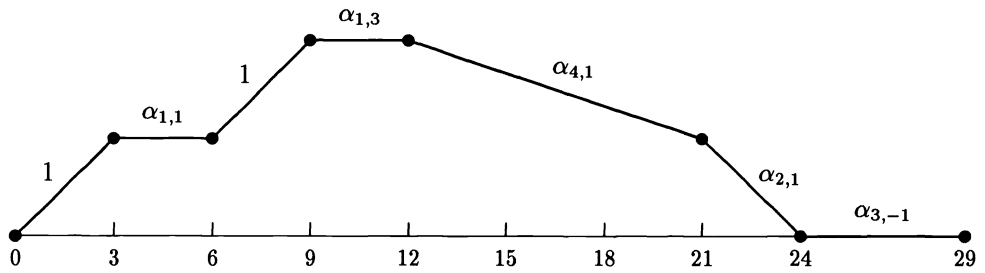


FIG. 5. *One of the two variants of* M_2.

3.3. Interpretation of biorthogonality. We now want to explain the biorthogonality of $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$ combinatorially. We find a combinatorial involution on a weighted set, which is weight-preserving–sign-reversing outside its fixed set. Using this involution, we establish the biorthogonality of $\{R_n(x)\}_{n\geq 0}$, $\{S_n(x)\}_{n\geq 0}$ combinatorially. Our approach is same as Viennot's [Vi1], [Vi2] to orthogonal polynomials, but our involution is more complicated than his. We omit the construction

of the involution here and refer the readers to [Ki] for details. The weighted set that we work with is a set of triples, which is defined as follows.

DEFINITION 3.7. For any triple (l,m,n) of nonnegative integers, let $A_{l,m,n}$ be the set of all triples (M,S,R) such that S is a path in $\mathcal{S}_m$, R is a path in $\mathcal{R}_n$ and M is a moment path in $\mathcal{M}_{dl+p(S)+p(R)}$, where $p(S)$ denotes the power of x in $v(S)$ and $p(R)$ the power of x in $v(R)$. Put a weight v on $A_{l,m,n}$ by

$$v((M,S,R)) = v(M)\bar{v}(S)\bar{v}(R),$$

where $\bar{v}(S) = v(S)|_{x=1}$, $\bar{v}(R) = v(R)|_{x=1}$.

Let L be the linear functional defined by $L(x^n) = v(\mathcal{M}_n)$, for all $n \geq 0$. It is not obvious that this is the linear functional with respect to which $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$ are biorthogonal. This fact is a corollary of Lemma 3.2. We state Lemma 3.2 and its corollary without a proof. We must define some functions to do this.

Let $M = (w_0, w_1, \cdots, w_t)$ denote a moment path in $\mathcal{M}_n$. For each i, $0 \leq i \leq t$, $(w_i, w_{i+1}, \cdots, w_t)$ is called a tail of M. We define a function g on $\mathcal{M}_n$ by

$$g(M) = \begin{cases} \text{the smallest } i \text{ s.t. } w_i \neq 1, & \text{if } M \neq (w_0) \text{ where } w_0 = \alpha_{j,k}, \text{ for some } k < 0, \\ \infty, & \text{otherwise.} \end{cases}$$

We define a function $\bar{h}_1$ on $\mathcal{M}_n$ by

$$\bar{h}_1(M) = \begin{cases} i+k, & \text{if } g(M) < \infty \text{ and } w_{g(M)} = \alpha_{i,k} \text{ for some } i,\ k, \\ \infty, & \text{if } g(M) = \infty. \end{cases}$$

Note that, if $g(M) < \infty$, then $g(M)$ is the number of the consecutive steps with weight 1 in the beginning of M. Finally, we define a function h_2 on $\mathcal{M}_n$ by

$$h_2(M) = \begin{cases} -d+k, \text{ if } w_t = \alpha_{j,k}, \text{ where } j \neq d+1 \text{ and } -j+1 \leq k \leq 0, \\ \text{the largest } i \text{ s.t.}(\alpha_{d+1,i}, \alpha_{d+1,i-d}, \cdots, \alpha_{d+1,k}) \text{ is a tail of } M, \text{ otherwise.} \end{cases}$$

LEMMA 3.2. *For any triple (l,m,n) of nonnegative integers,*

$$L(x^{dl}S_m(x)R_n(x)) = \sum_{(M,S,R)\in F_{l,m,n}} v(M),$$

where $F_{l,m,n}$ is the set of all triples $(M,S,R) \in A_{l,m,n}$ such that $\bar{h}_1(M) > m$, $h_2(M) > n-2d$, $p(S) = dm$ and $p(R) = n$.

Proof. See [Ki] for a complete proof. □

The case where $l = 0$ in Lemma 3.2 gives the biorthogonality of $\{R_n(x)\}_{n\geq 0}$ and $\{S_n(x)\}_{n\geq 0}$.

COROLLARY 3.1. *For any pair (m,n) of nonnegative integers,*

$$L(S_m(x)R_n(x)) \begin{cases} = 0, & \text{if } m \neq n, \\ \neq 0, & \text{if } m = n. \end{cases}$$

Proof. We must show that $F_{l,m,n} = \varnothing$ if $l = 0$. See [Ki] for a complete proof.

REFERENCES

[AV] W. A. Al-Salam and A. Verma, *q-Konhauser polynomials*, Pacific J. Math., 108 (1983), pp. 1–7.

[Ca] L. Carlitz, *A note on certain biorthogonal polynomials*, Pacific J. Math., 24 (1968), pp. 425–430.

[Ch] T. S. Chihara, *An Introduction to Orthogonal Polynomials*, Gordon and Breach, New York, 1978.

[De] J. Deruyts, *Sur une class de Polynômes conjugés*, Memoires coorounés et Memoires de Savant Étrangers Academie Royal des Sciences des letters et des Beaux-Art de Belgigue, Tome 48, 1886.

[Di] M. F. Didon, *Sur certains systemés de polynomes associés*, Ann. Sci. Ecole Norm. Sup. (4), 6 (1869), pp. 111–125.

[Ki] D. Kim, *A combinatorial approach to biorthogonal polynomials*, Ph.D. thesis, Univ. of Minnesota, Duluth, MN, 1989.

[Ko1] J. D. E. Konhauser, *Some properties of biorthogonal polynomials*, J. Math. Anal. Appl., 11 (1965), pp. 242–260.

[Ko2] ———, *Biorthogonal polynomials suggested by the Laguerre polynomials*, Pacific J. Math., 21 (1967), pp. 303–314.

[Pr] S. Preiser, *An investigation of the biorthogonal polynomials derivable from ordinary differential equations of the third order*, J. Math. Anal. Appl., 4 (1962), pp. 38–64.

[Ro] H. Van Rossum, *Formally Biorthogonal Polynomials*, LNM (888), Vol. 888, Springer-Verlag, Berlin, New York, pp. 341–351.

[SF] L. Spencer and U. Fano, *Penetration and diffusion of X-rays. Calculation of spatial distribution by polynomial expansion*, Journal of Research, National Bureau of Standards, 46 (1951), pp. 446–461.

[SJ] H. M. Srivastava and V. K. Jain, *New results involving a certain class of q-orthogonal polynomials*, preprint, 1987.

[Sz] G. Szegö, *Orthogonal Polynomials*, 4th ed., American Mathematical Society, Providence, RI, 1975.

[Vi1] G. Viennot, *Une theorie combinatoire des polynomes orthogonaux generaux*, Notes de conférences données à l'Université du Québec à Montréal, 1983.

[Vi2] ———, *Combinatorial theory for general orthogonal polynomials with extensions and applications*, LNM (1171), Vol. 1171, Springer-Verlag, Berlin, New York, pp. 139–157.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 422–427, August 1992
©1992 Society for Industrial and Applied Mathematics
011

THE PROBLEM OF COMPATIBLE REPRESENTATIVES*

DONALD E. KNUTH† AND ARVIND RAGHUNATHAN‡

Abstract. This paper attaches a name to a natural class of combinatorial problems and points out that the class includes many important special cases. One special case, a simple problem of placing nonoverlapping labels on a rectangular map, is shown to be NP-complete.

Key words. backtracking, coloring, compatibility, independent sets, mapmaking, matching, NP-complete, preclusion, radio communication

AMS(MOS) subject classifications. 68R99, 90C27

1. Introduction.

Many combinatorial tasks can be formulated in the following way: Is there a sequence $(x_1, x_2, \ldots, x_n)$ such that $x_j \in A_j$ for all j, and x_j is compatible with x_k for all $j < k$? Here $A_1, A_2, \ldots, A_n$ are given sets, and "compatibility" is a given relation on $A_1 \cup A_2 \cup \cdots \cup A_n$.

This problem is NP-hard in general. For example, if all sets A_j are the same, and if compatibility is a symmetric, irreflexive relation, a sequence of compatible representatives is nothing but an n-clique in the compatibility graph.

The problem of coloring a graph G with c colors is another NP-hard special case of the general compatibility question. Let A_j be the the set of pairs $\{(j,1), \ldots, (j,c)\}$, and say that (j,a) is compatible with (k,b) if either $a \neq b$ or v_j is not adjacent to v_k in G, where the vertices of G are $\{v_1, \ldots, v_n\}$. Then a sequence of compatible representatives is essentially a c-coloring of G. Therefore the problem is NP-hard for all $c \geq 3$.

On the other hand, the compatibility problem also has important special cases that are efficiently solvable. If the compatibility relation is "$\neq$", then a solution sequence $(x_1, \ldots, x_n)$ is traditionally called a system of distinct representatives [4] [3, Chap. 5], and the problem of finding such systems is well known to be equivalent to bipartite matching. Indeed, if the compatibility relation is the complement of any equivalence relation, a sequence $(x_1, x_2, \ldots, x_n)$ of compatible representatives exists if and only if there is a matching of cardinality n in a bipartite graph on the vertices $\{v_1, \ldots, v_n, c_1, \ldots, c_m\}$, where $\{c_1, \ldots, c_m\}$ are the equivalence classes, and we have the adjacency relation v_j — c_k if and only if A_j contains an element of class c_k.

Another nice special case is equivalent to identifying increasing subsequences of a permutation. Let $\pi_1 \ldots \pi_m$ be a permutation of $\{1, \ldots, m\}$, and let A_j be the set of pairs $\{(j,1), \ldots, (j,m)\}$. Say that (j,a) is compatible with (k,b) if and only if $j < k$ and $\pi_a < \pi_b$. Then a compatible sequence $\big((1,a_1), \ldots, (n,a_n)\big)$ is equivalent to an increasing subsequence $(\pi_{a_1}, \ldots, \pi_{a_n})$ of $\pi_1 \ldots \pi_m$.

The example in the prevous paragraph illustrates that compatibility need not be a symmetric relation. But the sets A_j are pairwise disjoint, and, in that case, we could just as well assume that compatibility is symmetric and reflexive, since our definition

*Received by the editors June 13, 1989; accepted for publication (in revised form) August 7, 1991.

†Computer Science Department, Stanford University, Stanford, California. This author's research was supported in part by National Science Foundation grant CCR-8610181.

‡Division of Computer Science, University of California, Davis, California. This author's research was supported in part by Semiconductor Research Corporation grant SRC-82-11-008.

of compatible representatives makes it immaterial whether elements x_j of A_j and x_k of A_k are compatible, unless we have $j < k$.

There are, however, important special cases in which compatibility is asymmetric. Consider, for example, a scheduling problem in which A_j is a set of tasks that can be done at time j, and where x_j is compatible with x_k only when task x_j does not require the prior completion of x_k.

Cartographers face an interesting case of the general compatibility problem when they attach alphabetic labels to dots on a map. Let A_j represent the possible ways to place the name of city j, and let x_j be compatible with x_k when positions x_j and x_k do not overlap each other or otherwise mislead a potential reader. Then a good map should be a solution to the problem of compatible representatives.

Note that the cartographic problem makes sense even if the sets A_j are infinite. The task of placing disjoint labels is a fairly natural question of combinatorial geometry that does not appear to be a special case of any other well-known problem.

In light of this discussion, it seems worthwhile to add the problem of compatible representatives to the class of "combinatorial problems that deserve a name," and to investigate heuristics and additional special cases that prove to have efficient solutions.

2. Simple special cases. We have noted that the compatibility problem is equivalent to bipartite matching when incompatibility is an equivalence relation. The problem also has a polynomial-time solution when compatibility is transitive. Let $B_1 = A_1$, and for $j > 1$ let

$$B_j = \{\, y \in A_j \mid \exists x \in B_{j-1} \ (x \text{ compatible with } y) \,\}.$$

Then the compatibility problem has a solution if and only if B_n is nonempty. We can decide this in at most $\sum_{j=2}^{n} \|A_{j-1}\| \, \|A_j\|$ steps.

Another noteworthy special case occurs when each set A_j contains at most two elements. Then the compatibility problem is equivalent to an instance of 2SAT: We can assume that $A_j = \{v_j, \overline{v}_j\}$; the clauses are $(\overline{\sigma}_j \vee \overline{\sigma}_k)$ for every pair of literals such that $j < k$ and σ_j is incompatible with σ_k.

In general, if each $\|A_j\| \leq k$ and $k \geq 2$, the problem reduces directly to an instance of kSAT in which each literal occurs positively just once. The literals are (j, a) for $a \in A_j$, and the clauses are

$$\bigvee_{a \in A_j} (j, a), \quad \text{for } 1 \leq j \leq n;$$

$$\overline{(j, a)} \vee \overline{(k, b)}, \quad \text{for } 1 \leq j < k \leq n \text{ and } a \text{ incompatible with } b.$$

Conversely, any instance of kSAT with m clauses reduces to the compatibility problem of finding representatives $(x_1, \ldots, x_m)$, with x_j a member of the jth clause and with two literals compatible if and only if they are not negatives of each other.

The general compatibility problem with finite sets A_j can also be reduced to an independent set problem in a natural way. Consider the graph G with vertices (j, a) for $a \in A_j$, having edges

$$(j, a) \text{ — } (j, b), \quad \text{if } a \neq b;$$
$$(j, a) \text{ — } (k, b), \quad \text{if } j < k \text{ and } a \text{ is incompatible with } b.$$

Then G has an independent set of size n if and only if the compatibility problem has a solution.

Therefore we obtain simple solutions of the compatibility problem when there is a simple solution to the corresponding independent set problem. One such case occurs when compatibility is a symmetric relation that satisfies the following condition: If $i < j < k$ and the elements a_i, a_j, a_k are mutually compatible, then (1) every element of A_i is compatible with either a_j or a_k; (2) every element of A_j is compatible with either a_i or a_k; (3) every element of A_k is compatible with either a_i or a_j; and (4) every element not in $A_i \cup A_j \cup A_k$ is compatible with either a_i, a_j, or a_k. In such a case, the graph G is claw-free, and we can use Minty's algorithm [7] to find a maximum independent set.

Grötschel, Lovász, and Schrijver [2, Chap. 9] have compiled a survey of cases where the independent set problem is known to have a simple solution.

3. Another hard case. A very special case of the general mapmaker's problem, alluded to in the introduction, proves to be NP-complete.

Consider a set of integer points $p_1, \ldots, p_n$ on the plane. We wish to find integer points $x_1, \ldots, x_n$ with the following properties for all $j \neq k$:

$$|x_j - p_j| = 1\,; \quad |x_j - p_k| > 1\,; \quad |x_j - x_k| \geq 2\,.$$

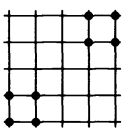
(Motivation: Each x_j is the center of a 2×2 square in which a "label" for point p_j can be placed. The label at x_j should be closer to p_j than to any other point; distinct labels should not overlap.) We call this the MFL problem, for "METAFONT labeling," because it arises in connection with the task of attaching labels to points in diagrams drawn by METAFONT [5, p. 328].

Solutions to the MFL problem can conveniently be represented by showing each point p_j as a heavy dot and drawing an arrow from p_j to x_j for each j; at most four possibilities exist from each of the given points. For example, it is easy to see that a cluster of four adjacent points can be labeled in only two ways:

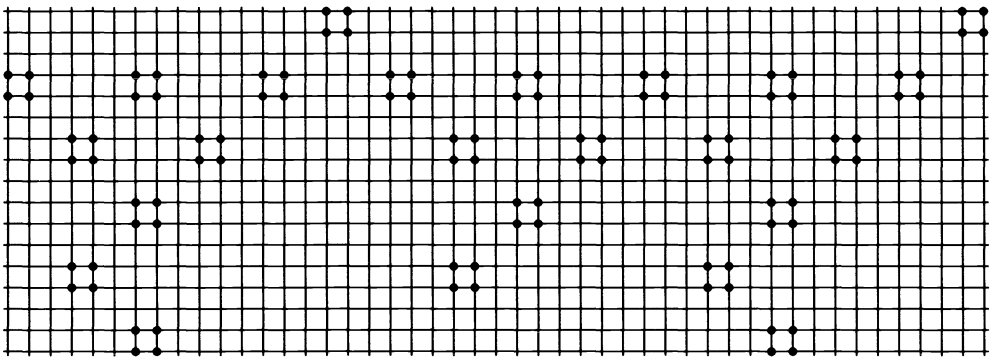
There is no way to attach a label to the middle point in a configuration like

because each of the four positions adjacent to that point is too close to one of the other given points. The MFL problem provides an amusing pastime for people who are sitting in a boring meeting and who happen to have a tablet of graph paper on which to doodle.

The general MFL problem is clearly in NP. To show that it is NP-complete, we observe first that there are only two solutions to the problem

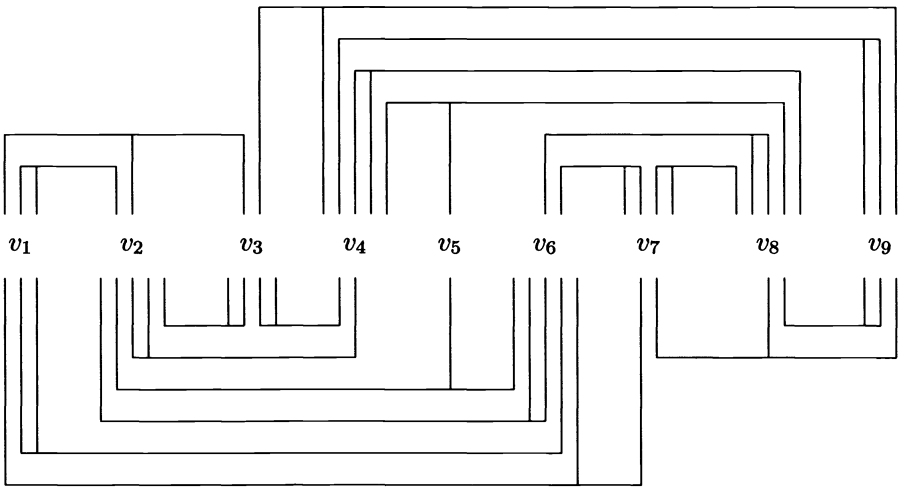


namely, the two solutions for four-point clusters given earlier, using the same orientation in each cluster. Thus we can construct large chainlike tree networks of four-point clusters, for example,



in which there are only two solutions, "positive" and "negative." This construction provides a way to represent the values of Boolean variables in a satisfiability problem.

We can now use Lichtenstein's theorem that planar 3SAT is NP-complete [6]. An instance of planar 3SAT is a set of variables $v_1, \ldots, v_n$ arranged in a straight line, together with a set of three-legged clauses above and below them, where the clauses are properly nested so that none of the legs between clauses and variables cross each other. We can always put the clauses into a rectilinear configuration such as

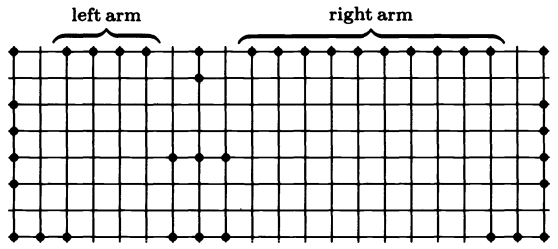


which corresponds to Lichtenstein's "crossover box" [6, Fig. 5].

We construct an instance of MFL from a given instance of planar 3SAT by representing the vertical legs for each variable as chains of four-point clusters; this guarantees that each variable has one of two values, corresponding to the common orientation of all its clusters. We can easily stretch out the diagram so that there is no interference between the variables except at places where three legs of a clause come together in a horizontal segment.

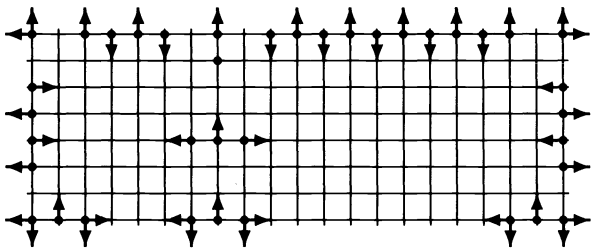
It remains to specify the representation of the clauses. By symmetry, we need only describe the representation that appears above the variables. Each horizontal

section of a comblike clause in the upper portion is represented by a configuration of the form



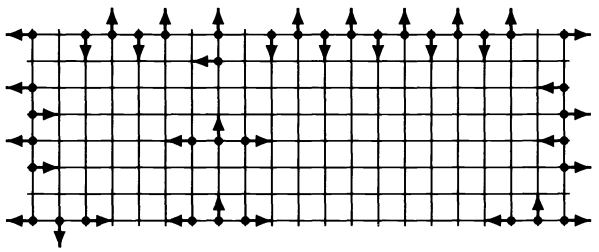
with $6l+4$ dots in the left arm and $6m+4$ dots in the right arm, for some l and m. (The three triples at the bottom connect to clusters that represent variables, as explained below. Those clusters occur at positions that are congruent mod 6; the arms of a comb stretch out so that they reach the variables appropriate to the clause.)

In each group of three dots at the bottom of this construction, the arrow for the middle dot must go either up or down. All three middle arrows cannot go up, because that forces

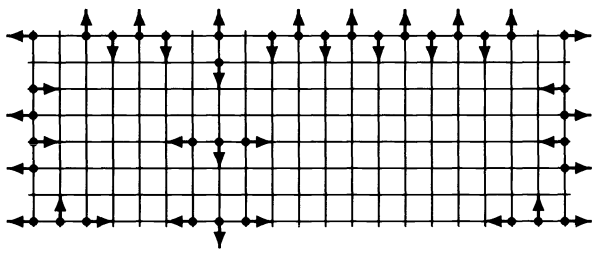


and there is no way to attach an arrow to the middle dot in the second row.

However, there are solutions in which any one of the middle arrows goes down. For example, we can choose



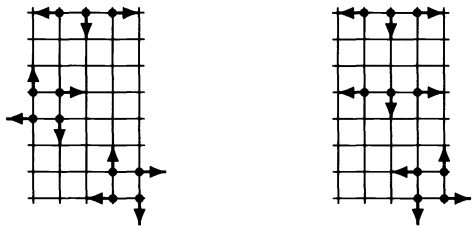
or



and there is a third solution that is essentially a mirror image of the first.

We can place four-point clusters below a row of three dots in such a way that a downward arrow on the top middle dot forces an orientation on the clusters, but an

upward arrow on the top middle dot forces nothing:



By choosing one of these junction configurations for each variable in the clause, depending on whether the variable is negated, we obtain an instance of MFL that has a solution if and only if the given planar clauses are satisfiable.

4. Backtracking. We have now proved that MFL is NP-hard. However, in practice, a solution or proof of nonexistence can often be found quickly by backtracking, using the idea of "preclusion" introduced by Golomb and Baumert [1]. When a trial value x_j is selected from A_j, it precludes all selections of other x_k that are incompatible with it; precluded values can be (temporarily) removed from A_k. The problem of compatible representatives is precisely the abstract general setting that supports this notion of preclusion.

Golomb and Baumert suggest choosing x_j at each stage from a currently smallest set A_j whose representative has not yet been chosen. If we are simply looking for a solution, instead of enumerating all solutions, it would also be worthwhile to select elements that preclude as few others as possible.

For example, if an element of A_j does not preclude any others, we can set x_j equal to that element without loss of generality. If $x \in A_j$ precludes only one element $y \in A_k$ and no others, and if we find no solution when $x_j = x$, then we can set $x_k = y$ without loss of generality.

5. Further work. A recent paper by Simon [8] considers the assignment of channels to transmitters in a radio communication system. This is another case of a compatibility problem, rather like the mapmaker's problem, because nearby transmitters must not broadcast on the same channel. Simon presents a polynomial-time approximation scheme that is guaranteed to find at least a fixed fraction of the optimum number of compatible channels. This suggests that useful approximation schemes for other instances of the general compatibility problem might remain to be found.

REFERENCES

[1] S. Golomb and L. D. Baumert, *Backtrack programming*, J. Assoc. Comput. Mach., 12 (1965), pp. 516–524.

[2] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, New York, 1987.

[3] M. Hall, Jr., *Combinatorial Theory*, Blaisdell, Waltham, MA, 1967; 2nd ed., Wiley–Interscience, New York, 1986.

[4] P. Hall, *On representatives of subsets*, J. London Math. Soc., 10 (1935), pp. 26–30.

[5] D. E. Knuth, *The METAFONTbook*, Addison–Wesley, Reading, MA, 1986.

[6] D. Lichtenstein, *Planar formulæ and their uses*, SIAM J. Comput., 11 (1982), pp. 329–343.

[7] G. Minty, *On maximal independent sets of vertices in claw-free graphs*, J. Combin. Theory B, 28 (1980), pp. 284–304.

[8] H. U. Simon, *Approximation algorithms for channel assignment in cellular radio networks*, in Fundamentals of Computation Theory, Proc. of FCT'89, J. Csirik, J. Demetrovics, and F. Gécseg, eds., Lecture Notes in Computer Science, 380 (1989), pp. 405–415.

SIAM J. DISC. MATH.
Vol. 5, No. 3, pp. 428–450, August 1992
© 1992 Society for Industrial and Applied Mathematics
012

SHORT ENCODINGS OF EVOLVING STRUCTURES*

DANIEL D. SLEATOR†¶, ROBERT E. TARJAN‡¶, AND WILLIAM P. THURSTON§

Abstract. A derivation in a transformational system such as a graph grammar may be redundant in the sense that the exact order of the transformations may not affect the final outcome; all that matters is that each transformation, when applied, is applied to the correct substructure. By taking advantage of this redundancy, we can develop an efficient encoding scheme for such derivations. This encoding scheme has a number of diverse applications. It can be used in efficient enumeration of combinatorial objects or for compact representation of program and data structure transformations. It can also be used to derive lower bounds on lengths of derivations. It is shown, for example, that $\Omega(n \log n)$ applications of the associative and commutative laws are required in the worst case to transform an n-variable expression over a binary associative, commutative operation into some other equivalent expression. Similarly, it is shown that $\Omega(n \log n)$ "diagonal flips" are required in the worst case to transform one n-vertex numbered triangulated planar graph into some other one. Both of these lower bounds have matching upper bounds. An $O(n \log n)$ upper bound for associative, commutative operations was known previously, whereas here an $O(n \log n)$ upper bound for diagonal flips is obtained.

Key words. graph grammar, graph transformations, associativity, commutativity, diagonal flips, triangulations

AMS(MOS) subject classifications. 05, 68

1. Introduction.

The object of this paper is to study succinct representations of derivations in transformational systems. To model transformational systems, we use *graph grammars* [2]. Roughly speaking, a graph grammar consists of a finite set of productions $\{L_i \rightarrow R_i\}$. (Section 2 gives a precise definition of the form of graph grammar that we use.) Each production $L_i \rightarrow R_i$ consists of a connected graph L_i, called the *left side* of the production, and a graph R_i, called the *right side* of the production. A production $L_i \rightarrow R_i$ is *applicable* to a graph G if G contains a subgraph isomorphic to L_i. The production is applied to G by replacing an occurrence of L_i in G by a copy of R_i. (There may be more than one way of applying a production to G, since G may contain more than one copy of the left side.) A *derivation* is a sequence of graphs $G = G_0, G_1, G_2, \cdots, G_m = G'$ such that each G_i is obtained from G_{i-1} by applying one production once. The derivation *transforms* graph G into graph G'. A particular application of a production during a derivation is called an *action*.

Let Γ be a fixed graph grammar, and let G be a fixed starting graph of size n. Consider the collection $R(G, \Gamma, m)$ of all graphs obtainable from G by derivations

* Received by the editors June 12, 1990; accepted for publication (in revised form) August 7, 1991.

† School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213. This research was supported in part by National Science Foundation grant CCR-8658139 and by the Defense Advanced Research Projects Agency, monitored by the Space and Naval Warfare Systems Command under contract N00039-87-C-0251.

‡ Computer Science Department, Princeton University, Princeton, New Jersey 08544 and NEC Research Institute, Princeton, New Jersey 08540. This author's research at Princeton University was partially supported by National Science Foundation grant CCR-8920505 and Office of Naval Research contract N00014-91-J-1463.

§ Department of Mathematics, University of California at Berkeley, Berkeley, California 94720. This author's research was done while at the Department of Mathematics, Princeton University, Princeton, New Jersey 08544 and was partially supported by National Science Foundation grant DMS-8806067.01.

¶ This author's research was partially supported by DIMACS (Center for Discrete Mathematics and Theoretical Computer Science), a National Science Foundation Science and Technology Center, grant NSF-STC88-09648.

of length m or less. Our main result is an efficient method of encoding any graph in $R(G, \Gamma, m)$. To encode any such graph, use $O(m + \log \binom{n}{m})$ bits. (In most cases of interest, $m > n$, and the second term in this bound is zero.) This bound is an improvement by a logarithmic factor over the obvious bound of $O(m \log s)$, where $s \geq n$ is the size of the largest graph occurring in the derivation [9]. This logarithmic improvement is crucial in obtaining the tight lower bounds discussed below.

Our encoding represents an equivalence class of derivations obtained by permuting commutative applications of the productions. The efficiency of such an encoding arises from the fact that there may be many derivations equivalent to any given one, a fact that follows from the localized nature of applications of the production rules. For simplicity, we formulate our result in the setting of labeled undirected graphs; it holds for more general combinatorial structures such as hypergraphs and simplicial complexes, however.

Our result has a number of general and specific applications, both theoretical and practical. Our main theoretical application is in demonstrating the existence of pairs of graphs that are far apart, in the sense that any derivation of one graph of the pair from the other must take many actions. If $N(G)$ is a lower bound on the number of graphs derivable from a graph G of size n, then there is a graph G' such that any derivation of G' from G has length $\Omega(\log N(G) - n)$. This is because our encoding scheme implies that the number of graphs derivable from G' by derivations of length m or less is at most c^{n+m}, for some constant c that depends only on the grammar and not on m and n.

Our first application involves transformations of arithmetic expressions. Consider the collection of fully parenthesized expressions of n variables over an associative, commutative binary operation. A *move* consists of applying either the commutative law (exchanging two subexpressions that are combined by the operation) or the associative law (erasing a pair of matching parentheses to put three expressions at the same level, and adding a new pair of parentheses to alternatively regroup this triple). We show that, given any n-variable expression E, there is an equivalent expression whose distance from E in this metric is $\Omega(n \log n)$. This solves an open problem of Culik and Wood [1], who obtained a matching upper bound. Thus the worst-case distance between two equivalent expressions is $\Theta(n \log n)$. This contrasts with the corresponding bound of $2n - O(1)$ if commutativity is not allowed [6].

As a second application of our lower bound, we consider the collection of numbered triangulations of the plane, transformed by the "flip" operation. This operation removes an edge, thereby creating a quadrilateral face, and replaces it with the other diagonal of the face. A flip is only allowed if it does not create a multiple edge. Our encoding method proves that there exist pairs of n-vertex triangulations that are $\Omega(n \log n)$ flips apart. We show, furthermore, that this bound is tight by giving a method for converting any n-vertex triangulation into any other in $O(n \log n)$ flips. This improves the previous $O(n^2)$ upper bound of Wagner [8].

We envision several other applications of our technique. First, it can be used to efficiently encode graphs or other combinatorial structures that are close to a given one (in the sense of being obtainable by a small number of transformations). Such encodings may be useful in situations that require the representation of multiple versions of a structure, as in program transformation systems and other applications of *persistent* data structures [3]. Second, it provides a way to enumerate graphs of various kinds that are generated by graph grammars or other such transformational systems. By enumerating our encodings rather than enumerating sequences of productions, all

of the desired graphs are generated, but with far fewer redundant copies of isomorphic graphs.

The remainder of this paper consists of five sections. In §2 we give a precise formulation of graph grammars and graph grammar derivations, describe our encoding scheme for derivations, and use this to prove upper bounds on the number of graphs obtainable by short derivations. Section 3 gives several refinements and improvements of our method. Sections 4 and 5 show how the encoding scheme applies to prove our lower bound results for expressions and plane triangulations. Section 6 contains our upper bound on the distance between plane triangulations; it is independent of the rest of the paper.

2. Encoding graph derivations. We are concerned with graphs that are undirected and of degree at most b (a fixed constant independent of n). Each end of each edge is labeled with an integer called an *edge-end label*. The edge-end labels incident on a vertex are distinct and between 1 and b inclusive. It is useful to be able to refer to *half* of an edge. Each such *half-edge* has one end vertex from the original edge, and one edge-end label. We allow the graph to have multiple edges between the same pair of vertices, and even to have self-loops. (It is easy to modify our construction to disallow such structures, although doing so would only weaken our lower bounds.)

A *graph grammar* (usually denoted by Γ) is a finite set of productions $\{L_i \rightarrow_i R_i | i = 1, 2, \cdots\}$. The ith production is comprised of three parts: L_i, the *left side* of the production; R_i, *the right side* of the production; and $\rightarrow_i$, the *correspondence* of the production. The three parts of a production have the following characteristics:

L_i: This is a connected, undirected, edge-end labeled graph, with degree bounded by b. Strictly speaking, L_i is not a graph, because it has a set of half-edges that have only one end vertex. The one end vertex of each half-edge that is attached to a vertex of L_i has an edge-end label.

R_i: This is also a graph with edge-end labels, and half-edges, of which it has the same number as L_i.

$\rightarrow_i$: This is a one-to-one map between the half-edges of L_i and those of R_i.

The production $L_i \rightarrow_i R_i$ applies to a graph G if G contains a set of vertices S such that $G(S)$ (the subgraph induced by S in G) is isomorphic (including edge-end labels) to L_i. The induced subgraph $G(S)$ is most simply defined by retaining half of every edge incident to a vertex in S. The half-edges of $G(S)$ come from the edges of G with one endpoint in S and one not in S. The production is applied by replacing this occurrence of L_i in G by R_i, where each half-edge of R_i is attached to a half-edge of $G - G(S)$ just as the corresponding half-edge of L_i was attached. Sections 3 and 4 give examples of specific graph grammars.

Each vertex occurring in an L_i has a unique *position number* from the set $\{1, 2, \cdots, c\}$, where c is the total number of vertices in all left sides. The position numbers are used to uniquely specify a vertex in a production. The vertices of each R_i are also numbered $1, 2, \cdots$ within each production. These numbers are the *right position numbers*.

A *derivation* is a sequence of graphs $G = G_0, G_1, \cdots, G_m = G'$ such that each G_i is obtained from G_{i-1} by applying one production once. An *action* is a particular application of a production during the derivation. The derivation *transforms* G into G'. The *length* of such a derivation is m, the number of actions in it.

We construct a pair of functions $\text{ENCODE}_{G,\Gamma}$ and $\text{DECODE}_{G,\Gamma}$. The function $\text{ENCODE}_{G,\Gamma}$ takes a derivation D that transforms G into some other graph G' and

returns a string of symbols from the alphabet $\Sigma = \{0, 1, 2, \cdots, c\}$ of length $n + r \cdot m$. Here n is the number of vertices of G, m is the length of the derivation D, c is the number of vertices in left sides of Γ, and r is the number of vertices in the largest right side of Γ. This sequence is called the *encoding* of the derivation. The function $\text{DECODE}_{G,\Gamma}$ takes as input such an encoding and returns the graph G'. That is, $\text{DECODE}_{G,\Gamma}(\text{ENCODE}_{G,\Gamma}(D)) = G'$.

For our purposes, it is useful to think of the process of applying a production as *destroying* vertices (the ones that are matched to the vertices of L_i) and *creating* new and different ones (the ones introduced by R_i). The actions of a derivation D of length m are numbered $1, 2, \cdots, m$ in the order in which they occur. Each vertex that is created during the derivation can be identified uniquely by specifying the number of the action that created it and the position number of the vertex in R_i that created it. This is the *name* of the vertex. The *required vertices* of an action are the vertices that are destroyed by it. An action is said to be *ready* at some time during a derivation if all of its required vertices exist at that time. Readiness implies that the entire copy of L_i that is to be replaced (including all of its edges and half-edges) is present as well.

LEMMA 2.1. *Consider a derivation D that transforms G into G'. If the actions of D are reordered in any way so that each production is ready when it is applied, then the new derivation also transforms G into G'.*

Proof. By induction, it is sufficient to prove that, if a_t and a_{t+1} (two consecutive actions of D) are such that none of the required vertices of action a_{t+1} is created by a_t, then, if these actions are swapped, the resulting derivation also transforms G into G'. Since either order is allowed, we know that those vertices created by a_t are not used by a_{t+1} and those created by a_{t+1} are not used by a_t. It follows that the actions commute, since they do not involve any of the same vertices. □

We can now give an explicit algorithm for computing the encoding of a derivation D. First, the actions of D are numbered, the vertices of the derivation are named, and the required vertices of each action are computed.

Our encoding algorithm assigns to each vertex of the derivation a unique *number*. First, the vertices of G are numbered $\{1, 2, \cdots, n\}$ in an arbitrary order. (The same ordering must be used by the decoding procedure described below.) The remaining vertices are numbered in conjunction with the construction of a *canonical* derivation D', which is a reordering of the actions of D.

The actions of the canonical derivation D' are computed one at a time. At any time, it is easy to determine which actions are ready; these are the ones whose required vertices exist. Let q be the ready action that destroys the lowest-numbered vertex among all ready actions. This action is the one chosen to be the next action of D'. This action is applied to the graph, and the vertices created by it are now numbered consecutively starting after the largest vertex number used thus far. (If several vertices are created by q, they are numbered in the order of the right position numbers in the right side of the production that created them.)

After computing the canonical derivation D', the algorithm proceeds to compute a *label* for every vertex that occurs in the derivation. The label of a vertex v is zero if v is not destroyed by any action in the derivation. Otherwise, it is the position number of the role played by v in the production that destroys it. The desired encoding is a list of, at most, $n + r \cdot m$ labels of all the vertices in increasing order by vertex number.

We can now describe the decoding procedure. This algorithm takes the graph G (with vertex numbers that agree with those of the encoding procedure), the grammar Γ, and the encoding (the list of labels), and determines G'. The procedure works

by constructing the canonical derivation D', from which it is easy to get G'. As in most data compression/decompression methods, the decoding algorithm mimics the behavior of the encoding algorithm step by step.

The crucial fact concerning the labels of the vertices existing at any time during the canonical derivation is that from these it is possible to determine exactly which actions were ready at the corresponding stage of the encoding process. This is accomplished by the following *matching* procedure.

If a vertex v has a nonzero label, the label determines i, the production that eventually destroys v, and also the role v plays in this production. For each such vertex, check its neighborhood to see if it is isomorphic to L_i (including edge-end labels). This check is easy, since we know which vertex of L_i must match v, L_i is connected, and that there are edge-end labels to follow. (Recall that the edge-end labels incident to a vertex are disjoint.) If such a subgraph is found, then the labels of these vertices are checked to see if they match the position numbers of the roles that they are supposed to play in the proposed action. If all of these tests are passed, then the action is ready.

LEMMA 2.2. *The matching procedure determines the ready productions that existed at the corresponding stage of the encoding process.*

Proof. If an action is ready, then the matching procedure certainly finds it because the vertices corresponding to the left side of the ready action will exist and are labeled in a way consistent with all the conditions checked above.

On the other hand, suppose that the above check is satisfied starting from some vertex v. Let i be the production indicated by the label of v, and let S be the set of vertices that form the subgraph isomorphic to L_i. We claim that, in any continuation of this derivation, all vertices of S must be destroyed simultaneously by a single action. Since all these vertices are destroyed by one action, this action must now be ready.

It remains to show that all vertices of S must be destroyed simultaneously. Consider the first action a in some continuation of the derivation that destroys some vertex w of S. Since a is the first action involving the vertices of S, at the moment action a is applied, all of the vertices of S exist (and have the same labels). From the vertex w, the matching algorithm described above constructs the set S. There is no other possible matching pattern involving w. Therefore the action a destroys all the vertices of S simultaneously. □

Now, given that we know the ready productions and the numbering of the vertices of the current graph, it is easy to find q (the next production of D') because it is the ready action that destroys the lowest-numbered vertex. This action is applied to the graph. The vertices created by it are numbered sequentially (as in the encoding procedure) and are labeled as specified by the encoding. This step is repeated to determine all of the productions of D'. The process terminates when there are no ready productions.

The following theorem, which bounds the number of graphs obtainable from a given one as a function of the length of a derivation, is a consequence of our encoding scheme.

THEOREM 2.3. *Let G be a graph of n vertices, Γ be a graph grammar, c be the number of vertices in left sides of Γ, and r be the maximum number of vertices in any right side of a production of Γ. Let $R(G, \Gamma, m)$ be the set of graphs obtainable from G by derivations in Γ of length at most m. Then $|R(G, \Gamma, m)| \leq (c+1)^{n+r \cdot m}$.*

Proof. Encode the derivation using the scheme described above. The length of the encoding is, at most, $n + r \cdot m$ symbols. This encoding can be padded with zeros

so that its length is exactly $n + r \cdot m$. (This does not interfere with the decoding process, since it is self-terminating.) The alphabet is of size $c + 1$, so the number of such encodings is $(c+1)^{n+r \cdot m}$. Each graph reachable by m or fewer actions is the outcome of applying the decoding procedure to one of these encodings. Therefore the number of such graphs is, at most, the number of such encodings. □

3. Generalizations and improvements. This section describes various extensions and improvements to our encoding scheme, most of which are used later in this paper.

3.1. Encoding short derivations. Our encoding scheme can be modified to make it more efficient when the length of the derivation is short compared to the size of the starting graph. In this case, most of the labels of the vertices of the initial graph are zero. The more efficient encoding specifies which vertex labels are nonzero and only includes labels for these in the vertex label list. Let k be the number of vertices that have nonzero label in the initial labeling of G, and let m, n, r, and c be defined as above. Then the size of this encoding (in bits) is

$$\lceil \log_2 n \rceil + \lceil \log_2 \binom{n}{k} \rceil + (k + mr) \lceil \log_2 (c+1) \rceil.$$

The first term is for bits to encode k, and the second term encodes the subset of vertices with nonzero labels.

THEOREM 3.1. *It holds that*

$$\log |R(G, \Gamma, m)| = O(\log \binom{n}{m} + m),$$

where $R(G, \Gamma, m)$ is the number of graphs obtainable by derivations of length at most m in grammar Γ starting from a graph G of n vertices. (If $m > n$, then $\log \binom{n}{m}$ is interpreted as zero.)

Proof. Theorem 2.3 shows that $\log |R(G, \Gamma, m)| = O(n + m)$. If $r \cdot m > \frac{1}{2}n$, then $O(n+m) = O(m) = O(\log \binom{n}{m} + m)$. If $r \cdot m \leq \frac{1}{2}n$, then we use the above encoding scheme. Since each action causes, at most, r vertices of G to have nonzero labels, we know that

$$k \leq r \cdot m \leq \tfrac{1}{2}n.$$

It follows that

$$(k + mr) \lceil \log_2 (c+1) \rceil = O(m)$$

and that

$$\log_2 \binom{n}{k} \leq \log_2 \binom{n}{rm} \leq \log_2 \binom{n}{m}^r = r \log_2 \binom{n}{m}.$$

Finally, we know that $\log_2 n \leq \log_2 \binom{n}{m}$. The theorem follows from these inequalities and the bound on the number of bits used by the efficient encoding scheme. □

3.2. Leaders and followers. The labels on the set of vertices destroyed by an action contain redundant information. For example, each label of this set has sufficient information to determine which production is the one that destroys all of them. There

is a way to eliminate this redundancy and thereby reduce the size of the encoding in most cases.

The new encoding algorithm begins by computing the standard encoding described above. It then applies a map f to each symbol of the encoded string, giving the new encoding. It remains to define the map f.

Let one vertex of each L_i be chosen to be the *leader*, and let all the other vertices be *followers*. For each L_i, choose a spanning tree. (This can be done, since each left side is connected.) For each follower vertex v, let $\mathrm{DIR}(v)$ be the value of the edge-end label of the v end of the first edge on the path (in the spanning tree) from v to the leader of L_i. (In other words, starting from any vertex in L_i, following the $\mathrm{DIR}(\cdot)$ direction repeatedly leads to the leader.)

The map f is defined as follows ($|\Gamma|$ is the number of productions of Γ, and $v(x)$ is the vertex of a left side with position number x):

$$f(x) = \begin{cases} 0 & \text{if } x = 0, \\ \mathrm{i} & \text{if } v(x) \text{ is the leader of } L_i, \\ |\Gamma| + \mathrm{DIR}(v(x)) & \text{if } v(x) \text{ is a follower.} \end{cases}$$

The decoding algorithm must be modified to accommodate this new encoding. The only difference is in the matching step, which is revised as follows. For each vertex v that is a leader, check its neighborhood to see if it is isomorphic to L_i. If such an isomorphic subgraph is found, then the labels of these vertices are checked to see if they are all followers, and, if a directed edge is drawn from each follower w in the direction of $\mathrm{DIR}(w)$ (which is the label of w minus $|\Gamma|$), then the result is a directed spanning tree rooted at v. If all of these tests are passed, then the action is ready.

We now must verify that Lemma 2.2 still holds, that is, that the sets of vertices satisfying the new matching procedure above exactly correspond to the ready actions. The first part of the proof remains easy; any ready action of the original derivation results in a match in the above procedure. On the other hand, a match also indicates that the corresponding action is ready. Let S be the set of matched vertices. Starting from any follower vertex $w \in S$, the entire set S can be constructed uniquely. Similarly, from the leader vertex v of S, the set S can be uniquely constructed. This is a sufficient condition to guarantee that all vertices of S are destroyed simultaneously, which (as shown above) is the condition that we need to prove that the action is ready.

It may be possible to further reduce the alphabet size by making use of the flexibility that exists in choosing which spanning tree to use on each left side. The number of labels can be reduced from $|\Gamma| + b + 1$ to $|\Gamma| + d + 1$, where d is the number of different directions used in the directed spanning trees of the left sides.

The leader-follower technique applies in any situation in which there is a production with more than one vertex on the left side. It may decrease the size of the label alphabet, but it can never increase it. If the technique applies, then it can be used in conjunction with the next technique to further reduce the alphabet size.

3.3. Eliminating the zero label. Suppose that, for any graph occurring in a derivation using Γ, there exists a way of labeling it with nonzero labels, so that no production is ready. Then the zero label can be eliminated. The encoding procedure must be modified slightly to eliminate the zero labels, while the decoding procedure remains the same.

We now describe how the encoding procedure is revised. First, compute the labeling of all the vertices as before. The vertices with zero labels are exactly those that are

eventually in G', the final graph of the derivation. These are called the *terminal* vertices. Compute the labeling of G' with nonzero labels, so that no production is ready. For each terminal vertex, replace its label with that terminal label just computed in G'.

It is easy to see that this works by reviewing the proof of Lemma 2.2. The proof only differs at the point where it is shown that if the labels match the pattern of some left side L_i, then the production i applied to that set of vertices S is ready. The crucial statement is that, if this situation occurs, then all the vertices of S must be destroyed simultaneously. This is still true. All of the vertices cannot be terminal ones, since their labels admit the application of a production. The set cannot be comprised of both nonterminals and terminals because then the nonterminals would never be allowed to change. Therefore all the vertices of S must be nonterminals, and the previous argument shows that the production is ready.

Note that in any situation in which the leader-follower technique applies, we can eliminate the zero label. This is done by labeling all the terminal vertices as followers.

3.4. Tags. It is sometimes useful to carry extra information along during a derivation. (Sections 4 and 5 give examples of this.) To accommodate this, we allow each vertex to have a *tag* associated with it. Each production also supplies an arbitrary function that is used to define the values of the tags of the vertices created in terms of the tags of the vertices destroyed. Because the tags are computed locally (as a function only of tags of the vertices on the left side of the production), the commutativity that we have exploited in constructing our encoding is still present. Therefore our encoding method and theorems apply to tagged graphs without any changes.

4. Expressions over an associative, commutative operation. Let $X = \{x_1, x_2, \cdots, x_n\}$ be a fixed set of variables, let $\oplus$ be a binary operation, and let E_n be the set of fully parenthesized expressions over $\oplus$ in which each variable x_i occurs exactly once. We consider the problem of estimating how many applications of the associative and commutative laws are required to transform any expression in E_n into any other.

To make this problem somewhat more concrete, we restate it as a problem on binary trees. Our binary tree terminology is that of Knuth [5]. Let T_n be the set of full binary trees with n external nodes, numbered $1, 2, \cdots, n$. Any permutation of $1, 2, \cdots, n$ is allowed; thus $|T_n| = n!\binom{2n-2}{n-1}\frac{1}{n} = (n-1)!\binom{2n-2}{n-1}$ [4]. We permit two transformations of a tree $T \in T_n$: a *twist*, in which the left and right subtrees of a specified internal node are exchanged, and a *rotation*, in which an internal node changes places with one of its children while symmetric order in the tree is preserved. (See Fig. 4.1.) The problem is to estimate the minimum number of twists and rotations needed to transform any tree in T_n into any other. We denote this number by R_n.

This problem is equivalent to the expression transformation problem. The isomorphism (also shown in Fig. 4.1) between expressions and trees is the standard one—an external node labeled i corresponds to the expression "x_i"; an internal node corresponds to the expression $(E_\ell \oplus E_r)$, where E_ℓ and E_r are the expressions corresponding to the left and right children of the node. A twist corresponds to the application of the commutative law; a rotation, to an application of the associative law.

Culik and Wood [1] derived an $O(n \log n)$ bound on R_n. We derive a matching $\Omega(n \log n)$ bound. (Culik and Wood actually worked with a slightly different transformational system, but their result applies to our system, and vice versa.)

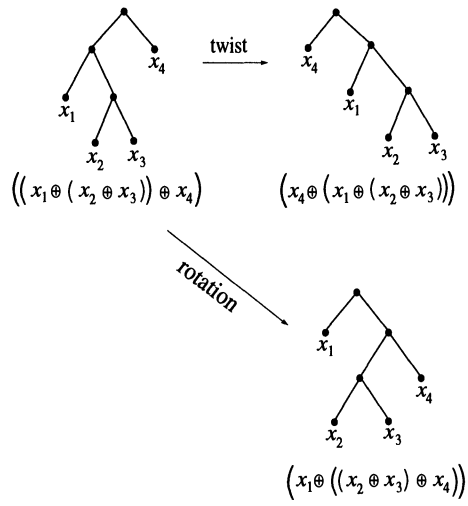


FIG. 4.1. *Illustrating a twist and a rotation.*

These transformations can be represented as productions in a graph grammar. The graphs that we consider differ slightly from the above binary trees. To transform a tree into the corresponding graph, add an extra node of degree one, called the *superroot*, and connect it to the root of the tree. The edge-end labels of the three edges incident to an internal node are 1, 2, and 3, for the edges connecting the node to its left child, right child, and parent, respectively. (The superroot is the parent of the root.) The $n + 1$ edge-ends that are incident on vertices of degree one are irrelevant, since these are never involved in any production. The vertices of degree one are tagged with a name that is carried along during the derivation. Figure 4.2 shows an expression tree and the corresponding graph.

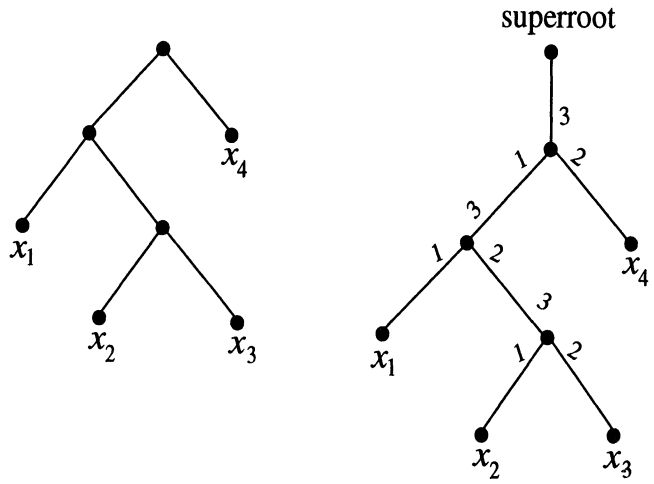


FIG. 4.2. *A tree and its corresponding graph.*

The grammar to represent this process has three productions: one for a twist, one for a left rotation, and one for a right rotation. These productions are shown in Fig. 4.3.

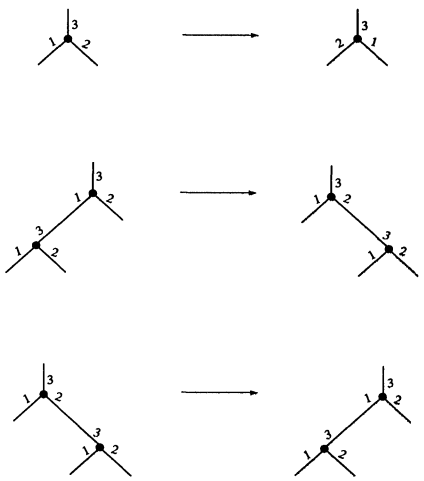


FIG. 4.3. *The productions for a twist and rotations. The correspondence between the half-edges is obtained by pairing the topmost edges and walking clockwise simultaneously around the left and right diagrams.*

From Theorem 2.3, it immediately follows that, starting from a tree with n external nodes, the number of trees reachable in m or fewer twists and rotations is, at most, $6^{2n+2m+1}$. The leader-follower technique can be used to prove a tighter bound. By choosing the upper vertex of the left side of each rotation to be the leader, and the other to be the follower, the label alphabet size is reduced to 5. The zero elimination technique now applies. This reduces the alphabet size to 4, and the bound to $4^{2n+2m+1}$. This can be further improved by specializing the encoding and decoding procedures for this application. We do not need to encode the labels for the n leaves or the superroot because these are not involved in any actions. This improves the bound to 4^{n+2m-1}. The total number of bits needed to encode any tree derivable in m or fewer moves is, at most, $2n + 4m - 2$.

We summarize this result in the following theorem.

THEOREM 4.1. *For any expression E of n variables:*

1. *The number of different arithmetic expressions obtainable by m applications of the commutative and associative laws starting from E is, at most, $2^{2n+4m-2}$;*

2. *There exists an expression E' such that the number of operations required to transform E into E' is $\Omega(n \log n)$.*

Proof. Part 1 follows from the prior discussion. Part 2 follows from the fact that there are $(n-1)!\binom{2n-2}{n-1}$ expressions obtainable starting from E. To obtain all of them in m moves, we must have that

$$2^{2n+4m-2} \geq (n-1)!\binom{2n-2}{n-1} = \Omega(n!),$$

$$2n + 4m - 2 = \Omega(n \log n),$$

$$m = \Omega(n \log n) .$$ □

5. Numbered plane triangulations: A lower bound. A *numbered plane triangulation* (henceforth, just called a triangulation) is an undirected graph embedded in the plane, all of whose faces are triangles and whose vertices are numbered sequentially from 1. We denote by P_n the set of all n-vertex triangulations. A *flip* of an edge in a triangulation is the operation of removing an edge, thereby forming a quadrilateral face, and adding the other diagonal of the face. (See Fig. 5.1.) A flip is allowed only if it does not introduce a multiple edge.

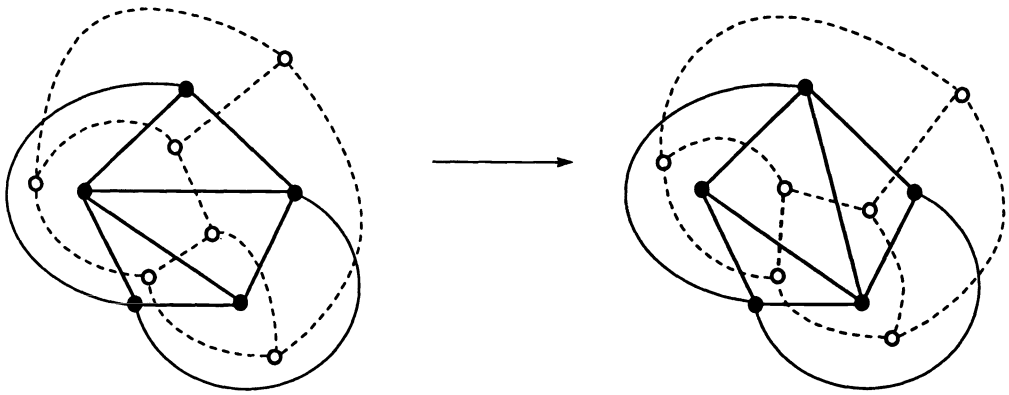


FIG. 5.1. *A flip in a triangulated graph and the corresponding operation in the dual graph.*

Let F_n be the minimum number of flips needed to convert any n-vertex triangulation into any other. We wish to estimate F_n. It is easy to establish that F_n is $O(n^2)$; Wagner [8] gave a construction. We show in §6 that F_n is $O(n \log n)$; in this section, we use our succinct encoding approach to prove that F_n is $\Omega(n \log n)$.

There is no upper bound on the degree of a vertex in a plane triangulation. Therefore, to apply our technique, we work in the space of planar graphs that are dual to plane triangulations. In such a graph, every vertex has degree 3. (Each vertex of the dual graph (a face in the original graph) maintains as a tag the set of vertex numbers of the vertices in the original graph to which it is incident. These tags along with the dual graph are sufficient to reconstruct the original numbered plane triangulation. This observation is required to get a reliable upper bound on the number of reachable numbered plane triangulations.) The edge-end labels of the initial graph are chosen arbitrarily, subject to the constraint that walking one step clockwise around a vertex increases the label by 1 (modulo 3). This ordering of the edge-end labels encodes the embedding of the plane triangulation.

There are several different ways to represent sequences of diagonal flips as derivations in a graph grammar. One way is shown in Fig. 5.2.

This method uses two productions: one for doing the flip, and the other for preparing the edge-end labels to allow the flip. Each flip in the original derivation may correspond to as many as five actions: two to cycle the edge-end labels on one end, two for the other end, and one for the actual flip. A sequence of m flips becomes

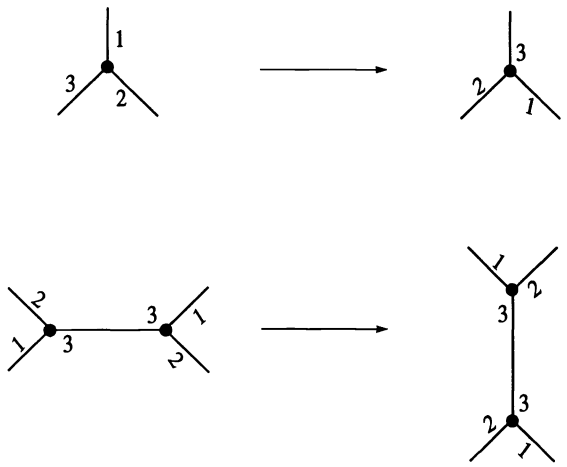


FIG. 5.2. *Two productions for representing flip sequences as graph grammar derivations.*

a sequence of as many as $5m$ actions. A plane triangulation of n vertices has $2n - 4$ faces. Therefore the dual graphs in which the derivations take place have $2n - 4$ vertices. The number of vertices in left sides of productions (c) is 3, and the number of vertices in the largest right side (r) is 2.

We can now apply Theorem 2.3 to bound the number of n node numbered plane triangulations reachable in m flips by $4^{2n+10m-4}$. This implies that, for any triangulation P, at most $4^{2n+10m-4}$ distinct triangulations can be obtained by doing m or fewer flips. Since P_n contains at least $(n-3)!$ triangulations (there are this many different sorted wheels; see §6), there must be at least two triangulations, and indeed many pairs of triangulations, that are $\Omega(n \log n)$ flips apart; that is, $F_n = \Omega(n \log n)$.

The bound on the number of reachable configurations can be tightened significantly by the use of a different graph grammar. This grammar is shown in Fig. 5.3.

Because this grammar includes each of the six ways that the ends of the edge to be flipped can be labeled, there is a one-to-one correspondence between diagonal flips in the plane triangulation and applications of one of the productions to the dual graph. Using the leader-follower trick and eliminating the zero label reduces the number of different labels to 9. Each production creates two new labels, so our improved encoding scheme proves that the number of graphs reachable in m moves is at most $9^{2n+2m-4}$.

Leader vertices can be avoided entirely. An encoding without leaders can be made to work by using the convention that a production involving a pair of adjacent vertices is ready if and only if their labels mutually point at each other. (That is, following the DIR(v) edge from v leads to w, and following the DIR(w) edge from w leads to v.) To verify that the zero label (indicating a terminal vertex) is not necessary, we must show that there exists a labeling of any planar graph of degree 3 with follower labels such that no pair of adjacent vertices point to each other. This can be done as follows. If the graph is a tree, choose a place in the middle of some edge, and make all the vertices point away from this. If the graph has a cycle, choose the labels of the vertices on the cycle to point consistently around it. Now choose a subset of the remaining edges, so that these edges plus the cycle form a subgraph with all of the vertices and exactly one cycle. (This is a spanning tree with one extra edge.) The

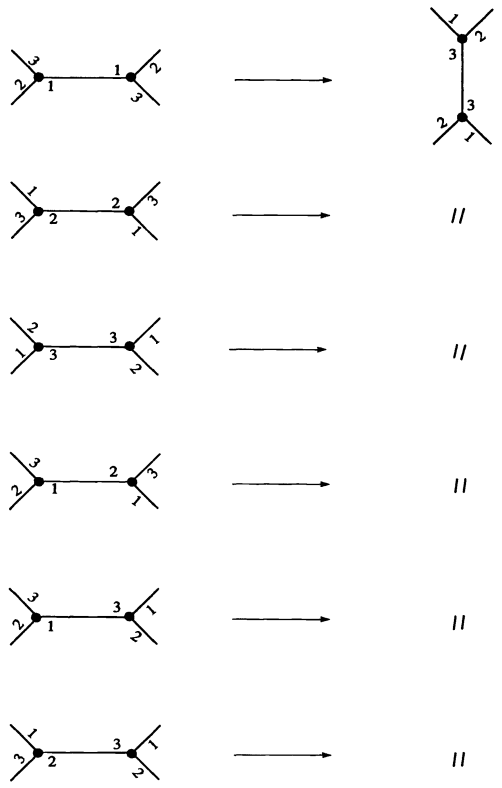


FIG. 5.3. *Six productions give a tighter bound on flip distance.*

follower label on a noncycle vertex points toward the cycle along the path in the tree. This gives the required match-free labeling. This argument bounds the number of reachable configurations by $3^{2n+2m-4}$.

The set of configurations reachable in m or fewer flips is not changed if we do not allow a sequence to make a flip, then immediately make another flip that cancels it out. This observation means that of the nine possible labelings of the pair of vertices resulting after a move, we can restrict our attention to eight of them. This improves the bound to $3^{2n-4}8^m$.

We summarize the results of this section in the following theorem.

THEOREM 5.1. *For any plane triangulation T of n vertices, the following hold:*

1. *The number of different plane triangulations obtainable by m or fewer flips starting from T is, at most, $3^{2n-4}8^m$;*

2. *There exists a plane triangulation T' such that the number of flips required to transform T into T' is $\Omega(n \log n)$.*

6. Numbered plane triangulations: An upper bound.

THEOREM 6.1. *Let G_1 and G_2 be two n-vertex numbered plane triangulations (with no multiple edges). If $n \geq 5$, then there is a sequence of $O(n \log n)$ diagonal flips that transforms G_1 into G_2 in such a way that there are no multiple edges in any*

intermediate state.

Proof. We show that any such triangulation G can be transformed into a particular canonical form called a *sorted wheel* in $O(n \log n)$ diagonal flips. Using this transformation, we can transform G_1 into the sorted wheel, then transform the sorted wheel into G_2 (using the transformation in reverse).

A *wheel* of $n \geq 5$ vertices is a planar graph that has two special vertices called *hubs* and $n-2$ other vertices called *rim vertices*. There is an edge from each hub to each rim vertex (these are the *spokes*). There are $n-2$ other edges in the graph, and these form a simple cycle through all of the rim vertices. There is a unique way of embedding the wheel in a sphere.

A *sorted wheel* of n vertices is a wheel with labeled vertices embedded in the sphere. The hubs are labeled 1 and n, and the vertices of the rim are labeled $2, 3, \cdots, n-1$ in clockwise order when viewed from hub 1.

We first consider the special case of $n = 5$. Any graph G of five vertices satisfying the hypotheses of the theorem is a wheel. We show this by first applying Euler's formula, which implies that G must have six triangular faces and nine edges, and that the sum of the degrees of the vertices is 18. No vertex can have degree two, because then its two neighbors would be connected by two different edges, which violates the assumption that G has no multiple edges. Furthermore, no vertex can have degree greater than four. It follows that the multiset of the degrees of the vertices is $\{3, 3, 4, 4, 4\}$. The three vertices of degree four must be attached to all the other vertices in the graph. This accounts for all of the edges incident on the vertices of degree three, which therefore must not be neighbors. It follows that the graph is a wheel in which the vertices of degree three are the hubs, and the vertices of degree four are the rim.

We finish the proof for $n = 5$ in two stages. First, we show that we can make vertices 1 and 5 the hubs of the wheel. Second, we show that if the resulting structure is not the sorted wheel (it must be its mirror image), then it can be transformed into the sorted wheel.

If vertices 1 and 5 are on the rim, then a diagonal flip of the edge between them makes them the two hubs. If 1 is a hub and 5 is on the rim, then we flip the edge between the other two rim vertices creating a configuration where both 1 and 5 are on the rim, which we handle as above. A similar technique suffices if 5 is a hub and 1 is on the rim.

Figure 6.1 shows how the mirror image of the sorted wheel of five vertices is transformed into the sorted wheel by the application of five diagonal flips.

We are now ready to consider the case where $n \geq 6$. The transformation of the graph G into a sorted wheel is broken up into three phases: constructing a Hamiltonian circuit, transforming the Hamiltonian circuit into a wheel with hubs 1 and n, and sorting the rim of the wheel. These three steps are described in the following three sections.

6.1. Constructing a Hamiltonian circuit. By Tutte's theorem on planar graphs [7] (and by a theorem of Whitney [10]), any 4-connected planar graph has a Hamiltonian circuit. The graph G under consideration is 3-connected, since it is planar, triangulated, and has no multiple edges. Unfortunately, it may not be 4-connected. If it is not 4-connected, then it must have a *separating triangle*; that is, a triangle whose removal separates the graph. We show how to transform the given graph G into one that has no separating triangles by making $O(n)$ diagonal flips. This completes our construction of the Hamiltonian circuit.

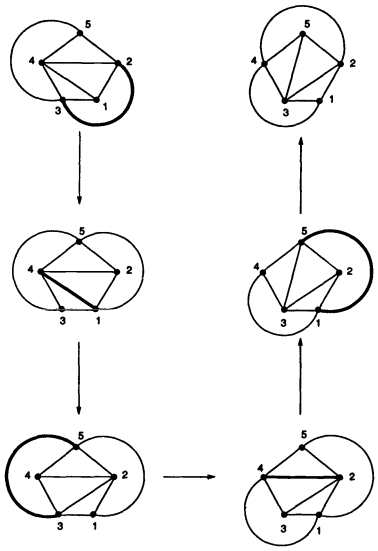


FIG. 6.1

The graph G is given to us embedded on a sphere. We choose a face arbitrarily and map the embedding on the sphere to an embedding on the plane such that the chosen face is infinite. Each separating triangle of G partitions the faces and remaining vertices of G into two components. The *interior* component is the one not containing the infinite face. Let I_1 be the set of faces interior to a separating triangle T_1, and let I_2 be the set of faces interior to a separating triangle T_2. Either I_1 and I_2 are disjoint, or satisfy $I_1 \subset I_2$ or $I_2 \subset I_1$. It follows from these relations that there must always be a set of innermost separating triangles, i.e., those that do not contain another separating triangle in their interior.

Our algorithm for eliminating separating triangles works from innermost separating triangles outward. A diagonal flip operation is applied to an edge of one of the innermost separating triangles. The chosen edge is any one that does not introduce a new separating triangle. We prove below that there always exists such an edge. It follows immediately that this algorithm eliminates all of the separating triangles in $O(n)$ diagonal flips because each flip reduces by at least one the number of edges that are in separating triangles.

It remains to show that there is always an edge of an innermost separating triangle such that if that edge is flipped then no new separating triangle is created. The following case analysis shows this. Consider an innermost separating triangle with vertices a, b, and c. Let d be the vertex inside the triangle such that triangle (a, b, d) is empty. (There must be such a vertex since triangle (a, b, c) is a separating triangle, and there must be something inside of it.) Similarly, there must be a vertex e outside of triangle (a, b, c) such that (a, b, e) is an empty triangle. Figure 6.2 shows the situation.

We assume that flipping edge (a, b) creates a new separating triangle and show that flipping one of the other edges does not create one. We know that the separating triangle that was created by flipping (a, b) must be (d, c, e), and that (d, c) and (c, e) are edges of the original graph. Triangles (a, d, c) and (b, d, c) must be empty;

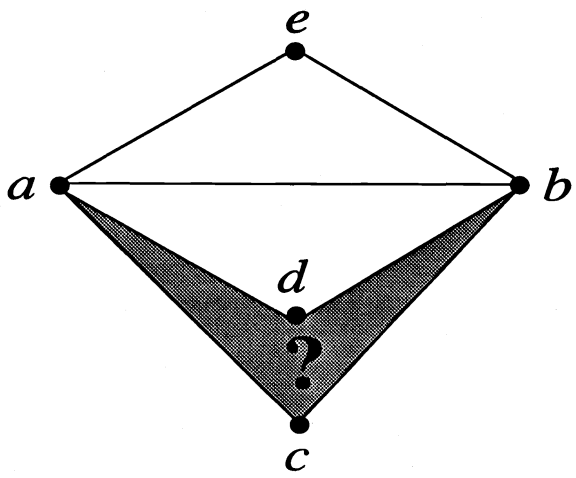


FIG. 6.2

otherwise, (a, b, c) would not be an innermost separating triangle. We now know that the structure of the graph near triangle (a, b, c) is as shown in Fig. 6.3.

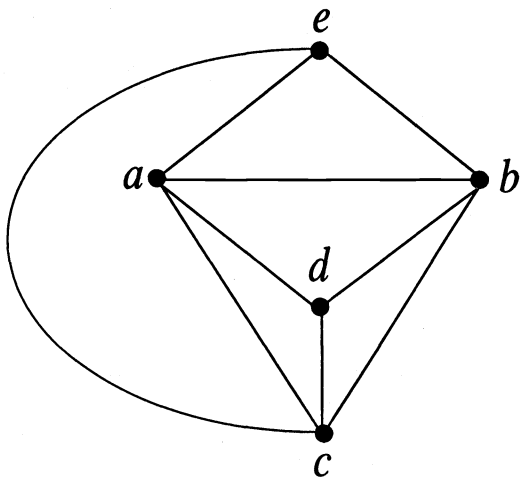


FIG. 6.3

Since the graph has at least six vertices, we know that there must be another vertex f outside of triangle (e, b, c) such that (b, f, c) is an empty triangle. Now it is clear that flipping edge (b, c) cannot create a separating triangle. This completes our construction of a Hamiltonian circuit.

6.2. Transforming the Hamiltonian circuit into a wheel with hubs 1 and n. Given that there is a Hamiltonian circuit, we can regard the graph as consisting of a cycle and two triangulations of an n-gon, one on each side of the cycle. By definition, a triangulation of a polygon has no interior vertices. A *coning* triangulation

of a polygon is one in which all of the interior edges of the polygon are incident to the same vertex. We use several facts about diagonal flips in triangulations of a polygon [6].

FACT 1. *Any triangulation of an n-gon can be transformed into the coning triangulation with all edges incident on a vertex v by making at most $n-2$ diagonal flips, each of which increases the degree of v by one.*

FACT 2. *Any triangulation of an n-gon can be transformed into any other in at most $2n-4$ diagonal flips.*

FACT 3. *In any triangulation of an n-gon, there is a vertex v such that v is incident to only two edges, and those are the boundary edges that connect v to its two neighbors around the polygon.*

Call the two triangulations of the n-gon that comprise the current version of G the *top* triangulation and the *bottom* triangulation. Let v be a vertex such that Fact 3 holds for v in the top triangulation. Now we can apply Fact 1 to vertex v in the bottom triangulation to transform that triangulation into a coning triangulation to vertex v. This process will never introduce multiple edge because all the new edges added to the bottom side of the triangulation are incident to vertex v, which has no edges on the top side. The situation is depicted in Fig. 6.4.

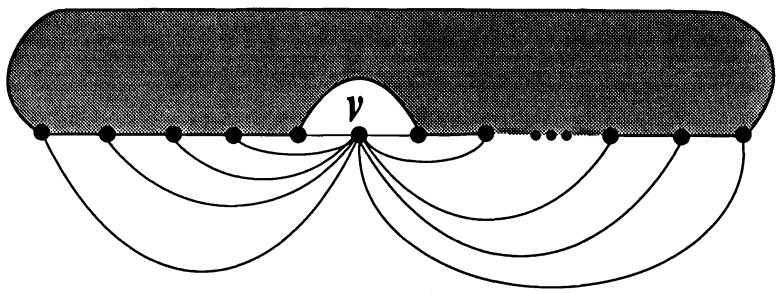


FIG. 6.4

We now change our definition of the top and bottom sides. We view vertex v as belonging to the interior of the bottom side, which is a hub with $n-1$ spokes connecting v to all other vertices. The top side becomes a triangulation of an $(n-1)$-gon. At least one of vertices 1 or n is on this $(n-1)$-gon. Without loss of generality, assume that 1 is on this $(n-1)$-gon. (If 1 is not on this polygon, then the following construction can be fixed by swapping the roles of n and 1.) Now we transform the triangulation of the $(n-1)$-gon (the top side) into a coning triangulation to vertex 1. The result is shown in Fig. 6.5.

We now flip edge $(v,1)$ and move 1 into the top side. The result is a wheel with hubs 1 and v, as shown in Fig. 6.6.

It remains to transform this wheel into one with vertices 1 and n as the hubs. If $v=n$, then we are done; otherwise, it only remains to replace v by n. We begin by flipping any edge around the rim of the wheel, resulting in the situation shown in Fig. 6.7.

Now we retriangulate the bottom $(n-1)$-gon, so that it is a coning to n. Then we flip edge $(n,1)$ and move n into the bottom half to give the triangulation shown

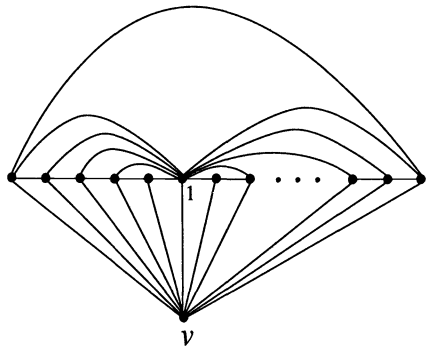


FIG. 6.5

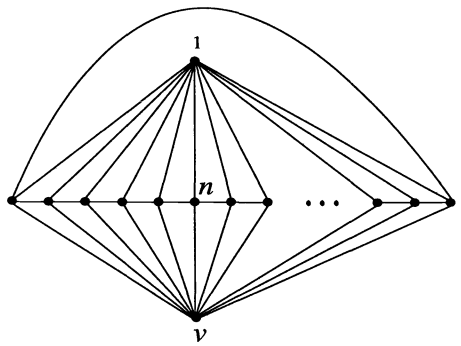


FIG. 6.6

in Fig. 6.8.

This construction works without creating multiple edges, as long as the rim of the wheel is always at least of size four. This is certainly the case, since $n \geq 6$.

6.3. Sorting the rim of the wheel. We first give a sequence of four flips that can exchange any pair of adjacent vertices around the rim of the wheel. Figure 6.9 shows this sequence. This works as long as the number of vertices on the rim is at least 4.

If $6 \leq n \leq 15$, we can use repeated transpositions to sort the wheel. We henceforth assume that $n \geq 16$. A *double wheel* is a wheel with two rims, as shown in Fig. 6.10.

The number of vertices in the *top rim* differs from the number in the *bottom rim* by at most one. Furthermore, all the edges in the region bounded by the two rims cross from one rim to the other.

We now show how to use $O(n)$ diagonal flips to transform a double wheel into a single wheel. We call this transformation a *merge step*. The merge allows us to form

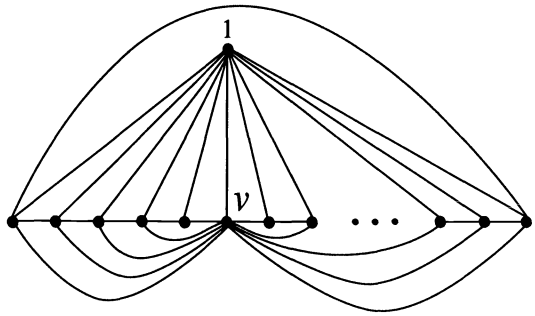


FIG. 6.7

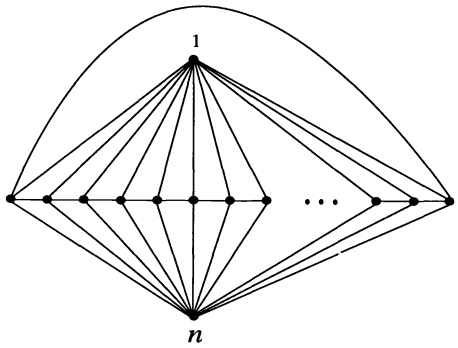


FIG. 6.8

any ordering of the vertices around the rim of the wheel subject to the constraint that the order is consistent with that defined by the orderings on the two rims of the double wheel. That is, if we traverse the rim of the wheel in clockwise order (from the point of view of, say, vertex 1), then the traversal encounters all the vertices that came from the bottom rim (top rim) of the double wheel in the same cyclic order in which they occurred in the bottom rim (top rim) of the double wheel. A more intuitive way to think of this process is to imagine two decks of cards (the double wheel) that are shuffled into one (the rim of the wheel). This is also analogous to the way a merge-sorting algorithm combines two sorted subfiles into a sorted file.

We can also apply the merge step in reverse (an *unmerge*) to split a wheel into a double wheel. A wheel can be sorted by applying a sequence of $\lceil \log_2(n-2) \rceil$ unmerge-merge pairs. (Observe that a merge sort can be implemented using these primitives. Each pass of the sorting algorithm through all of the data corresponds to one of the unmerge-merge pairs.)

It remains to show how to implement the merge step (never introducing multiple edges) in $O(n)$ diagonal flips. An edge (i, j) is called an *amicable edge* if (1) i is on

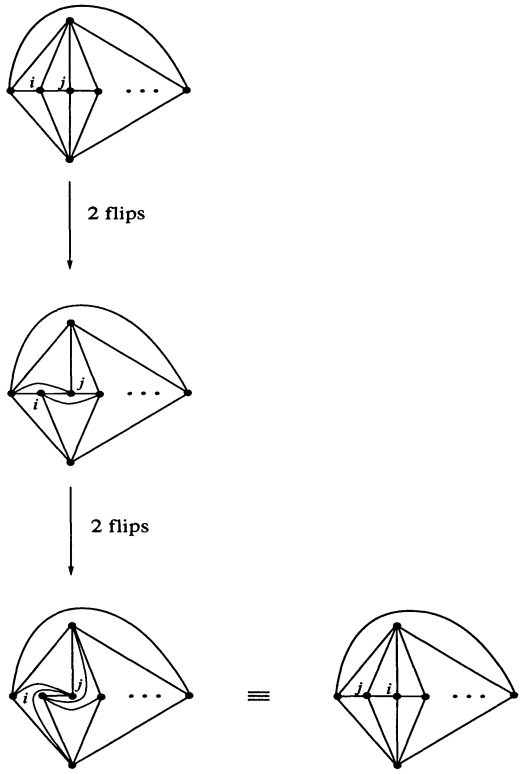


FIG. 6.9

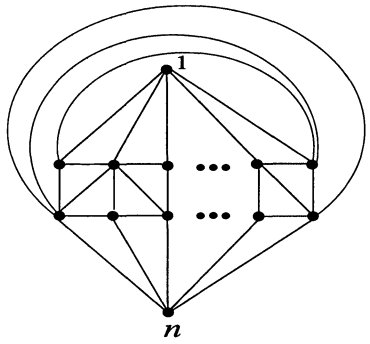


FIG. 6.10

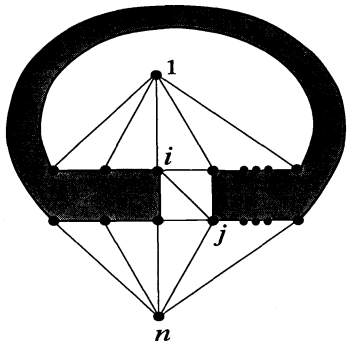


FIG. 6.11

one side of the rim of a double wheel and j is on the other side, (2) the quadrilateral obtained by removing edge (i, j) has one edge on each rim of the wheel and two edges crossing from one side of the rim to the other, and (3) the other vertex of the quadrilateral on the same side of the rim as i is counterclockwise from i (with respect to 1). Figure 6.11 shows an amicable edge (i, j).

In any double wheel there must be an amicable edge. By flipping three edges in the vicinity of an amicable edge, we can create an $(n-2)$-gon such that the edges on the outside of the polygon do not connect any pair of vertices of the polygon. When this operation is applied to the above diagram, the result is shown in Fig. 6.12.

It is now the case that we can apply any algorithm for retriangulating the $(n-2)$-gon between the two rims without fear of creating multiple edges.

We can apply this technique three times to transform a double wheel with one triangulation between the rims into a double wheel with any other triangulation between the two rims. Let (i, j) be an amicable edge of the initial double wheel. Let (k, l) be an amicable edge of the desired final triangulation. (These pairs of vertices may or may not be disjoint.) Let x and x_c be two neighbors on the top rim of the wheel such that x_c is a counterclockwise neighbor of x, and neither x nor x_c is i, or k, or a counterclockwise neighbor of either i or j. Since the length of the rim is at least 7, there must be such a pair. Define y and y_c similarly on the bottom rim.

To transform the triangulation between the rims from any one to any other, we first cut the double rim at amicable edge (i, j), as shown in Fig. 6.12. We then retriangulate the region between the two rims such that (x, y) is an amicable edge. Then we close up the cut of amicable pair (i, j) and open up the one for amicable edge (x, y). We then retriangulate the polygon between the rims, so that the pair (k, l) becomes an amicable edge. We then close up the cut at amicable edge (x, y) and open up the cut at pair (k, l). Now we triangulate the $(n-2)$-gon as specified by the desired final triangulation between the rims. Closing up the cut at amicable pair (k, l) completes the construction of the desired triangulation between the rims. This

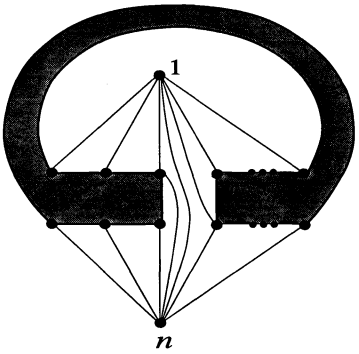


FIG. 6.12

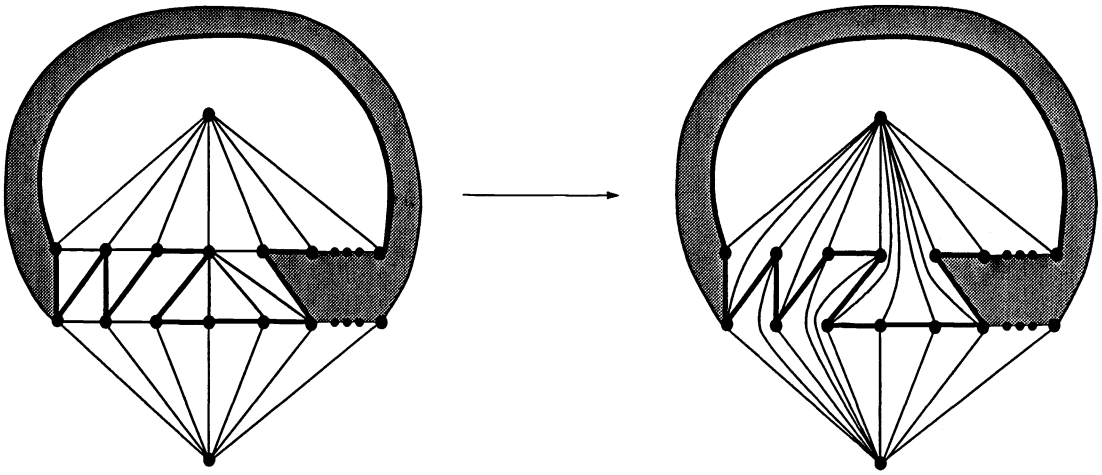


FIG. 6.13

process can never introduce any multiple edges, and it uses $O(n)$ diagonal flips.

As the desired triangulation between the rims, we choose any one such that there is an edge joining each pair of vertices that are adjacent on the rim of the desired wheel. It is easy to see that there must be such a triangulation between the rims.

The last step of the process is to convert such a double wheel into a single wheel. Figure 6.13 illustrates how this is done. The highlighted edges are those of the rim of the wheel.

This step does at most one flip for each vertex of the rim of the wheel and completes the merging process. This also completes the proof of the theorem. □

COROLLARY 6.2. *Let G_1 and G_2 be two n-vertex numbered plane triangulations (possibly with multiple edges). There exists a sequence of $O(n \log n)$ diagonal flips that transforms G_1 into G_2 in such a way that no flip ever creates a self-loop.*

Proof. An easy case analysis proves the result for $n = 4$. We show that multiple edges can be eliminated by flipping them one at a time. This takes only a linear number of flips, since each edge is flipped at most once. The corollary result follows by applying Theorem 6.1 to the graphs obtained by eliminating the multiple edges from G_1 and G_2.

We now prove our claim that if any multiple edge in a plane triangulation is flipped, the number of multiple edges is reduced. Let e_1 and e_2 be a pair of edges between vertices v and w in a plane triangulation. The cycle (e_1, e_2) divides the vertices (except v and w) into two disjoint sets: those on one side of the cycle and those on the other side. Neither of these two sets can be empty, since every face of the graph is a triangle. If either edge e_1 or e_2 is flipped, it is replaced by an edge that connects a vertex on one side of the cycle to one on the other side. Since before the flip there were no edges between vertices in these two sets (they are separated by a cycle), the edge created by the flip cannot be a multiple edge. □

REFERENCES

[1] K. CULIK AND D. WOOD, *A note on some tree similarity measures*, Inform. Process. Lett., 15 (1982), pp. 39–42.

[2] H. EHRIG, M. NAGL, G. ROZENBERG, AND A. ROSENFELD, EDS., *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science, 291, Springer-Verlag, Berlin, New York, 1987.

[3] J. R. DRISCOLL, N. SARNAK, D. D. SLEATOR, AND R. E. TARJAN, *Making data structures persistent*, J. Comput. System Sci., 38 (1989), pp. 86–124.

[4] D. E. KNUTH, *The Art of Computer Programming, Vol* 1: *Fundamental Algorithms*, Addison–Wesley, Reading, MA, 1968.

[5] ———, *The Art of Computer Programming, Vol* 3: *Sorting and Searching*, Addison–Wesley, Reading, MA, 1973.

[6] D. D. SLEATOR, R. E. TARJAN, AND W. P. THURSTON, *Rotation distance, triangulations, and hyperbolic geometry*, J. Amer. Math. Soc., 1 (1988), pp. 647–681.

[7] W. T. TUTTE, *A theorem on planar graphs*, Trans. Amer. Math. Soc., 82 (1956), pp. 99–116.

[8] K. WAGNER, *Bemerkungen zum Vierfarbenproblem*, J. Deutschen Math.–Verein., 46 (1936), pp. 26–32.

[9] E. WELZL, *Encoding graph derivations and implications for the theory of graph grammars*, Lecture Notes in Computer Science, 172, Springer-Verlag, Berlin, New York, 1984, pp. 503–513.

[10] H. WHITNEY, *A theorem on graphs*, Ann. Math., 32 (1931), pp. 378–390.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 451–459, November 1992
© 1992 Society for Industrial and Applied Mathematics

PAIR-SPLITTING SETS IN $AG(m, q)$*

A. BEUTELSPACHER†, D. JUNGNICKEL‡, P. C. VAN OORSCHOT§, AND S. A. VANSTONE¶

Abstract. This paper is concerned with pair-splitting sets in $AG_k(m, q)$, the design obtained from the points and k-flats in $AG(m, q)$. A pair-splitting set is a set of parallel classes $\{R_1, R_2, \cdots, R_s\}$ such that there is no pair of distinct points a, b such that a, b are contained in a common k-flat of each of the s parallel classes. It is easy to prove that a lower bound on s is $\lceil m/(m-k) \rceil$. The main result of this paper is to prove that this lower bound is always achievable for any choice of m, q, and k, where $0 \leqq k \leqq m-1$. The concept of pair-splitting sets arises naturally out of the problem of finding roots in $GF(q^m)$ of a polynomial over $GF(q^m)$. The connection between the two concepts is briefly discussed.

Key words. finite fields, affine geometry, factoring polynomials

AMS(MOS) subject classification. 11T15

1. Introduction.

The purpose of this paper is to study several different aspects of parallelism in $AG(m, q)$, the affine geometry of dimension m and order q. The results obtained have direct application to the problem of finding roots of polynomials over $GF(q^m)$. We describe the connection between these concepts later in this section.

For definitions and results in design theory and geometry that are not explicitly stated, refer to Beth, Jungnickel, and Lenz [3]. Let $D = AG_k(m, q)$ be the design obtained from the points and k-flats in $AG(m, q)$. $AG_k(m, q)$ is a resolvable design with one resolution given by the natural parallelism in the geometry. Let $T = \{R_1, R_2, \cdots, R_s\}$ be a set of s resolution classes of D. We define a transversal of T to be a set $W = \{B_1, B_2, \cdots, B_s\}$, where $B_i \in R_i$, $1 \leqq i \leqq s$, is a k-flat. Let $\Omega(W) = \cap_{B \in W} B$ denote the common intersection of all blocks in the transversal. We call T a *2-splitting* set (or *pair-splitting set*) if, for every transversal W of T, $|\Omega(W)| < 2$. A set of parallel classes T^* is called an *optimal* 2-splitting set if $|T^*| \leqq |T|$ for every 2-splitting set T of D. The following theorem is a special case of a more general result, which was proved in [7].

THEOREM 1.1. *Let $D = AG_k(m, q)$ and let T be an optimal 2-splitting set in D. Then*

$$\frac{m}{m-k} \leqq |T| \leqq \frac{2m}{m-k}.$$

A 2-splitting set that meets the lower bound in Theorem 1.1 is called a *perfect* 2-splitting set. A 2-splitting set T that satisfies $|T| = \lceil m/(m-k) \rceil$ is called a *quasi-perfect* 2-splitting set. The next theorem follows easily.

THEOREM 1.2. *There exists a perfect pair-splitting set in $AG_{m-1}(m, q)$.*

We note that the results of Berlekamp [2] give a method for finding such sets. Additional results on 2-splitting sets in $AG_{m-1}(m, q)$ can be found in [8]. The main result of this paper is the following theorem.

* Received by the editors December 11, 1987; accepted for publication (in revised form) October 18, 1991.

† Siemens AG, München, Germany.

‡ Department of Mathematics, University of Giessen, Giessen, Germany. This author's research was supported by Natural Sciences and Engineering Research Council of Canada grant IS-0367.

§ Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. This author's research was partially supported by Natural Sciences and Engineering Research Council of Canada grant A9258.

¶ Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. This author's research was partially supported by Natural Sciences and Engineering Research Council of Canada grant A9258.

THEOREM 1.3. *There exists a quasi-perfect pair-splitting set in* $AG_k(m, q)$, *and there exists a perfect pair-splitting set if* $m - k$ *divides* m.

The proof of this result is given in §4. In §2 we prove a few basic results on 2-splitting sets and set up some terminology. Section 3 considers the orbit structure of parallel classes of k-flats under the action of a certain group. Section 5 contains a generalization. Section 6 gives a direct construction for optimal pair splitting sets (for all q, m, k).

We conclude this section by giving a brief overview of the connection between 2-splitting sets and finding roots of polynomials over $GF(q^m)$.

Let $f(x)$ be a degree n polynomial over $\mathbb{F} = GF(q^m)$, which has n distinct roots in $\mathbb{F}$. There is a natural correspondence between the elements of $GF(q^m)$ and the points in $AG(m, q)$. With each block B in $AG_k(m, q)$, we form the polynomial

$$B(x) = \prod_{\beta \in B} (x - \beta).$$

Clearly, the degree of $B(x)$ is q^k, and $B(x)$ is an affine polynomial (see Berlekamp [1] for definitions). Let $T = \{R_1, R_2, \cdots, R_s\}$ be a 2-splitting set in $AG_k(m, q)$. It follows from the definition of 2-splitting set that at least one of

$$\gcd(f(x), B(x)), \qquad B \in R_i, \quad 1 \leqq i \leqq s$$

is nontrivial. Repeating the process with each nontrivial factor with degree greater than one eventually produces all of the roots of $f(x)$. Since no pair of roots of $f(x)$ can be contained in each block of a transversal, every pair of roots of $f(x)$ must be separated by some parallel class in the 2-splitting set. The results in Berlekamp [2] and van Oorschot and Vanstone [8] study the preceding procedure for the case where $k = m - 1$. This paper generalizes some of the ideas to arbitrary values of k.

2. Some basic results on pair-splitting sets. We begin by noting that $AG_k(m, q)$ contains

$$\begin{bmatrix} m \\ k \end{bmatrix}_q = \frac{(q^m - 1)(q^{m-1} - 1) \cdots (q^{m-k+1} - 1)}{(q^k - 1)(q^{k-1} - 1) \cdots (q - 1)}$$

parallel classes, a vast number. We show that a quasi-perfect pair-splitting set can always be selected from any orbit of parallel classes under the action of a suitable group, which narrows the number of parallel classes to be considered to at most $(q^m - 1)/(q - 1)$. To accomplish this, we must first introduce some notation.

Since any two m-dimensional vector spaces over $GF(q)$ are isomorphic, we can identify the underlying vector space of $AG(m, q)$ with the field $\mathbb{F} = GF(q^m)$. Then each $\alpha \in \mathbb{F}^*$ induces a collineation $\bar{\alpha}$ of $AG(m, q)$ by defining $\bar{\alpha}(x) = \alpha x$. We usually write α instead of $\bar{\alpha}$, not distinguishing between the field element α and the induced *multiplicative automorphism* $\bar{\alpha}$. Thus all multiplicative automorphisms of $AG(m, q)$ form a group G isomorphic to $\mathbb{F}^*$, and thus to the cyclic group of order $q^m - 1$. We call each orbit of parallel classes of k-flats under the action of G a *multiplicative orbit* of parallel classes or, more simply, a *k-orbit*. Our main result is that any k-orbit (for all k, m, and q) contains a quasi-perfect pair-splitting set of $AG_k(m, q)$. We note in passing that the use of the group G is a common tool in finite geometry (e.g., proving Singer's theorem [6] (see also [3] and Hirschfeld [4]) or its affine analogue (see Jungnickel [5])).

We now characterize the pair-splitting sets contained in multiplicative orbits; these are referred to as *multiplicative* pair-splitting sets.

LEMMA 2.1. *Let $\mathcal{B}$ be the k-orbit determined by a k-dimensional linear subspace V of $\mathbb{F} = GF(q^m)$ over $GF(q)$. Moreover, let $\gamma_1, \gamma_2, \cdots, \gamma_s$ be distinct elements in $\mathbb{F}^*$ and let $\beta_i = \gamma_i^{-1}$, $1 \leqq i \leqq s$. The parallel classes determined by $V\gamma_1, V\gamma_2, \cdots, V\gamma_s$ form a pair-splitting set of $AG_k(m, q)$ if and only if the subspace $U = \langle \beta_1, \beta_2, \cdots, \beta_s \rangle$ is not contained in any of the k-flats in $\mathcal{B}$.*

Proof. Assume first that $V\gamma_1, V\gamma_2, \cdots, V\gamma_s$ determines a pair-splitting set. Then

$$\bigcap_{i=1}^{s} V\gamma_i = \{0\}. \tag{1}$$

Assume, if possible, that $U = \langle \beta_1, \beta_2, \cdots, \beta_s \rangle$ is contained in $V\delta$ for some $\delta \in \mathbb{F}^*$. Then $U\delta^{-1} \leqq V$, and thus $\beta_i\delta^{-1} \in V$, $1 \leqq i \leqq s$, or, equivalently, $\delta^{-1} \in V\gamma_i$, $1 \leqq i \leqq s$, which contradicts (1). Therefore U is not contained in any k-flat in $\mathcal{B}$.

Conversely, assume that U is not contained in any k-flat in $\mathcal{B}$. If $V\gamma_1, V\gamma_2, \cdots, V\gamma_s$ do not determine a pair-splitting set, then there exist $a, b \in \mathbb{F}$ $(a \neq b)$, which are in the same coset of $V\gamma_i$ for all i, $1 \leqq i \leqq s$. Thus $0 \neq a - b \in \bigcap_{i=1}^{s} V\gamma_i$, i.e., $(a - b)\gamma_i^{-1} = (a - b)\beta_i \in V$ for $1 \leqq i \leqq s$. However, $(a - b)U = \langle (a - b)\beta_1, (a - b)\beta_2, \cdots, (a - b)\beta_s \rangle \subset V$, which gives the contradiction $U \subset V(a - b)^{-1}$. This completes the proof. □

It should be clear that the situation of Lemma 2.1 arises if and only if none of the flats in the multiplicative orbit of U are contained in any k-flat in $\mathcal{B}$. This leads us to the following terminology. We say that a d-orbit $\mathcal{D}$ is *skew* to a k-orbit $\mathcal{B}$ if any only if no d-flat in $\mathcal{D}$ is contained in any k-flat in $\mathcal{B}$. Lemma 2.1 immediately implies the following result.

THEOREM 2.1. *Let $\mathcal{B}$ be a k-orbit in $AG(m, q)$. Then the smallest size s of a pair-splitting set contained in $\mathcal{B}$ equals the smallest dimension d of a d-orbit $\mathcal{D}$, which is skew to $\mathcal{B}$.*

Theorems 1.1 and 2.1 give the following corollary.

COROLLARY 2.1. *Let $\mathcal{B}$ be any k-orbit of $AG(m, q)$. Then the smallest dimension d for which a d-orbit skew to $\mathcal{B}$ exists satisfies the inequality*

$$d \geqq \left\lceil \frac{m}{m-k} \right\rceil.$$

As already mentioned, it is our goal to show that we always have equality in Corollary 2.1. Before doing so, we briefly study the k-orbits of G, as this seems not to be in the literature.

3. Multiplicative orbits in $AG(m, q)$. In this section we want to consider the possible sizes of k-orbits in $AG(m, q)$. We recall the following simple but important lemma.

LEMMA 3.1. *Let m, k, and q be positive integers. Then $(q^m - 1, q^k - 1) = q^d - 1$ with $d = (m, k)$. (Here (x, y) denotes the greatest common divisor of x and y.)*

As in the previous section, let $\mathbb{F} = GF(q^m)$ and let G be the group of multiplicative automorphisms of $AG(m, q)$ induced by $\mathbb{F}^*$. Clearly, $\gamma \in \mathbb{F}^*$ fixes a parallel class of k-flats in $AG(m, q)$ if and only if γ fixes the linear k-space in this parallel class. Every $\gamma \in GF(q)^*$, however, fixes every linear subspace, and thus the stabilizer of a parallel class always contains a (cyclic) group of order $q - 1$; so the largest conceivable orbit size is

$$(q^m - 1)/(q - 1) = q^{m-1} + q^{m-2} + \cdots + q + 1.$$

We call a k-orbit *regular* if it has this maximal size. We now give a construction that produces orbits of smaller size in the case where $(k, m) > 1$.

CONSTRUCTION 3.1. *Let $c \neq 1$ be a common divisor of k and m. Then $\mathbb{F} = GF(q^m)$ contains the field $GF(q^c)$, and thus $\mathbb{F}: GF(q^c)$ determines the affine space $AG(m/c, q^c)$, which is naturally contained in $AG(m, q)$ (since any i-flat of $AG(m/c, q^c)$ is also a ci-flat of $AG(m, q)$). In particular, choose a linear k/c-space V in $\mathbb{F}$: $GF(q^c)$; clearly, each $\gamma \in GF(q^c)^*$ fixes V, and thus V determines a k-orbit of $AG(m, q)$, which has a stabilizer of order divisible by $q^c - 1$ and thus has size $\leqq (q^m - 1)/(q^c - 1)$.*

LEMMA 3.2. *Let $\mathcal{B}$ be any k-orbit of $AG(m, q)$. Then the stabilizer of $\mathcal{B}$ in G has order $q^c - 1$ for some common divisor c of k and m; moreover, $\mathcal{B}$ is the orbit of a k/c space in $AG(m/c, q^c)$, as in Construction 3.1.*

Proof. Let H be the stabilizer of $\mathcal{B}$. Then H is the stabilizer of any linear subspace V contained in $\mathcal{B}$. Since G acts regularly on $\mathbb{F}^*$, we see that H acts semiregularly on both $\mathbb{F}^*$ and $V \setminus \{0\}$. Thus $|H|$ is a common divisor of $q^m - 1$ and $q^k - 1$, i.e., a divisor of $q^d - 1$ with $d = (m, k)$ by Lemma 3.1. Now consider $\delta, \gamma \in \mathbb{F}^* \cap H$. Since

$$V\delta = V \quad \text{and} \quad V\gamma = V,$$

then $V(\delta + \gamma) = V\delta + V\gamma = V + V = V$, provided that $\delta + \gamma \neq 0$. Similarly, $V(\delta\gamma) = V$, and so $H \cup \{0\}$ is closed under sums and products and so is a subfield of $GF(q^m)$. It now follows that $|H| = q^c - 1$ for some positive integer c and that c divides d. Finally, since the cyclic group $\mathbb{F}^*$ contains a unique subgroup of order $q^c - 1$, i.e., $GF(q^c)^*$, we see that V is invariant under $GF(q^c)^*$ and thus is, in fact, a k/c-space in $AG(m/c, q^c)$. This completes the proof. □

THEOREM 3.1. *The size of a smallest k-orbit in $AG(m, q)$ is $(q^m - 1)/(q^d - 1)$, where $d = (m, k)$. (We call such orbits* minimal.*) There exist k-orbits of size $(q^m - 1)/(q^c - 1)$ for every common divisor c of m and k, and these are the only orbit sizes.*

Proof. By Construction 3.1, there exist k-orbits with a stabilizer in G of order divisible by $q^c - 1$. This is the precise order of the stabilizer if and only if the orbit belongs to a k/c-space of $AG(m/c, q^c)$ that is not a k/cf-space of $AG(m/cf, q^{cf})$ for any common divisor $f \neq 1$ of k/c and m/c. To simplify notation, put $l = k/c$, $n = m/c$, and $r = q^c$. It then suffices to show the existence of an l-space U of $AG(n, r)$ that is not an l/p-space of $AG(n/p, r^p)$ for any common prime divisor p of n and l. Now let $p_1, \cdots, p_e$ be the common prime divisors of r and l and put $W_i = GF(r^{p_i})$ for $i = 1, \cdots, e$. Then, $W = W_1 + \cdots + W_e$ is an h-subspace of $AG(h, r)$, where $h \leqq p_1 + \cdots + p_e \leqq p_i \cdot \cdots \cdot p_e$, and W contains the 1-subspace $U_0 = GF(r)$. It is easy to check that $h \leqq n - l$ (since $l < n$ and since $p_i \cdots p_e$ divides both l and n). Thus we may select an $(l - 1)$-space U_1 that is disjoint from W; then $U = U_0 + U_1$ is an l-space satisfying $U \cap W = U_0$. We claim that U is the desired l-space. Assume that U is in $AG(n/p, r^p)$, where $p = p_i$ is a prime dividing both n and l; then U is fixed under $GF(r^p)^*$. Since $1 \in U_0$, this implies that $W_i \subset U$, a contradiction. □

These results allow us to make the following observations.

THEOREM 3.2. *Every k-orbit of $AG(m, q)$ is regular if and only if $(k, m) = 1$. Every k-orbit of $AG(m, q)$ is either minimal or regular if and only if $d = (m, k)$ is a prime. If k divides m, then $AG(m, q)$ contains a unique minimal k-orbit.*

Proof. The first two assertions are immediate from Theorem 3.1. If k divides m, any minimal orbit belongs to a 1-space of $AG(m/k, q^k)$ by Lemma 3.2. Since G is transitive on such spaces, we get the last assertion. □

More generally, given any specific pair of positive integers m, k, we may recursively determine the exact number of orbits of any possible size by using the lattice of divisors of $d = (m, k)$. We give the following example.

Example 3.1. Let $k = 2d$ and $m = nd$, where n is odd, and assume that $d = pp'$ is the product of two distinct primes. By Theorems 3.1 and 3.2, every minimal orbit has size $(q^m - 1)/(q^d - 1)$ and belongs to a 2-space of $AG(n, q^d)$. Thus the number of minimal orbits in this case is

$$\sigma_{pp'} = \sigma_d = \begin{bmatrix} n \\ 2 \end{bmatrix}_{q^d} (q^d - 1)/(q^m - 1) = (q^{d(n-1)} - 1)/(q^{2d} - 1).$$

Every orbit of size $(q^m - 1)/(q^p - 1)$ belongs to a k/p-space in $AG(m/p, q^p)$, which is not a 2-space of $AG(n, q^d)$. Thus we have σ_p such orbits, where

$$\sigma_p = \left(\begin{bmatrix} np' \\ 2p' \end{bmatrix}_{q^p} - \begin{bmatrix} n \\ 2 \end{bmatrix}_{q^d} \right) (q^p - 1)/(q^m - 1);$$

similarly, we obtain the number $\sigma_{p'}$ of orbits of size $(q^m - 1)/(q^{p'} - 1)$. Finally, the regular orbits belong to k-spaces of $AG(m, q)$, which are neither k/p-spaces of $AG(m/p, q^p)$ nor k/p'-spaces of $AG(m/p', q^{p'})$. This gives σ_1 regular orbits, where

$$\sigma_1 = \left(\begin{bmatrix} m \\ k \end{bmatrix}_q - \begin{bmatrix} np' \\ 2p' \end{bmatrix}_{q^p} - \begin{bmatrix} np \\ 2p \end{bmatrix}_{q^{p'}} + \begin{bmatrix} n \\ 2 \end{bmatrix}_{q^d} \right) (q - 1)/(q^m - 1).$$

It should now be clear how the divisor lattice of d can be used to compute the numbers of orbits of the various possible sizes, in general. For our purposes, however, it does not seem worthwhile to produce a general result along these lines. Note that the first part of Theorem 3.2 corresponds to Lemma 4.2.6 of Hirschfeld [4]; our proof, however, is somewhat simpler (being coordinate-free).

4. Skew and covering multiplicative orbits. We now prove our main results.

THEOREM 4.1. *Let $\mathscr{B}$ be any k-orbit of $AG(m, q)$, where $1 \leqq k \leqq m - 1$, and let d be any positive integer satisfying $m \geqq d \geqq m/(m - k)$. Then there exists a d-orbit of $AG(m, q)$, which is skew to $\mathscr{B}$.*

Proof. The assertion is trivial if $d > k$. If $d = k$, there are distinct multiplicative k-orbits (as $d = k \neq 1, m - 1$), and, again, the assertion is obvious. Thus assume that $d \leqq k - 1$. Let V be a linear k-space contained in $\mathscr{B}$. Then any d-orbit that is not skew to $\mathscr{B}$ contains a linear d-space U contained in V. Thus the number of d-orbits not skew to $\mathscr{B}$ is bounded from above by the number $\begin{bmatrix} k \\ d \end{bmatrix}_q$ of d-spaces contained in V, and it clearly suffices to show that the total number of d-orbits exceeds $\begin{bmatrix} k \\ d \end{bmatrix}_q$. Now $AG(m, q)$ contains exactly $\begin{bmatrix} m \\ d \end{bmatrix}_q$ linear d-spaces, and each d-orbit has size at most $(q^m - 1)/(q - 1)$ (as seen in § 3). Hence there are at least $(q - 1)\begin{bmatrix} m \\ d \end{bmatrix}_q/(q^m - 1)$ d-orbits, and we want to show that

$$(2) \qquad (q - 1)\begin{bmatrix} m \\ d \end{bmatrix}_q > (q^m - 1)\begin{bmatrix} k \\ d \end{bmatrix}_q,$$

$$(q - 1)(q^{m-1} - 1)(q^{m-2} - 1) \cdots (q^{m-d+1} - 1) > (q^k - 1)(q^{k-1} - 1) \cdots (q^{k-d+1} - 1),$$

or, reordering terms,

$$(3) \qquad \frac{q^{m-1} - 1}{q^k - 1} \cdot \frac{q^{m-2} - 1}{q^{k-1} - 1} \cdot \cdots \cdot \frac{q^{m-d+1} - 1}{q^{k-d+2} - 1} > \frac{q^{k-d+1} - 1}{q - 1}.$$

However, $(q^{m-1-a}-1)/(q^{k-a}-1) > q^{m-k-1}$ for $a = 0, \cdots, d-2$, and thus the left-hand side of (3) is larger than $q^{(m-k-1)(d-1)}$. Also, the right-hand side of (3) is smaller than q^{k-d+1}, and thus it suffices to show that

$$(m-k-1)(d-1) \geqq k-d+1.$$

It is easily deduced that this inequality is equivalent to our hypothesis $d \geqq m/(m-k)$, which finishes the proof. □

Together with Corollary 2.1 and Theorem 2.1, we now have the following result.

THEOREM 4.2. *Let $\mathscr{B}$ be any k-orbit of $AG(m, q)$, where $1 \leqq k \leqq m-1$. Then the smallest integer d for which a d-orbit skew to $\mathscr{B}$ exists is $d = \lceil m/(m-k) \rceil$. Equivalently, the size of a smallest multiplicative pair-splitting set contained in $\mathscr{B}$ is d.*

Theorem 4.2 guarantees the existence of a quasi-perfect multiplicative pair-splitting set of $AG_k(m, q)$ (in any arbitrary k-orbit), but unfortunately does not provide us with an efficient way of finding such a set. This matter is discussed further in § 6. We can easily construct such sets explicitly in some special cases, shown below.

Example 4.1. Assume that $m = 2nk$, where k is odd; by Theorem 4.2, we have that $d = 2$. Since 2 and k divide m, choose the k-orbit $\mathscr{B}$ represented by $V = GF(q^k)$, and let U be the 2-space $U = GF(q^2)$. Since any image of U (under $\gamma \in \mathbb{F}^*$) is invariant under $GF(q^2)^*$ and since $GF(q^k)$ does not contain $GF(q^2)$, we see that the 2-orbit belonging to U is indeed skew to $\mathscr{B}$: no $U\gamma$ can be contained in $GF(q^k)$. Using a primitive element ω of $GF(q^m)^*$, we may describe U and V explicitly, as follows:

$$U = \{0\} \cup \{\omega^{[(q^m-1)/(q^2-1)]i} : 0 \leqq i \leqq q^2-2\},$$

$$V = \{0\} \cup \{\omega^{[(q^m-1)/(q^k-1)]i} : 0 \leqq i \leqq q^k-2\},$$

and a corresponding pair-splitting set is given by V and $V\omega^{(q^m-1)/(q^2-1)}$.

We conclude with a remark. From a geometric point of view, we might also be interested in the question of *covering* a given d-orbit $\mathscr{D}$ by a k-orbit $\mathscr{B}$; that is, each d-flat in $\mathscr{D}$ should be contained in some k-flat in $\mathscr{B}$. Of course, this can be done for any $k \geqq d$, trivially. However, we might ask for which k every k-orbit covers $\mathscr{D}$. Clearly, this holds for every k satisfying $d < m/(m-k)$, i.e., $k > (m(d-1))/d$, by Corollary 2.1. On the other hand, if $d \geqq m/(m-k)$, then Theorem 4.1 guarantees the existence of some d-orbit $\mathscr{D}'$ that is not covered by all k-orbits. We now show that here, in fact, *no* d-orbit is covered by all k-orbits. Let the d-orbit $\mathscr{D}$ be represented by the d-space U. Clearly, each k-orbit covering $\mathscr{D}$ contains a k-space V containing U. Thus the number of k-orbits covering $\mathscr{D}$ is at most the number of k-spaces containing U, which is equal to $\begin{bmatrix} m-d \\ k-d \end{bmatrix}_q$. Again, the total number of k-orbits is at least $(q-1)\begin{bmatrix} m \\ k \end{bmatrix}_q/(q^m-1)$, as each k-orbit has size $\leqq (q^m-1)/(q-1)$. Thus it suffices to show that

$$(q-1)\begin{bmatrix} m \\ k \end{bmatrix}_q > \begin{bmatrix} m-d \\ k-d \end{bmatrix}_q (q^m-1), \tag{4}$$

which can be proved as the corresponding inequality in the proof of Theorem 4.1: after substituting and reordering, (4) is seen to be equivalent to

$$\frac{q^{m-1}-1}{q^k-1} \cdot \cdots \cdot \frac{q^{m-d+1}-1}{q^{k-d+2}-1} > \frac{q^{k-d+1}-1}{q-1}, \tag{5}$$

which is exactly the same expression as in the proof of Theorem 4.1; hence (4) holds for $d \geqq m/(m-k)$, or, equivalently, $k \leqq (m(d-1))/d$. We have proved the following theorem.

THEOREM 4.3. *Let $\mathscr{D}$ be a positive integer, where $1 \leqq d \leqq m-1$. Then every d-orbit of $AG(m, q)$ is covered by every k-orbit if and only if $k > (m(d-1))/d$.*

5. A generalization. In applying pair-splitting sets to the factorization of polynomials, we might sometimes be able to guarantee that all the roots of the polynomial under consideration are contained in a given l-space L of $\mathbb{F}$. It then would be sufficient to assure that L can be split. More formally, a set S of s parallel classes of k-spaces of $AG(m, q)$ is said to be a *pair-splitting set relative to L* if and only if no two elements a, $b \in L$ are in a common k-flat in each of the s parallel classes in S. This amounts to saying that the restriction of S to the pairwise balanced design $\mathscr{D}'$ induced by $AG_k(m, q)$ on L is a pair-splitting set for $\mathscr{D}'$. As in the proof of Theorem 1.1, we then get the next lemma.

LEMMA 5.1. *Any pair-splitting set of $AG_k(m, q)$ relative to an l-space of $AG(m, q)$ contains at least $\lceil l/(m-k) \rceil$ parallel classes.*

Now denote by L^* the set $L^* = L \setminus \{0\}$. The proof of Lemma 2.1 can be modified to show the following result.

LEMMA 5.2. *Let V be a k-dimensional linear subspace of $\mathbb{F} = GF(q^m)$ over $GF(q)$, and let L be an l-space. Moreover, let $\gamma_1, \gamma_2, \cdots, \gamma_s$ be distinct elements in $\mathbb{F}^*$, and let $\beta_i = \gamma_i^{-1}$ for $1 \leqq i \leqq s$. Then the parallel classes determined by $V\gamma_1, \cdots, V\gamma_2$ form a pair-splitting set relative to L if and only if the subspace $U = \langle \beta_1, \cdots, \beta_2 \rangle$ is not contained in any of the k-spaces $V\delta^{-1}$, $\delta \in L^*$.*

Thus we must replace the orbit $\mathscr{B}$ of parallel classes under G (determined by V) by the smaller "partial orbit" of parallel classes under $(L^*)^{-1}$. (Note that this notion is no longer independent of the choice of the representative k-space V, and thus the geometric terminology of skew orbits given previously does not make much sense for our generalization!) Note also that neither the β_i nor the γ_i are restricted to elements in L^*. We now prove an analogue of Theorem 4.2.

THEOREM 5.1. *The size of a smallest pair-splitting set in $AG_k(m, q)$ relative to an l-space L is precisely $d = \lceil l/(m-k) \rceil$.*

Proof. In view of Lemma 5.1, it suffices to construct an example with exactly d parallel classes. Thus choose a k-space V. In view of Lemma 5.2, we must find a d-space U such that $U\delta \not\subseteq V$ for all $\delta \in L^*$. As in the proof of Theorem 4.1, there are at most $\begin{bmatrix} k \\ d \end{bmatrix}_q$ d-spaces U violating this condition. On the other hand, there are $\begin{bmatrix} m \\ d \end{bmatrix}_q$ d-spaces in $AG(m, q)$, which give rise to at least $(q-1)\begin{bmatrix} m \\ d \end{bmatrix}_q/(q-1)$ "partial orbits" with respect to L^*. (Note that different partial orbits might even overlap!) Thus it suffices to show that

$$(q-1)\begin{bmatrix} m \\ d \end{bmatrix}_q > (q^l - 1)\begin{bmatrix} k \\ d \end{bmatrix}_q \tag{6}$$

or, equivalently,

$$\frac{q^m - 1}{q^k - 1} \cdot \cdots \cdot \frac{q^{m-d+1} - 1}{q^{k-d+1} - 1} > \frac{q^l - 1}{q - 1}, \tag{7}$$

which is seen to be true, as in the corresponding argument in the proof of Theorem 4.1. □

6. A construction for optimal pair-splitting sets. In this section we describe a procedure for constructing a pair-splitting set of $\lceil m/(m-k) \rceil$ parallel classes in $AG_k(m, q)$. We begin by stating a simple observation.

LEMMA 6.1. *A set R of t parallel classes in $AG_k(m, q)$ is a pair-splitting set if and only if the linear subspaces contained in these classes have only the zero element in common.*

The proof is straightforward and is omitted.

To find t k-dimensional linear subspaces that have only the zero vector in common, we use the orthogonal complement of a subspace. The orthogonal complement of a k-dimensional subspace S is the set of all vectors orthogonal (under the standard inner product) to all vectors in S. This subspace is denoted $S^{\perp}$ and has dimension $m - k$.

LEMMA 6.2. *The linear subspaces $W_1, W_2, \cdots, W_t$ have only the zero vector in common if and only if $\cup_{i=1}^{t} W_i^{\perp}$ contains a basis for $AG(m, q)$.*

Proof. It is clear that, if S_1 and S_2 are linear subspaces and $S_1 \subseteq S_2$, then $S_2^{\perp} \subseteq S_1^{\perp}$.

Suppose that $W_1, W_2, \cdots, W_t$ have only the zero vector in common, but $\cup_{i=1}^{t} W_i^{\perp}$ does not contain a basis. Then $\cup_{i=1}^{t} W_i^{\perp} \subseteq S$, where S is a linear subspace of dimension $m - 1$. Since $W_i^{\perp} \subseteq S$, $1 \leqq i \leqq t$, then $S^{\perp} \subseteq W_i$. $S^{\perp}$, however, has dimension one, contradicting the fact that $\cap_{i=1}^{t} W_i = \{0\}$.

Conversely, suppose that $\cup_{i=1}^{t} W_i^{\perp}$ contains a basis. If $\cap_{i=1}^{t} W_i = S$, then $\cup_{i=1}^{t} W_i^{\perp} \subseteq S^{\perp}$, which implies $S = \{0\}$. □

As an immediate consequence of the preceding two lemmata we have the following result.

COROLLARY 6.1. *A set R of t parallel classes in $AG_k(m, q)$ is pair-splitting if and only if the orthogonal complements of the linear subspaces in R contain a basis for $AG(m, q)$.*

Let us first consider the case where $m - k$ divides m, and let $a = m/(m - k)$. $GF(q^{m-k})$ is an $(m-k)$-dimensional subspace S of $AG(m, q)$. Select a basis $\beta_1, \beta_2, \cdots, \beta_a$ for $GF(q^m)$ over $GF(q^{m-k})$. Now $\beta_i S$, $1 \leqq i \leqq a$ are $(m - k)$-dimensional subspaces of $AG(m, k)$. It readily follows that $\cup_{i=1}^{a} \beta_i S$ contains a basis for $GF(q^m)$ or $AG(m, q)$. By Lemma 6.2, $\{(\beta_1 S)^{\perp}, (\beta_2 S)^{\perp}, \cdots, (\beta_a S)^{\perp}\}$ is a set of k-dimensional linear subspaces having only the zero vector in common. It follows from Lemma 6.1 that the translates of these spaces give a set of a parallel classes in $AG_k(m, q)$ that form a pair-splitting set.

Now suppose that $(m - k)$ does not divide m. Let $m = a(m - k) + b$, where $0 < b < m - k$. Consider an $a(m - k)$-flat H in $AG(m, q)$. It is isomorphic to $AG(a(m - k), q)$, and hence, by the results of the previous paragraph, we can find a set S of a subspaces of dimension $m - k$ that contain a basis for H. Since $b < m - k$, it is a simple matter to select a subspace U of dimension $(m - k)$ such that $(\cup_{W \in S} W) \cup U$ contains a basis of $AG(m, q)$. It now follows from Lemmas 6.1 and 6.2 that there exist $(a + 1)$ parallel classes in $AG(m, q)$ that form a pair-splitting set. This set is optimal.

We can modify the preceding construction in the case where $m - k \mid m$ to produce a pair-splitting set in a multiplicative orbit. As before, select a basis $\beta_1, \beta_2, \cdots, \beta_a$ for $GF(q^m)$ over $GF(q^{m-k})$. Let L_i be the one-dimensional subspace generated by β_i. Let $L_i^{\perp}$ be the $(a - 1)$-dimensional orthogonal complement of L_i over $GF(q^{m-k})$. Now $\cap_{i=1}^{a} L_i^{\perp} = \{0\}$, and, because $\{L_i^{\perp}\}$ are a hyperplanes in $AG(a, q^{m-k})$, they lie in a Singer cycle. Hence there exist elements in $GF(q^{m-k})$, $\gamma_1, \gamma_2, \cdots, \gamma_a$ such that $\{(L_i^{\perp} : 1 \leqq i \leqq a\} = \{\gamma_i L_1^{\perp} : 1 \leqq i \leqq a\}$. This set gives rise to a pair-splitting set in a multiplicative orbit of $AG(m, q)$. The optimal pair-splitting sets that were constructed in the preceding paragraphs are not necessarily contained in a multiplicative orbit. To find a near optimal pair-splitting set from a single orbit, we might take a probabilistic approach.

Suppose that $\mathscr{B}$ is a k-orbit in $AG(m, q)$. Consider the probability P of randomly selecting a d-orbit that is skew to $\mathscr{B}$. An upper bound on the number of d-orbits not

skew to $\mathscr{B}$ is $\begin{bmatrix} k \\ d \end{bmatrix}_q$ and a lower bound on the number of d-orbits in the entire space is $\begin{bmatrix} m \\ d \end{bmatrix}_q / ((q^m - 1)/(q - 1))$. Hence

$$1 - P \leqq \left(\begin{bmatrix} k \\ d \end{bmatrix}_q \frac{(q^m - 1)}{(q-1)} \right) \Big/ \begin{bmatrix} m \\ d \end{bmatrix}_q < q^{m - d(m-k)}.$$

This and Theorem 2.1 show that the probability of selecting $d = \lceil m/(m-k) \rceil + i$ linearly independent elements whose inverses give rise to d parallel classes in $\mathscr{B}$ that form a splitting set is $P > 1 - q^{-i(m-k)}$. Hence the probability of randomly selecting a splitting set that is close to optimal is good. It can be shown that it can be determined whether a set of parallel classes is pair-splitting, in time polynomial in the dimension of the space. This is based on the facts that division of affine polynomials by affine polynomials is more efficient than division of ordinary polynomials and that intersections of k-flats can be found by computing the greatest common divisor of the corresponding polynomial representations [9].

7. Conclusion. The results of this paper answer what we believe to be several interesting geometric questions and show that methods previously used for factoring polynomials over extension fields can be generalized from the use of hyperplanes to the use of k-flats. In particular, the paper demonstrates that the trace function used by Berlekamp [2] is not fundamental to the factoring technique he proposed.

The concept of a pair-splitting set can be generalized [7] to an arbitrary resolvable or near-resolvable block design. In the general case, the exact size of an optimal pair-splitting set is unknown. Moreover, there is no known efficient procedure for either finding such a set or for checking whether a given set of parallel classes is pair-splitting. In contrast, the results of this paper are surprisingly strong but seem to depend critically on the rich geometric and algebraic structure of the affine spaces.

Acknowledgment. The first two authors would like to thank the University of Waterloo for its hospitality during the time of their research.

REFERENCES

[1] E. R. BERLEKAMP, *Algebraic Coding Theory*, McGraw-Hill, New York, 1968.
[2] ———, *Factoring polynomials over large finite fields*, Math. Comp., 24 (1970), pp. 713–735.
[3] T. BETH, D. JUNGNICKEL, AND H. LENZ, *Design Theory*, Bibliographisches Institut, Mannheim, 1985; Cambridge University Press, Cambridge, 1986.
[4] J. W. P. HIRSCHFELD, *Projective Geometries over Finite Fields*, Oxford University Press, Oxford, 1979.
[5] D. JUNGNICKEL, *On automorphism groups of divisible designs*, Canadian J. Math., 34 (1982), pp. 257–297.
[6] J. SINGER, *A theorem in finite projective geometry and some applications to number theory,* Trans. Amer. Math. Soc., 43 (1938), pp. 377–385.
[7] P. C. van Oorschot and S. A. Vanstone, *On splitting sets in block designs and finding roots of polynomials*, Discrete Math., 84 (1990), pp. 71–85.
[8] ———, *A geometric approach to root finding in $GF(q^m)$*, IEEE Trans. Inform. Theory, IT-35 (1989), pp. 444–453.
[9] P. C. van Oorschot, *Combinatorial and computational issues related to finding roots of polynomials over finite fields*, Ph.D. thesis, University of Waterloo, Waterloo, Ontario, Canada, 1988.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 460–466, November 1992
© 1992 Society for Industrial and Applied Mathematics

LAPLACIAN PERMANENTS OF TREES*

PHILLIP BOTTI†, RUSSELL MERRIS†‡, AND CHERYL VEGA†

Abstract. Let T be a tree on n vertices. The Laplacian matrix $L(T)$ is the difference of the diagonal matrix of vertex degrees and the adjacency matrix. The main result of this article is that, for "almost all" trees T, there is a nonisomorphic tree T' such that per $L(T)$ = per $L(T')$. The proof follows the approach taken by Schwenk in [*New Directions in the Theory of Graphs*, F. Harary, ed., Academic Press, New York, 1973, pp. 275–307]. The difficulty is finding a single pair of "super" trees from which to start. The search for this pair was greatly facilitated by a new algorithm for computing Laplacian permanents of trees. This algorithm is also reported. Finally, the algorithm is used to establish inequalities for per $L(T)$.

Key words. algorithm, cospectral, permanent, tree, Laplacian matrix

AMS(MOS) subject classifications. 05C05, 15A15

Let $T = (V, E)$ be a tree with vertex set V and edge set E. Denote by $d(v)$ the degree of vertex v, and by $D(T)$ the $n \times n$ diagonal matrix of vertex degrees. Then the *Laplacian matrix* is defined by $L(T) = D(T) - A(T)$, where $A(T)$ is the symmetric 0–1 adjacency matrix. The Laplacian matrix has been the object of considerable recent study stimulated in part by Fiedler's *algebraic connectivity* [6] and in part by applications in chemistry [5], [9], [13], statistics [4], and parallel algorithms for sparse matrix computations [14].

Strictly speaking, $L(T)$ depends on some ordering of V. It is for this reason that we typically study not $L(T)$ itself, but some function of $L(T)$ that is invariant under permutation similarity. The permanent is one such function. Work on per $L(T)$ is reported, e.g., in [1], [2], [8], [12], [17], [18]. After a promising start, this work seems to have stalled, perhaps due to the notorious computational intractability of the permanent [7], [10], [16].

In a talk at the April 1990 Lisbon Workshop on Multilinear Algebra, Brualdi mentioned a matrix "contraction" that was useful in [3]. We use these contractions to produce a (fast) graph-theoretic algorithm for computing the permanent of $L(T)$ directly from T.

ALGORITHM. Let $T = (V, E)$ be a tree on two or more vertices.

Step 1. Initialize.
- (a) Define $p(v) = d(v)$, $v \in V$.
- (b) Define $q(e) = 1$, $e \in E$.

Step 2. Contract.
- (a) Choose a pendant vertex x and let $e = \{x, y\}$ denote the (unique) edge incident with x.
- (b) Replace $p(y)$ with $p(x)p(y) + q(e)$.
- (c) For each edge e' incident with y, replace $q(e')$ with $p(x)q(e')$.
- (d) Eliminate vertex x and edge e.

Step 3. Finished? If y is the only remaining vertex, go to Step 4. Otherwise, go to Step 2.

Step 4. Answer: per $L(T) = p(y)$.

* Received by the editors July 27, 1990; accepted for publication (in revised form) July 18, 1991.

† Department of Mathematics and Computer Science, California State University, Hayward, California 94542.

‡ This author's work was supported by a California State University, Hayward Research, Scholarship and Creative Activity Award and by National Security Agency grant MDA904-90-H-4024.

Proof. Let B be any $k \times k$ matrix and suppose that some column of B has just two nonzero entries, say, a and b. Since our interest is in computing the permanent, we may assume that

$$B = \begin{pmatrix} a & \text{—}u\text{—} \\ b & \text{—}w\text{—} \\ 0 & C \end{pmatrix},$$

where u and w are $1 \times (k-1)$ submatrices and C is a $(k-2) \times (k-1)$ submatrix. A "contraction" of B is a $(k-1)$-square matrix

$$B^{\#} = \begin{pmatrix} \text{—}aw + bu\text{—} \\ C \end{pmatrix}.$$

It follows from the Laplace expansion theorem for permanents and the multilinearity of the permanent function that per (B) = per $(B^{\#})$.

We now turn to $L(T) = D(T) - A(T)$. Since T is a tree, $L(T)$ is acyclic—only diagonal products corresponding to matchings contribute to the permanent. That is, only permutations whose disjoint cycle factorizations consist entirely of 1-cycles and 2-cycles need be considered. In particular, per $L(T)$ = per (B), where $B = D(T) + A(T)$. The algorithm corresponds to repeated contractions on B. Initially, the (v, v)-entry of B is $p(v) = d(v)$, $v \in V$, and the (v_1, v_2)-entry is either $q(e) = 1$ or 0, depending on whether $e = \{v_1, v_2\}$ is an edge.

Choose some pendant vertex x and its (only) neighbor y. We interpret $B^{\#}$ as a matrix obtained from B by eliminating the row and column corresponding to x and modifying the row corresponding to y. After the first iteration, it is clear that the new value of $p(y)$ (in $B^{\#}$) is what it should be. Note, however, that only the entry in row y corresponding to e' has been multiplied by $p(x)$. The entry in column y corresponding to e' remains unchanged. Thus $B^{\#}$ is not symmetric. However, its rows and columns are indexed by the remaining vertices; the (v, v)-entry is $p(v)$, and, for any edge $e = \{v_1, v_2\}$, the *product* of the (v_1, v_2)-entry and the (v_2, v_1)-entry is $q(e)$. We now replace B with $B^{\#}$ and iterate again.

Consider an intermediate stage. In the tree, we eliminate what will have become a vertex-edge pendant pair (x, e). In matrix B, we eliminate row and column x and modifying row y, where $\{x, y\} = e$. The (y, y)-entry of (the new) $B^{\#}$ is the sum of $p(x)p(y)$ and the product of the (x, y)-entry and the (y, x)-entry of B, i.e., $p(x)p(y) + q(e)$. Exactly one of the two entries corresponding to e' is multiplied by $p(x)$, namely, the one in row y. Thus the product of the two entries in $B^{\#}$ corresponding to e' is the (new) value of $q(e')$. (Note that both the (v_1, v_2)-entry and the (v_2, v_1)-entry of $B^{\#}$ are zero if $\{v_1, v_2\}$ is not an edge.) □

Example. Figure 1 illustrates an implementation of the algorithm. The corresponding matrix result is

$$\text{per} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 3 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix} = 60.$$

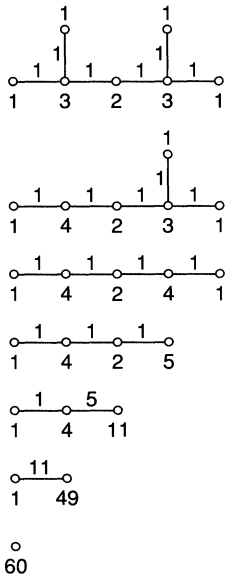


FIG. 1

Of course, we may save time, e.g., by contracting all the original pendants at once. This is done in Fig. 2. Indeed, Figs. 1 and 2 illustrate the smallest pair of nonisomorphic trees with the same Laplacian permanent. There is another such pair on 8 vertices, 5 pairs on 9 vertices, and 15 pairs among the 106 trees on 10 vertices. We now present the main result of this article.

THEOREM 1. *Let t_n be the number of (nonisomorphic, unlabeled) trees on n vertices. Let s_n be the number of such trees T for which there exists a nonisomorphic tree T' such that* per $L(T)$ = per $L(T')$. *Then* $\lim_{n\to\infty}(s_n/t_n) = 1$.

Proof. Since the publication of [15], results of this kind are proved in two stages. In the first stage, we find a single pair of nonisomorphic trees with certain properties. In the second stage, we appeal to Schwenk's probabilistic result that almost all trees have a prescribed (finite) branch. Our proof is no exception. Figure 3 shows two trees, each with a (single, pendant) vertex marked with the letter "A." In either case, if the algorithm is employed with the design to "finish up" at vertex A, in the penultimate step, $p(y)$ =

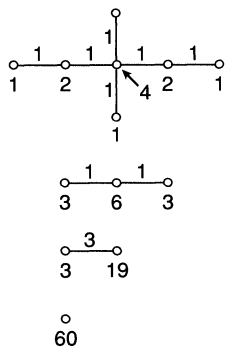


FIG. 2

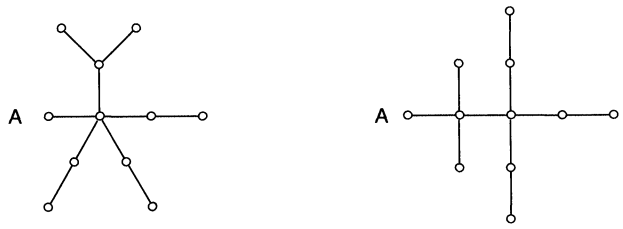


FIG. 3

837 and $q(e') = 135$. It follows not only that the permanent is 972 in both cases, but, more importantly for us, it is achieved in the same manner. Suppose now that T is some tree with a large number of vertices. Suppose further that T has some vertex "A" with a branch isomorphic to the tree T_1 in Fig. 3. Form tree T' by replacing this branch with the tree T_2. It is evident from the algorithm that per $L(T) =$ per $L(T')$. By [15, Thm. 7], "almost all" trees have a branch isomorphic to T_1. □

Because $L(T)$ is positive semidefinite, it follows from the Hadamard theorem for permanents [11] that

$$\text{per } L(T) \geqq h(T) = \prod_{v \in V} d(v).$$

In fact, it was proved in [2] that per $L(T) \geqq 2h(T)$, with equality if and only if $T = K_{1,n-1}$, the "star." However, the Hadamard theorem for permanents establishes that per $L(T) \geqq \prod p(v)$ at every stage of the algorithm. For example, see the following theorem.

THEOREM 2. *Let $T = (V, E)$ be a tree on $n \geqq 3$ vertices. For each $v \in V$, define $r(v)$ to be $d(v)$* plus *the number of pendant vertices adjacent to v. Then*

$$\text{per } L(T) \geqq \prod_{v \in V} r(v). \tag{1}$$

Proof. Contract all the original pendant vertices and apply the Hadamard theorem for permanents. □

Note, in Fig. 2, that $2h(T) = 32$, while the bound in (1) is 54. In fact, we claim that the bound given by (1) is always better than $2h(T)$, unless T is the star. To see this, consider a longest path in $T \neq K_{1,n-1}$, beginning and ending at pendants u and w, say. Suppose that x and y are the (distinct) vertices adjacent to u and w. Let $d_1 = d(x)$, $d_n = d(y)$, and d_2 through d_{n-1} be the degrees of the remaining vertices. Since $r(x) = 2d_1 - 1$ and $r(y) = 2d_2 - 1$,

$$\begin{aligned}
\text{per } L(T) &\geqq \prod_{v \in V} r(v) \\
&\geqq (2d_1 - 1)(d_2 \cdots d_{n-1})(2d_n - 1) \\
&\geqq 4h(T) - 2(d_1 + d_n)(d_2 \cdots d_{n-1}) + 1 \\
&> 2h(T)
\end{aligned}$$

because $d_1 \geqq 2$ and $d_n \geqq 2$ imply that $(d_1 + d_n) \leqq d_1 d_n$. □

THEOREM 3. *Let $T = (V, E)$ be a tree. Suppose that $v \in V$, has degree $d = d(v) > 2$. Suppose that one of the branches at v is a path with pendant vertex u. If w is*

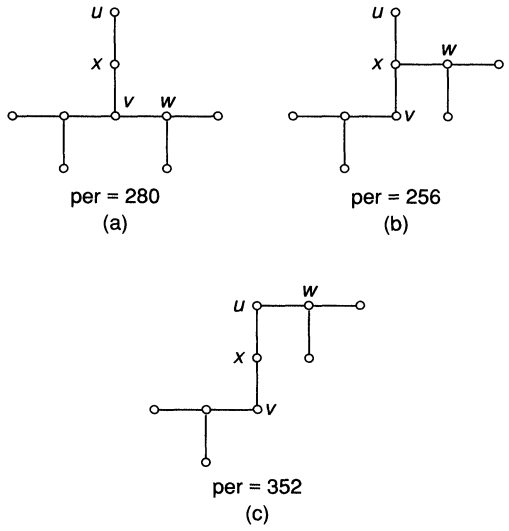


FIG. 4

adjacent to v *on a branch other than this path, then* per $L(T') >$ per $L(T)$, *where* T' *is the tree obtained from* T *by deleting edge* $\{v, w\}$ *and adding a new edge* $\{u, w\}$.

(We observe that moving a branch only part way out a path may *decrease* the Laplacian permanent. See Fig. 4.)

Proof. The tree T is illustrated in Fig. 5(a). Suppose that the path contains k edges. Employ the algorithm to prune away all but w from the branch of T at v containing w, and so arrive at Fig. 5(b). Pruning away the path from vertex v and then taking off w results in Fig. 5(c), where $a_1 = 1$, $a_2 = 3$, and $a_{j+1} = 2a_j + a_{j-1}$. Similarly, tree T' is illustrated in Fig. 6(a). First, prune away all but w from the branch of T' at u containing

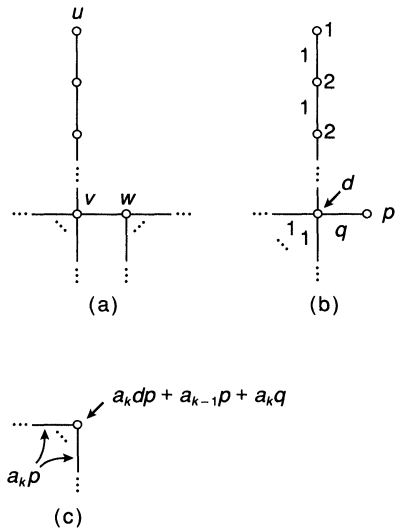


FIG. 5

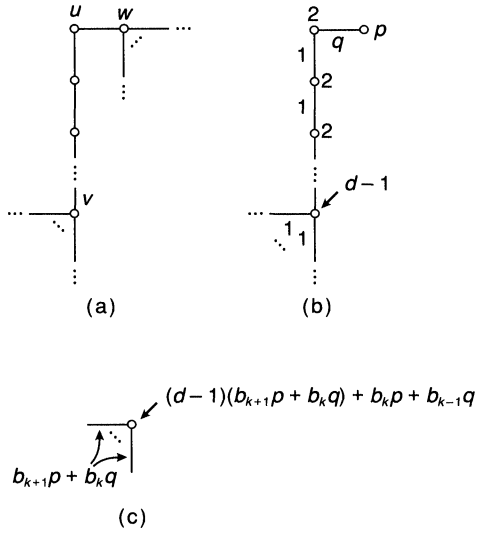


FIG. 6

w to arrive at Fig. 6(b). Then prune away the path from vertex v to obtain Fig. 6(c), where $b_1 = 1$, $b_2 = 2$, and $b_{j+1} = 2b_j + b_{j-1}$. It remains to show that

$$a_k p < b_{k+1} p + b_k q \tag{2}$$

and

$$a_k dp + a_{k-1} p + a_k q < (d-1)(b_{k+1} p + b_k q) + b_k p + b_{k-1} q. \tag{3}$$

Since $b_{k+1} = a_k + b_k$, (2) is immediate. Using this and the similar identity $a_{k+1} = b_{k+1} + b_k$, we may show that (3) reduces to the inequality $2 < d$. □

COROLLARY 1 (see [2, Thm. 2.5]). *Let T be a tree on n vertices. Then* per $L(T) \leqq$ per $L(P_n)$, *with equality if and only if $T = P_n$.*

Proof. Any tree on n vertices may be transformed into the path P_n by a succession of modifications, as described in Theorem 3. □

REFERENCES

[1] R. BAPAT, *A bound for the permanent of the Laplacian matrix*, Linear Algebra Appl., 74 (1986), pp. 219–223.

[2] R. BRUALDI AND J. GOLDWASSER, *Permanent of the Laplacian matrix of trees and bipartite graphs*, Discrete Math., 48 (1984), pp. 1–21.

[3] R. BRUALDI AND B. SHADER, *Matrices of 0's and 1's with restricted permanental minors*, Discrete Math., 96 (1991), pp. 161–174.

[4] G. CONSTANTINE, *Graph complexity and the Laplacian matrix in blocked experiments*, Linear and Multilinear Algebra, 28 (1990), pp. 49–56.

[5] B. EICHINGER, *Configuration statistics of Gaussian molecules*, Macromolecules, 13 (1980), pp. 1–11.

[6] M. FIEDLER, *Algebraic connectivity of graphs*, Czech. Math. J., 23 (1973), pp. 298–305.

[7] J. VON ZUR GATHEN, *Algebraic complexity theory*, Technical Report no. 207188, University of Toronto, Toronto, Ontario, Canada, 1988.

[8] J. GOLDWASSER, *Permanent of the Laplacian matrix of trees with a given matching*, Discrete Math., 61 (1986), pp. 197–212.

[9] I. GUTMAN, *Graph-theoretical formulation of Foresman's equations*, J. Chem. Phys., 68 (1978), pp. 1321–1322.

[10] W. HARTMANN, *On the complexity of immanants*, Linear and Multilinear Algebra, 18 (1985), pp. 127–140.
[11] M. MARCUS, *The Hadamard theorem for permanents*, Proc. Amer. Math. Soc., 15 (1964), pp. 967–973.
[12] R. MERRIS, *The Laplacian permanental polynomial for trees*, Czech. Math. J., 32 (1982), pp. 397–403.
[13] ———, *An edge version of the matrix-tree theorem and the Wiener index*, Linear and Multilinear Algebra, 25 (1989), pp. 291–305.
[14] A. POTHEN, H. SIMON, AND K. LIOU, *Partitioning sparse matrices with eigenvectors of graphs*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 430–452.
[15] A. J. SCHWENK, *Almost all trees are cospectral*, in New Directions in the Theory of Graphs, F. Harary, ed., Academic Press, New York, 1973, pp. 275–307.
[16] L. G. VALIANT, *The complexity of computing the permanent*, Theoret. Comput. Sci., 8 (1979), pp. 189–201.
[17] A. VRBA, *The permanent of the Laplacian matrix of a bipartite graph*, Czech. Math. J., 36 (1986), pp. 7–17.
[18] ———, *Principal subpermanents of the Laplacian matrix*, Linear and Multilinear Algebra, 19 (1986), pp. 335–346.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 467–474, November 1992
© 1992 Society for Industrial and Applied Mathematics
003

THE 1/3–2/3 CONJECTURE FOR 5-THIN POSETS*

GRAHAM BRIGHTWELL† AND COLIN WRIGHT‡

Abstract. The 1/3–2/3 conjecture says that, in any finite poset that is not totally ordered, vertices x and y can be found such that the proportion of linear extensions of the poset in which x is above y lies between 1/3 and 2/3 inclusive. In this paper, the conjecture is established in the case where every element of the poset is incomparable with at most five others. The proof involves the use of a computer to eliminate a large number of cases.

Key words. posets, linear extension, 1/3–2/3 conjecture

AMS(MOS) subject classification. 06A10

1. Introduction. Throughout this paper, $(X, <)$ denotes a partially ordered set (poset). Unless otherwise stated, the underlying set X is assumed to be finite. A *linear extension* of $(X, <)$ is a total order $\prec$ on X such that $x \prec y$ whenever $x < y$. We adopt the convention that the symbol $<$ indicates a partial order, while $\prec$ indicates a total order. If x and y are elements of $(X, <)$, the *probability* $P(x \prec y)$ that x is below y is defined to be the proportion of linear extensions $\prec$ of $(X, <)$ in which $x \prec y$. In particular, if $x < y$, then $P(x \prec y) = 1$. A fair amount is known about the behaviour of this probability (see, for instance, Brightwell [1], or the survey articles by Graham [4], Rival [8], and Winkler [10]), but one particularly appealing conjecture has remained unresolved.

CONJECTURE 1 (The 1/3–2/3 conjecture). *Let $(X, <)$ be a partial order that is not a total order. Then there exist distinct elements x, y of X such that*

$$1/3 \leqq P(x \prec y) \leqq 2/3.$$

The example of the poset with three elements and one relation shows that the bounds 1/3 and 2/3 cannot be improved.

Conjecture 1 was apparently first formulated by Fredman in about 1975 but did not appear in print until 1984 (Linial [7]). In that paper, Linial proved that the result is true for posets of width 2. Some further progress has been made since: Kahn and Saks [5] showed that the weaker version of Conjecture 1 with 3/11 and 8/11 replacing 1/3 and 2/3 is true, and Brightwell [2] established the conjecture in the case where $(X, <)$ is a semiorder. Very recently, Fishburn, Gehrlein, and Trotter [3] proved the conjecture for height 1 partial orders. Also, Komlós [6] proved the qualitative result that, as the width of a height 1 poset tends to infinity, the minimum of $|1/2 - \Pr(x \prec y)|$, for elements x, y of the poset, tends to zero. More information on Conjecture 1 can be found in the survey article by Saks [9].

For k a positive integer, we say that a poset $(X, <)$ is k-thin if every element of X is incomparable with at most k other elements of X. The purpose of this paper is to prove the following result.

* Received by the editors September 5, 1990; accepted for publication (in revised form) July 2, 1991. Much of this research was carried out while both authors were at the University of Cambridge, Cambridge, United Kingdom, and later while the second author was at the University of Manchester, Manchester, United Kingdom.

† London School of Economics and Political Science, Houghton Street, London WC2A 2AE, United Kingdom.

‡ Computer Laboratory, Chadwick Tower, University of Liverpool, Liverpool L69 3BX, United Kingdom.

THEOREM 2. *Let $(X, <)$ be a 5-thin partial order that is not a total order. Then there exist distinct elements x, y of X such that*

$$1/3 \leqq P(x \prec y) \leqq 2/3.$$

In other words, we prove Conjecture 1 in the case where $(X, <)$ is 5-thin. To be more accurate, we describe a computer program whose termination (without finding a counterexample) would imply this result. We also assure the reader that we have run the program, and that it did terminate without finding a counterexample.

It is perhaps worth noting here that Theorem 2 is a strong indication that Conjecture 1 is true. Indeed, if a poset $(X, <)$ is a counterexample to the 1/3–2/3 conjecture, every element tends to be fairly firmly fixed at a particular level in a linear extension; thus every element tends to be incomparable with few others. This is, of course, only a heuristic argument, and we would hesitate to suggest that Theorem 2 is actually a step toward a proof of Conjecture 1. We expand on this later.

2. Theory. If $(X, <)$ is a poset with no pair of elements x, y such that $1/3 \leqq P(x \prec y) \leqq 2/3$, then we say that the poset $(X, <)$ satisfies the 1/3–2/3 *condition*. Thus Conjecture 1 states that the only finite posets that satisfy the 1/3–2/3 condition are the total orders. We note that the definition of probability used here can be extended to the class of infinite posets that are k-thin for some k, and that there are infinite posets that are not total orders and yet satisfy the 1/3–2/3 condition. See Brightwell [1] for details.

Clearly, if $(X, <)$ satisfies the 1/3–2/3 condition, then the relation $\prec_0$, defined by $x \prec_0 y$ whenever $P(x \prec y) > 2/3$, is a total order, in fact, a linear extension of $(X, <)$. We often wish to emphasise the role of this linear order, so we also make the following definition. If $\prec_0$ is a linear extension of the poset $(X, <)$, we say that the *pair* $((X, <), \prec_0)$ satisfies the 1/3–2/3 *condition* if $P(x \prec y) > 2/3$ whenever $x \prec_0 y$. (Note that, if $((X, <), \prec_0)$ and $((X, <), \prec_1)$ both satisfy the 1/3–2/3 condition, then $\prec_0 = \prec_1$.)

Before we start, we make one trivial but important simplification. We claim that, to prove Theorem 2, we may assume without loss of generality that the poset $(X, <)$ has at least two minimal elements. Indeed, suppose that the result is false, and let $(X, <)$ be a counterexample with the minimum number of vertices. If $(X, <)$ has a unique minimal element x, then deleting it from the poset leaves us with a smaller counterexample, a contradiction.

Our next step is to find circumstances in which we can conclude that various posets do not satisfy the 1/3–2/3 condition. One of our main tools in this direction is a lemma that can be found in Brightwell [2]. To state this result, we need some additional terminology.

Suppose that x and y are incomparable elements of a poset $(X, <)$. An *up-separator* for (x, y) is an element u of X that covers x and is incomparable with y. Similarly, a *down-separator* for (x, y) is an element covered by y and incomparable with x. An element of X is a *separator* for (x, y) if it is either an up- or a down-separator.

Intuitively, separators for (x, y) are elements whose presence in X forces x to appear below y in a high proportion of the linear extensions. For instance, if u is an up-separator for (x, y), then u is below y in many linear extensions, which forces x to be below y also. The next lemma supports this intuition. For a proof, see Brightwell [2].

LEMMA 3. *If the pair $((X, <), \prec_0)$ satisfies the 1/3–2/3 condition, then, for every pair (x, y) of incomparable elements with $x \prec_0 y$, we have either* (i) *a separator u for (x, y) such that $x \prec_0 u \prec_0 y$, or* (ii) *at least two separators for (x, y).*

Let $(X, <)$ be a finite poset, and let $\prec_0$ be a linear extension of $(X, <)$. If x and y are incomparable elements of X with $x \prec_0 y$, say, then x and y are 2-*separated* (in $((X, <), \prec_0)$) if either there is a separator u for (x, y) with $x \prec_0 u \prec_0 y$, or there are at least 2 separators for (x, y). The pair $((X, <), \prec_0)$ is said to be 2-*separated* if every pair of incomparable elements is 2-separated in $((X, <), \prec_0)$. Thus Lemma 3 says that every pair $((X, <), \prec_0)$ satisfying the 1/3–2/3 condition is 2-separated.

Unfortunately, there are posets that are 2-separated in some order, yet do not satisfy the 1/3–2/3 condition. Figure 1, below, shows a particularly simple example.

Our strategy for proving Theorem 2 is roughly as follows. We are given a poset $(X, <)$ (5-thin, with at least 2 minimals), and we wish to prove not only that it does not satisfy the 1/3–2/3 condition, but also that no poset having a down-set isomorphic to $(X, <)$ can satisfy the condition.

To be more accurate, we define a *configuration* to be a triple $((X, <), F, \prec_0)$, where $(X, <)$ is a finite poset, F is a down-set of X, and $\prec_0$ is a linear order on F extending $<|_F$. We say that a configuration $((X, <), F, \prec_0)$ is 2-*separated* if, whenever x and y are elements of F with $x \prec_0 y$, then x and y are 2-separated in $(X, <)$. Also, for k a positive integer, we say that the configuration is *k-thin* if $(X, <)$ is k-thin.

If $(Y, <')$ is a poset and $\prec$ is a linear extension of $(Y, <')$, then $((Y, <'), \prec)$ is a *continuation* of the configuration $((X, <), F, \prec_0)$ if the following three conditions are satisfied:

(i) $(Y, <')$ contains (an isomorphic copy of) $(X, <)$ as a down-set,

(ii) All elements of Y incomparable with an element of F are in X,

(iii) $\prec|_F = \prec_0$.

A configuration is *discardable* if there is no continuation of it satisfying the 1/3–2/3 condition.

LEMMA 4. *Every configuration that is not 2-separated is discardable.*

Proof. Suppose that the configuration $((X, <), F, \prec_0)$ is not 2-separated, and let x and y be a pair of elements of F that are not 2-separated in $(X, <)$. Then these elements are not 2-separated in any continuation, since any separator must be incomparable with either x or y. Hence, by Lemma 3, no continuation satisfies the 1/3–2/3 condition, and the configuration is discardable. □

We prove Theorem 2 by producing a finite list of discardable configurations such that every pair $((Y, <'), \prec)$, with the poset 5-thin, having at least two minimals, and

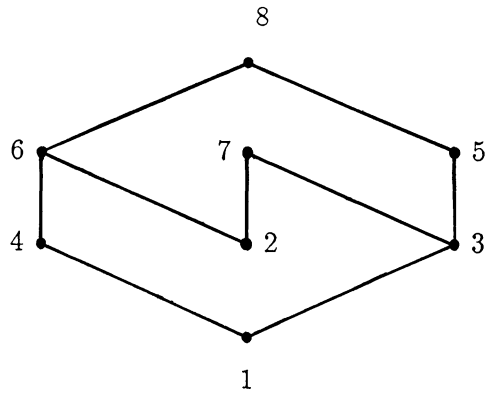


FIG. 1. *A 2-separated poset.*

the pair 2-separated, is a continuation of some configuration in the list. Clearly, the existence of such a list, together with Lemma 4, would imply that the only 5-thin posets satisfying the 1/3–2/3 condition are total orders.

(Observe that there is no such finite list if we omit the condition that there be at least two minimals. To see this, let Y_n be the poset formed by taking a total order on n elements, and adding two incomparable elements above everything else. Now, if we have any finite list of configurations that does not include a chain (in particular, a finite list of discardable configurations), then some Y_n fails to be a continuation of any configuration in the list.)

The example in Fig. 1 shows that we cannot produce such a list if our only technique for demonstrating a configuration to be discardable is based on 2-separation. Accordingly, our next step is to describe another method of showing configurations to be discardable.

Let $((Y, <'), \prec)$ be a continuation of a configuration $((X, <), F, \prec_0)$, and let $K = X \setminus F$. We can partition the set of linear extensions of $(Y, <')$ according to which elements of K come below the highest element of X in the linear extension.

For any linear extension $\prec$ of $(Y, <')$, define

$$V(\prec) = \{x \in K : x \prec y \text{ for some } y \in F\}.$$

Now set $\mathcal{V} = \{V \subseteq K : V = V(\prec) \text{ for some linear extension } \prec \text{ of } (Y, <')\}$, and Φ_V to be the event $V(\prec) = V$, for $V \in \mathcal{V}$. Then, for every pair x, y of incomparable elements of F, we have that

$$(*) \qquad P(x \prec y) = \sum_{V \in \mathcal{V}} P(x \prec y \mid \Phi_V) P(\Phi_V).$$

The point of this is that the various $P(x \prec y \mid \Phi_V)$ depend only on $(X, <)$, as every linear extension of $(Y, <')$ satisfying Φ_V is specified uniquely by a pair $(\prec', \prec'')$ of linear orders, where $\prec'$ is a linear extension of $<|_{F \cup V}$ in which the highest element is an element of F, and $\prec''$ is a linear extension of $<'|_{Y-F-V}$. Note also that $\mathcal{V}$ depends only on $(X, <)$, as it consists of all the down-sets V of K such that there is some element of F incomparable to everything in V.

Therefore, if there is some continuation $(Y, <')$ of the configuration $((X, <), F, \prec_0)$ that satisfies the 1/3–2/3 condition, then there is a probability vector $(P(\Phi_V))_{V \in \mathcal{V}}$ such that the sum $(*)$ is greater than 2/3 whenever $x \prec_0 y$. In other words, if there is no such probability vector, then the configuration is discardable.

For a given configuration $((X, <), F, \prec_0)$, let $((x_i, y_i))_{i=1}^{p}$ run through all pairs of incomparable elements of F with $x_i \prec_0 y_i$. The array of values $(P(x_i \prec y_i \mid \Phi_V))_{i=1, V \in \mathcal{V}}^{p}$ can be considered as a matrix $\mathbf{P} = \mathbf{P}_{iV}$. We have seen that the configuration is discardable if there is no probability vector $\mathbf{a}_V$ with $\mathbf{P}_{iV}\mathbf{a}_V > 2/3$. The theory of linear programming shows that this is the case precisely when there is a probability vector $\mathbf{b}_i$ such that $\mathbf{b}_i^T \mathbf{P}_{iV} \leqq 2/3$. Translating back, we are seeking a probability vector $(\mathbf{b}_i)_{i=1}^{p}$ such that

$$\sum_{i=1}^{p} \mathbf{b}_i P(x_i \prec y_i \mid \Phi_V) \leqq 2/3$$

for every $V \in \mathcal{V}$. We call such a vector $\mathbf{b}$ a *witness* for the configuration. If a configuration has a witness, it is discardable.

To summarise, what we have shown is that the following result implies Theorem 2.

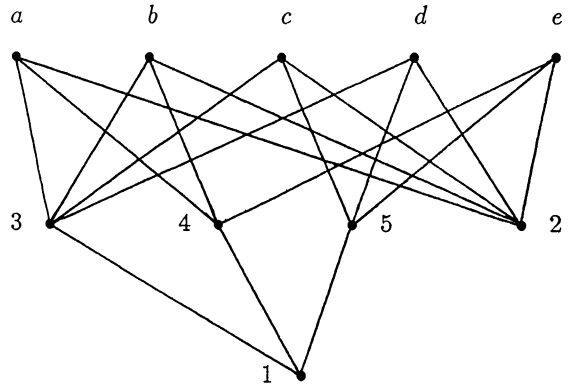


FIG. 2. *A discardable configuration.*

THEOREM 5. *There is a finite list $\mathcal{L}$ of configurations with the following properties*:
(a) *Every 5-thin poset $(Y, <')$ with at least two minimals, which is 2-separated in an order $\prec$, is a continuation of some configuration in $\mathcal{L}$*;
(b) *Every configuration in $\mathcal{L}$ has a witness.*

With the aid of a computer, we have indeed found such a list, consisting of some 38,372 configurations, thus proving Theorem 2.

3. An example. In this section, we give an example of a discardable configuration.

Figure 2, above, shows a poset $(X, <)$. The five elements labelled $1, \cdots, 5$ comprise the down-set F, with the labelling indicating the linear extension $\prec_0$. The configuration $((X, <), F, \prec_0)$ is 2-separated—for instance, c and d are two separators for $(3, 4)$, and 1 and e act as separators for $(2, 3)$. It can also be checked that the configuration is 5-thin.

Note that every element of $K = \{a, b, c, d, e\}$ is incomparable with just one element of F. Therefore the possible sets $V(\prec)$ are subsets of $\{x \in K : x \text{ incomparable with } n\}$, for $n = 3, 4, 5$. Hence $\mathcal{V} = \{\varnothing, \{a\}, \{b\}, \{a, b\}, \{c\}, \{d\}, \{c, d\}, \{e\}\}$.

Taking $V = \{a\}$ as an example, $\Phi_V = \Phi_{\{a\}}$ is the event that $a \prec 5 \prec b$. There are eight linear extensions of $F \cup \{a\}$ consistent with Φ_V, namely, those with 5 top, then a, then the other four elements in some legal order. Thus we calculate $P(1 \prec 2 \mid \Phi_V) = 3/4$, $P(2 \prec 3 \mid \Phi_V) = 5/8$, $P(3 \prec 4 \mid \Phi_V) = 1/2$, $P(4 \prec 5 \mid \Phi_V) = 1$, and $P(3 \prec 5 \mid \Phi_V) = 1$.

Table 1, below, gives the probabilities for a selection of these relations, conditioned on other events Φ_V.

TABLE 1

V	$P(2 \prec 3 \mid \Phi_V)$	$P(3 \prec 4 \mid \Phi_V)$	$P(4 \prec 5 \mid \Phi_V)$	Combined
$\varnothing$	3/5	1/2	1/2	11/20
$\{a\}$	5/8	1/2	1	31/48
$\{b\}$	5/8	1/2	1	31/48
$\{a, b\}$	5/8	1/2	1	31/48
$\{c\}$	5/8	1	0	31/48
$\{d\}$	5/8	1	0	31/48
$\{c, d\}$	5/8	1	0	31/48
$\{e\}$	1	0	1/2	7/12

The final column of this table gives the value

$$\tfrac{1}{2}P(2 \prec 3 | \Phi_V) + \tfrac{1}{3}P(3 \prec 4 | \Phi_V) + \tfrac{1}{6}P(4 \prec 5 | \Phi_V).$$

As is evident, this is less than $2/3$ for every $V \in \mathscr{V}$. Therefore the probability vector $(1/2, 1/3, 1/6)$ acts as a witness for this configuration. The interpretation is that the convex combination

$$\tfrac{1}{2}P(2 \prec 3) + \tfrac{1}{3}P(3 \prec 4) + \tfrac{1}{6}P(4 \prec 5)$$

is less than $2/3$ in any continuation of the configuration; so, in any continuation, at least one of the three probabilities is less than $2/3$.

4. The program. In this section, we give a very brief description of the programming methods used to find the list $\mathscr{L}$ mentioned above. A much fuller account, including a listing of the programs used, appears in [11].

The basic approach is to try to "grow" a counterexample. Thus, given a configuration, we try to find a witness for it. If we fail, we add another "layer" of vertices to the top of the configuration, in all possible ways subject to the conditions that our configuration be 5-thin and 2-separated. This gives us a list of new configurations, and we try to prove that each of these is discardable by finding a witness. Each time we do find a witness for a configuration, that configuration can be added to our list $\mathscr{L}$.

A priori, there is no reason why this process should terminate: we might be able to add new layers indefinitely, even if Theorem 2 is correct. However, if we can continue indefinitely, then we would be constructing a "one-way infinite" 5-thin poset satisfying the $1/3$–$2/3$ condition, and it seems unlikely that such a poset can exist if Theorem 2 is true. In practice, we are eventually left with no new configurations, which suffices to prove Theorem 5, and hence Theorem 2. In fact, we have even shown a little more. We say that a poset is *locally finite* if there are only finitely many elements in each interval $(z : x < z < y)$. Our method proves that, if a locally finite 5-thin poset with a minimal element satisfies the $1/3$–$2/3$ condition, then it is a total order.

We now describe the procedure we adopt for finding our list $\mathscr{L}$ of configurations, as in Theorem 5. The account below is somewhat simplified, and we have omitted certain simplifying checks and tests that were implemented at various points of the process. Refer to [11] for a full account.

We begin with the configuration $C_0 = ((X, <), F, \prec_1)$, where $(X, <)$ consists of two unrelated elements x and y, with $x \prec_1 y$ and $F = \varnothing$. Clearly, every poset with at least two minimal elements is a continuation of this basic configuration C_0.

Now C_0 evidently has no witness, so we generate all the 5-thin, 2-separated continuations $((Y, <'), \prec)$ of C_0 in which every element of Y is incomparable with either x or y.

One of the key points is that there are only finitely many such continuations. Indeed, if $((Y, <'), \prec)$ is a 5-thin continuation of any configuration $((X, <), F, \prec_0)$, then all but at most eight elements of $Y \setminus X$ are above all the elements of X. (For a proof, see Brightwell [1].) Thus there are only boundedly many 5-thin continuations $((Y, <'), \prec)$ in which every element of Y is incomparable with some element of X. Also, every continuation $((Z, <''), \prec)$ of $((X, <), F, \prec_0)$ is also a continuation of one of the boundedly many configurations $((Y, <'), X, \prec)$. The extra set of elements $Y \setminus X$ should be considered the next layer of the poset.

Having generated all the 5-thin, 2-separated continuations $((Y, <'), \prec)$ of C_0, we consider the set $\mathscr{N}$ of configurations $((Y, <'), X, \prec_0)$. We know that, if there is a 5-thin poset with at least two minimals satisfying the $1/3$–$2/3$ condition, then it is a

continuation of some configuration in $\mathcal{N}$. We take each of the configurations $C \in \mathcal{N}$ in turn, and try to find a witness for it. If a witness is found, the configuration C is discardable, and is put into the list $\mathcal{L}$. If a witness cannot be found, we form all the 5-thin, 2-separated continuations of C whose every element is incomparable with some element of Y, and repeat the process. Whenever we form a new configuration in this way, the specified down-set is just the set of vertices in the original configuration.

Thus, in practice, we maintain a list $\mathcal{M}$ of configurations "to be considered." Every time we fail to find a witness for a configuration, that configuration is put on the end of the list $\mathcal{M}$. When we later return to it, we put another layer of vertices at the top of the configuration, in all possible ways, and try to find witnesses for each of these new configurations.

If there were a 5-thin poset, with at least two minimals, that satisfies the 1/3–2/3 condition, then at all times it would be a continuation of some configuration in $\mathcal{M}$; eventually, we would construct this poset and would recognise it as a poset with no witness, whose last added layer was empty. As we have already said, this did not occur, and the list $\mathcal{M}$ was eventually found to be empty. At this stage, witnesses had been found for 38,372 configurations.

We do not propose to provide more detail here regarding the algorithms used, but we should mention a few facts concerning the implementation and running of the programs. These were written in BCPL as implemented on the IBM 3084Q at Cambridge University. The operating system was Phoenix. The total CPU time was approximately 4 hours 30 minutes; the total program length exceeded 2,500 lines, although this was in several independent programs. Refer to Wright [11] for more information about either the algorithms or their implementation.

Let us instead discuss the implications of our technique and result for Conjecture 1. First, let us stress that there is nothing special about 5-thinness. If, as we believe, there is no finite or "one-way infinite" poset satisfying the 1/3–2/3 condition (except for total orders), then our method of proof can, in principle, be used to prove Conjecture 1 for k-thin posets, for any fixed value of k. However, it seems inevitable that the size of the list of configurations required would grow extremely rapidly with k. To give some idea of the explosion involved, after one "round" of the algorithm, the number of configurations in our list $\mathcal{M}$ "to be dealt with" was 122: when we tried the same process for $k = 6$, after one round there were 16,536 configurations in $\mathcal{M}$. We can also expect the number of rounds required to be greater for 6-thin than for 5.

Nevertheless, if a result along the lines of "no poset containing an element x incomparable with 13 others satisfies the 1/3–2/3 condition" could be proved, then Conjecture 1 could perhaps be regarded as essentially solved. (Needless to say, we have no idea how to prove such a result!)

The state of knowledge as regards Conjecture 1 is rather curious. The only known infinite posets satisfying the 1/3–2/3 condition are 4-thin semiorders with width at most 3. Indeed, it seems natural to believe that posets satisfying the 1/3–2/3 condition will tend to have low width, be k-thin for some small k, and be "almost" semiorders. However, the cases where the conjecture has been proved are those of width 2, 5-thin, and semiorders. The only significant result "from the other end," eliminating an unlikely class of possible counterexamples, is that of Fishburn, Gehrlein, and Trotter [3] for height 1 posets. In view of these results, it would be highly surprising (at least to us) if Conjecture 1 were false.

REFERENCES

[1] G. R. BRIGHTWELL, *Linear extensions of infinite posets*, Discrete Math., 70 (1988), pp. 113–136.
[2] ———, *Semiorders and the* 1/3–2/3 *conjecture*, Order, 5 (1989), pp. 369–380.

[3] P. FISHBURN, W. GEHRLEIN, AND W. T. TROTTER, *The* 1/3–2/3 *conjecture for height* 1 *posets*, submitted.
[4] R. L. GRAHAM, *Applications of the* FKG *inequality and its relatives*, in Mathematical Programming—The State of the Art, Bonn 1982, A. Bachem, M. Grötschel, and B. Korte, eds., Springer-Verlag, Berlin, New York, pp. 115–131.
[5] J. KAHN AND M. SAKS, *Balancing poset extensions*, Order, 1 (1984), pp. 113–126.
[6] J. KOMLÓS, *A strange pigeon-hole principle*, Order, 7 (1991), pp. 107–114.
[7] N. LINIAL, *The information theoretic bound is good for merging*, SIAM J. Comput., 13 (1984), pp. 795–801.
[8] I. RIVAL, *Linear extensions of finite ordered sets,* Ann. Discrete Math., 23 (1984), pp. 355–370.
[9] M. SAKS, *Balancing linear extensions of ordered sets*, Order, 2 (1985), pp. 327–330.
[10] P. M. WINKLER, *Correlation and order*, in Combinatorics and Ordered Sets: Proceedings of a Summer Research Conference held August 11–17, 1984, Humboldt State University, Arcata, CA, Contemporary Mathematics, 57, I. Rival, ed., pp. 151–174.
[11] C. D. WRIGHT, *Combinatorial Algorithms*, Ph.D. thesis, Cambridge University, Cambridge, UK, 1990.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 475–490, November 1992
© 1992 Society for Industrial and Applied Mathematics
004

THE EQUIVALENT SUBGRAPH AND DIRECTED CUT POLYHEDRA ON SERIES-PARALLEL GRAPHS*

SUNIL CHOPRA†

Abstract. The families of minimal directed cuts and minimal equivalent subgraphs of a directed graph form a pair of blocking clutters. A directed graph is series-parallel if the undirected graph obtained on ignoring directions is series-parallel. It is shown that the minimal equivalent subgraph inequalities completely describe the directed cut polyhedron, and that the minimal directed cut inequalities completely describe the equivalent subgraph polyhedron on strongly connected series-parallel graphs.

Key words. polyhedra, series-parallel, directed cuts, equivalent subgraphs

AMS(MOS) subject classifications. 05C40, 90C27

1. Introduction. $G = (V, A)$ refers to a directed graph with node set V and arc set A. The arc a directed from node u to v is referred to as a or (u, v). Given a directed graph $G = (V, A)$, a subgraph $ES = (V, A_E)$ is said to be an *equivalent subgraph* if ES has a directed path between vertices u and v if and only if G does, for every pair $u, v \in V$. The set of arcs C is said to define a *directed cut* if there is no directed path from u to v in $\bar{G} = (V, A - C)$ for some pair of nodes u and v in V. G is said to be *strongly connected* if there is a directed path between every pair of nodes u and v. In this paper, we assume that G is strongly connected. Given a set of nodes $V_1 \subseteq V$, define $A(V_1, V - V_1)$, where $A(V_1, V - V_1) = \{(s, t) \in A \mid s \in V_1, t \in V - V_1\}$. The arcs in $A(V_1, V - V_1)$ define a directed cut for $1 \leqq |V_1| \leqq |V| - 1$.

Given a finite set A, a *clutter* C is a family of subsets of A such that no member of $\mathcal{F}$ contains another member of $\mathcal{F}$. Let $\mathcal{F}_B$ be the family of all minimal subsets of A having a nonempty intersection with each member of $\mathcal{F}$. $\mathcal{F}_B$ is called the *blocking clutter*, or simply the *blocker*, of $\mathcal{F}$.

Let $\mathcal{E}$ be the set of all minimal equivalent subgraphs of G, and $\mathbf{C}$ the set of all minimal directed cuts of G. Clearly, $\mathcal{E}$ and $\mathbf{C}$ form a pair of clutters defined on the arc set A.

PROPOSITION 1.1. *$\mathcal{E}$ and $\mathbf{C}$ form a pair of blocking clutters.*

Proof. We first show that, if $\bar{G} = (V, \bar{A})$ is not an equivalent subgraph of G, then there is a directed cut containing no arc from $\bar{A}$. Since $\bar{G}$ is not an equivalent subgraph, there is a pair of nodes u and v such that $\bar{G}$ has no directed path from u to v. Let V_1 be the set of nodes that can be reached from u by means of directed paths in $\bar{G}$. Clearly, $v \notin V_1$. The set of arcs $A(V_1, V - V_1)$ is nonempty, since G contains a directed path from u to v, and defines a directed cut in G that contains no arc from $\bar{A}$.

On the other hand, consider an arc set C that is not a directed cut. The graph $\tilde{G} = (V, A - C)$ contains a path between every pair of nodes u and v in V and is thus an equivalent subgraph that contains no arc from C.

This proves that $\mathcal{E}$ and C form a pair of blocking clutters. □

Given a vector w indexed by the arc set A, we refer to the element of w corresponding to arc $a = (u, v)$ as w_a, $w(a)$, or $w(u, v)$. Given an equivalent subgraph $ES = (V, A_E)$,

* Received by the editors July 2, 1990; accepted for publication (in revised form) November 6, 1991.

† Department of Managerial Economics and Decision Sciences, J. L. Kellogg Graduate School of Management, Northwestern University, Evanston, Illinois 60208.

define its incidence vector $x(ES)$, where

$$x_a(ES) = \begin{cases} 1 & \text{if } a \in A_E, \\ 0 & \text{otherwise.} \end{cases}$$

Define the *equivalent subgraph polyhedron* $PE(G)$, where

$$PE(G) = \text{conv}\,\{x(ES) \mid ES \in \mathscr{E}\} + R_+^A.$$

Given a directed cut C, define its incidence vector $y(C)$, where

$$y_a(C) = \begin{cases} 1 & \text{if } a \in C, \\ 0 & \text{otherwise.} \end{cases}$$

Define the *directed cut polyhedron* $PC(G)$, where

$$PC(G) = \text{conv}\,\{y(C) \mid C \in \mathbf{C}\} + R_+^A.$$

Define the polyhedra $LPE(G)$ and $LPC(G)$, where

$$LPE(G) = \left\{ x \in R_+^A \,\middle|\, \sum_{a \in C} x_a \geqq 1 \ \forall C \in \mathbf{C} \right\},$$

$$LPC(G) = \left\{ y \in R_+^A \,\middle|\, \sum_{a \in A_E} y_a \geqq 1 \ \forall ES = (V, A_E) \in \mathscr{E} \right\}.$$

In general, both $LPE(G)$ and $LPC(G)$ have fractional vertices. From the results of Fulkerson [3] on blocking polyhedra, the following result holds.

PROPOSITION 1.2. *Each vertex of* $PE(G)$ $(PC(G))$ *is a vertex of* $LPE(G)$ $(LPC(G))$, *and each integer vertex of* $LPE(G)$ $(LPC(G))$ *is a vertex of* $PE(G)$ $(PC(G))$. *Thus*

$$PE(G) = \text{conv}\,\{x \in LPE(G) \mid x \text{ integer}\},$$

$$PC(G) = \text{conv}\,\{y \in LPC(G) \mid y \text{ integer}\}.$$

If we assign nonnegative weights w_a for each arc in A, the minimum weight equivalent subgraph problem (MWES) is finding a minimum weight member of $\mathscr{E}$, and the minimum weight directed cut problem (MWDC) is finding a minimum weight member of $\mathbf{C}$. MWES is NP-hard in general (see Garey and Johnson [4]). Thus it is unlikely that a complete inequality description of $PE(G)$ can be obtained in general. It can be solved in polynomial time on series-parallel graphs (Richey, Parker, and Rardin [6]). MWDC can be solved in polynomial time using flow techniques.

The clutter $\mathbf{C}(\mathscr{E})$ is said to have the *weak max-flow min-cut* property (see Seymour [7]) if $PC(G) = LPC(G)$ $(PE(G) = LPE(G))$.

In the next section, we show that, for connected series-parallel graphs, $PE(G) = LPE(G)$ and $PC(G) = LPC(G)$; i.e., both clutters $\mathbf{C}$ and $\mathscr{E}$ have the weak max-flow min-cut property. This is not true even for the complete directed graph on four nodes. Thus, in some sense, the above result is the best possible one. Consider the complete directed graph KD_4 on four nodes, shown in Fig. 1.1.

Consider the vector $\bar{x}$, where

$$\bar{x}(12) = \bar{x}(23) = \bar{x}(34) = \bar{x}(41) = \bar{x}(24) = \bar{x}(42) = \bar{x}(13) = \bar{x}(31) = \tfrac{1}{2}.$$

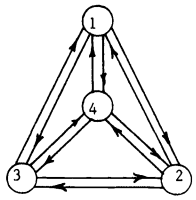


FIG. 1.1

$\bar{x}(a)$ is zero for all other arcs. $\bar{x}$ is a fractional vertex of $LPE(KD_4)$. Similarly, consider the vector $\bar{y}$, where

$$\bar{y}(12) = \bar{y}(21) = \bar{y}(14) = \bar{y}(41) = \bar{y}(31) = \bar{y}(34) = \bar{y}(43) = \bar{y}(42) = \tfrac{1}{3};$$

$$\bar{y}(3, 2) = \tfrac{2}{3}.$$

$\bar{y}(a)$ is zero for all other arcs. $\bar{y}$ is a fractional vertex of $LPC(KD_4)$.

At this stage, we mention that our definition of directed cuts is different from the "directed cut" defined by Lucchesi and Younger [5]. For $1 \leqq |V_1| \leqq |V| - 1$, Lucchesi and Younger define $A(V_1, V - V_1)$ to be a "directed cut" if $|A(V - V_1, V_1)| = 0$. Using this definition, they show that the family of "directed cuts" has the max-flow min-cut property (see [7]) for all connected directed graphs G.

2. *PE(G)* and *PC(G)* on series-parallel graphs. $UG = (V, E)$ refers to an undirected graph with edge set E and node set V. The undirected edge between u and v is referred to as $[u, v]$. UG is a *series-parallel* graph if it can be obtained from a forest by repeatedly adding an edge in parallel to an existing one or by replacing an edge by a path (Duffin [2]). The following characterization of series-parallel graphs is also well known.

PROPOSITION 2.1. *A connected series-parallel graph with no loops or nodes of degree one either contains a node of degree two or two parallel edges.*

This result can be extended to a form that is more suitable for our proof.

PROPOSITION 2.2. *A connected series-parallel graph with no loops or nodes of degree one contains one of the configurations of Fig.* 2.1, *below.*

Proof. Let UG be a connected series-parallel graph that is minimal, with respect to the property that it does not contain loops, nodes of degree one, or any one of Figs. 2.1(a)–2.1(d). By Proposition 2.1, it contains a node of degree two. Contract any of the edges incident to this node to get the graph UG'. By minimality of UG, one of Figs. 2.1(a)–2.1(d) must arise in UG'. If Fig. 2.1(a) arises in UG', then Fig. 2.1(c) was present in UG. If Fig. 2.1(b) arises in UG', then it was also present in UG. If Fig. 2.1(c) arises in UG', then either Fig. 2.1(b) or Fig. 2.1(d) was present in UG. If Fig. 2.1(d) arises, then Fig. 2.1(b) was present in UG. This contradicts the minimality of UG. The result thus follows. □

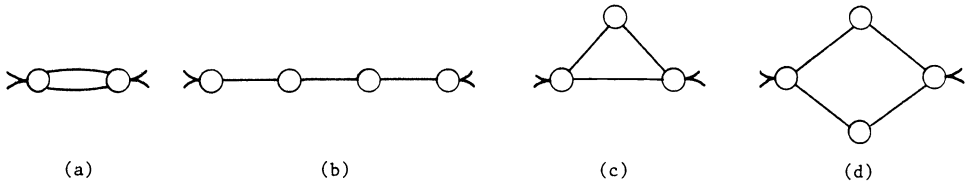


FIG. 2.1

Given any directed graph $G = (V, A)$, we can construct the corresponding undirected graph $UG = (V, E)$, where edge $[u, v] \in E$ if either $(u, v) \in A$ or $(v, u) \in A$. We say that G is *series-parallel* if the corresponding undirected graph UG is series-parallel.

Assume that $G = (V, A)$ is a connected, loopless directed graph such that $(u, v) \in A$ if and only if $(v, u) \in A$. G is clearly strongly connected. The main result of this paper is stated as Theorem 2.1.

THEOREM 2.1. *For a connected, loopless series-parallel directed graph $G = (V, A)$ such that $(u, v) \in A$ if and only if $(v, u) \in A$, we have that $PE(G) = LPE(G)$.*

Proof. Let $G = (V, A)$ be a connected, loopless series-parallel directed graph minimal with respect to the property that $LPE(G)$ has a fractional vertex $\bar{x}$. By Proposition 2.2, G has one of the configurations shown in Fig. 2.2, below.

In the following proof, we show that if G contains any of the above configurations (Fig. 2.2), then there exists a minor G' of G such that $LPE(G')$ also has a fractional vertex. The proof is presented as a sequence of five propositions, one for each of Figs. 2.2(a)–2.2(e).

PROPOSITION 2.3. *If G contains Fig.* 2.2(a), *set $G' = (V', A')$, where $V' = V - \{u_2\}$, $A' = A - \{(u_1, u_2), (u_2, u_1)\}$. The polyhedron $LPE(G')$ contains a fractional vertex.*

Proof. Since $\bar{x}$ is a vertex of $LPE(G)$, we must have that $\bar{x}(u_1, u_2) = \bar{x}(u_2, u_1) = 1$. x', the restriction of $\bar{x}$ to A', is a fractional vertex of $LPE(G')$. This contradicts the minimality of G. □

PROPOSITION 2.4. *If G contains Fig.* 2.2(b), *there exists a minor G' of G such that $LPE(G')$ has a fractional vertex.*

Proof. Let a_1 and a_2 be the two arcs from u_1 to u_2 and b_1 and b_2, the two arcs from u_2 to u_1. Since $\bar{x}$ is a vertex of $LPE(G)$, one of $\bar{x}(a_1)$, $\bar{x}(a_2)$ and one of $\bar{x}(b_1)$, $\bar{x}(b_2)$ must be zero. Without loss of generality, assume that $\bar{x}(a_2) = \bar{x}(b_2) = 0$. Form $G' = (V, A')$, where $A' = A - \{a_2, b_2\}$. x', the restriction of $\bar{x}$ to A', is a fractional vertex of $LPE(G')$, □

PROPOSITION 2.5. *If G contains Fig.* 2.2(c), *there exists a minor G' of G such that $LPE(G')$ has a fractional vertex.*

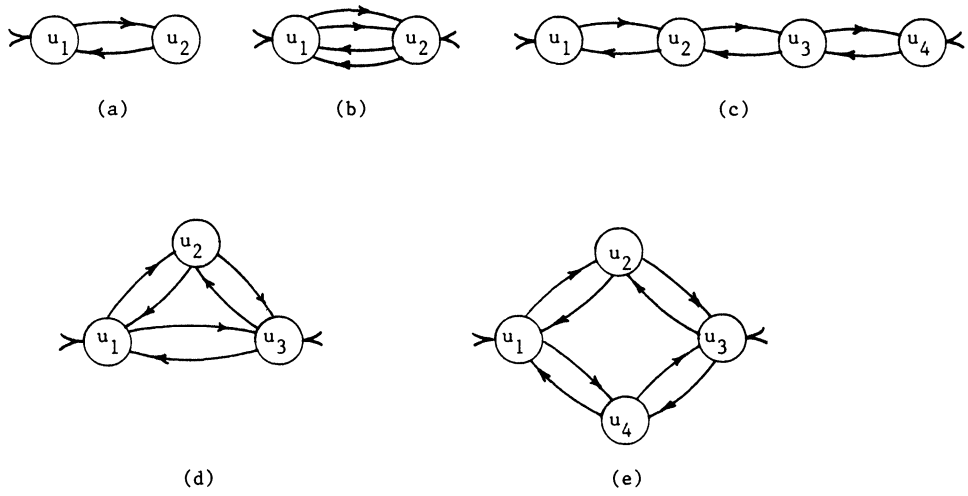


FIG. 2.2

Proof. Without loss of generality, we can assume that $\bar{x}(u_1, u_2) \geqq \bar{x}(u_2, u_3)$, $\bar{x}(u_3, u_4)$. If $\bar{x}(u_3, u_2) > \bar{x}(u_2, u_1)$, define $\tilde{x}$, where

$$\tilde{x}(a) = \begin{cases} \bar{x}(a) & \text{if } a \in A - \{(u_2, u_1), (u_3, u_2)\}, \\ \bar{x}(u_3, u_2) & \text{if } a = (u_2, u_1), \\ \bar{x}(u_2, u_1) & \text{if } a = (u_3, u_2). \end{cases}$$

$\tilde{x}$ is also a vertex of $LPE(G)$. We can perform a similar transformation if $\bar{x}(u_4, u_3) > \bar{x}(u_2, u_1)$. Thus we can assume that $\bar{x}(u_2, u_1) \geqq \bar{x}(u_3, u_2)$, $\bar{x}(u_4, u_3)$.

Contract the arcs (u_1, u_2) and (u_2, u_1) to obtain $G' = (V', A')$. Let x' be the restriction of $\bar{x}$ to A'. We claim that x' is a fractional vertex of $LPE(G')$. Assume not. Then we can write

$$x' = \sum \alpha_i x'_i + \delta, \tag{2.1}$$

where $\alpha_i \geqq 0$, $\sum \alpha_i = 1$, $\delta \in R_+^{A'}$, and each x'_i is the incidence vector of a minimal equivalent subgraph of G'. All minimal equivalent subgraphs of G' have one of the following configurations; see Fig. 2.3.

Define

$$\gamma_1 = \{\textstyle\sum \alpha_i | \, x'_i \text{ has Fig. 2.3(a)}\},$$

$$\gamma_2 = \{\textstyle\sum \alpha_i | \, x'_i \text{ has Fig. 2.3(b)}\},$$

$$\gamma_3 = \{\textstyle\sum \alpha_i | \, x'_i \text{ has Fig. 2.3(c)}\},$$

$$\gamma_4 = \{\textstyle\sum \alpha_i | \, x'_i \text{ has Fig. 2.3(d)}\}.$$

Without loss of generality, we can assume that $\gamma_4 \leqq \gamma_3$. In fact, we can assume that $\gamma_4 = 0$. If $\gamma_4 > 0$, take a set S_3 of solutions x'_i containing Fig. 2.3(c) such that $\sum_{i \in S_3} \alpha_i \geqq \gamma_4$ and $\sum_{i \in S_3 - \{i^*\}} \alpha_i \leqq \gamma_4$. The solution i^* is treated as two different solutions, $i^*(1)$ and $i^*(2)$ with weights $\alpha_{i^*(1)} = \gamma_4 - \sum_{i \in S_3 - \{i^*\}} \alpha_i$ and $\alpha_{i^*(2)} = \alpha_{i^*} - \alpha_{i^*(1)}$, respectively. The solutions in the set $S_3 - \{i^*\} \cup \{i^*(1)\}$ are transformed to $\tilde{x}_i$, where

$$\tilde{x}_i(a) = \begin{cases} x'_i(a) & \text{for } a \in A' - \{(u_3, u_2), (u_3, u_4)\}, \\ 1 & \text{for } a = (u_3, u_4), \\ 0 & \text{otherwise.} \end{cases}$$

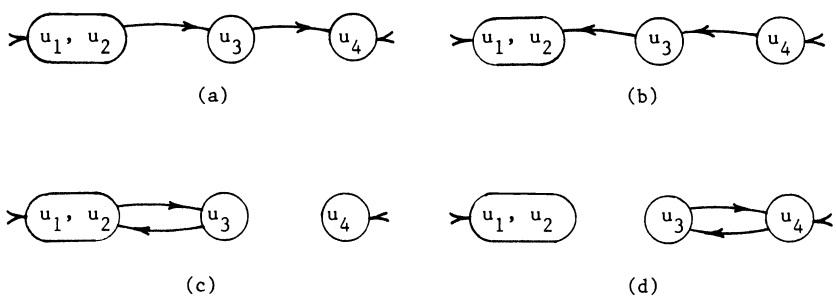


FIG. 2.3

Take the solutions containing Fig. 2.3(d) (the total weight of such solutions is γ_4) and transform them to $\tilde{x}_i$, where

$$\tilde{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_3, u_4), (u_3, u_2)\}, \\ 1 & \text{for } a = (u_3, u_2), \\ 0 & \text{otherwise.} \end{cases}$$

The two transformations are shown in Fig. 2.4. For all other solutions, set $\tilde{x}_i = x_i'$. Define $\tilde{\gamma}_1 = \gamma_1 + \gamma_4$, $\tilde{\gamma}_2 = \gamma_2 + \gamma_4$, $\tilde{\gamma}_3 = \gamma_3 - \gamma_4$, $\tilde{\gamma}_4 = 0$. We can write $x' = \sum \alpha_i \tilde{x}_i + \delta$. Thus we assume that $\gamma_4 = 0$.

Given $x_i' \in LPE(G')$ in (2.1), form $\bar{x}_i \in LPE(G)$ as follows:

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{if } a \in A - \{(u_1, u_2), (u_2, u_1)\}, \\ 1 & \text{if } a = (u_1, u_2) \text{ and } x_i' \text{ has Fig. 2.3(a) or Fig. 2.3(c)}, \\ 0 & \text{if } a = (u_1, u_2) \text{ and } x_i' \text{ has Fig. 2.3(b)}, \\ 1 & \text{if } a = (u_2, u_1) \text{ and } x_i' \text{ has Fig. 2.3(b) or Fig. 2.3(c)}, \\ 0 & \text{if } a = (u_2, u_1) \text{ and } x_i' \text{ has Fig. 2.3(a)}. \end{cases}$$

Each $\bar{x}_i$ is the incidence vector of an equivalent subgraph of G and

$$\bar{x} = \sum \alpha_i \bar{x}_i + \bar{\delta}, \tag{2.2}$$

where $\bar{\delta}_a = \delta_a$ for $a \in A - \{(u_1, u_2), (u_2, u_1)\}$ and $\bar{\delta}_a \geqq 0$ for $a \in \{(u_1, u_2), (u_2, u_1)\}$. Thus $\bar{x}$ is not a vertex of $LPE(G)$, contradicting our assumption. This shows that, if $\bar{x}$ is a vertex of $LPE(G)$, then x' is a vertex of $LPE(G)$, contradicting the minimality of G. Thus G does not contain Fig. 2.2(c). □

PROPOSITION 2.6. *If G contains Fig. 2.2(d), there exists a minor G' of G such that $LPE(G')$ has a fractional vertex.*

Proof. First, we show that $\bar{x}_a = 0$ for at least one arc a in T, where $T = \{(u_1, u_2), (u_2, u_1), (u_1, u_3), (u_3, u_1), (u_2, u_3), (u_3, u_2)\}$. Assume not. Define $\bar{x}_1$ and $\bar{x}_2$, where

$$\bar{x}_i(a) = \begin{cases} \bar{x}(a) & \text{for } a \in A - T, \\ \bar{x}(a) + (-1)^i \varepsilon & \text{for } a \in \{(u_1, u_2), (u_2, u_3), (u_3, u_1)\}, \\ \bar{x}(a) - (-1)^i \varepsilon & \text{for } a \in \{(u_2, u_1), (u_3, u_2), (u_1, u_3)\} \end{cases}$$

for $i = 1, 2$. $\bar{x}_1, \bar{x}_2 \in LPE(G)$ for ε sufficiently small and $\bar{x} = (\bar{x}_1 + \bar{x}_2)/2$. Thus $\bar{x}_a = 0$ for at least one arc $a \in T$.

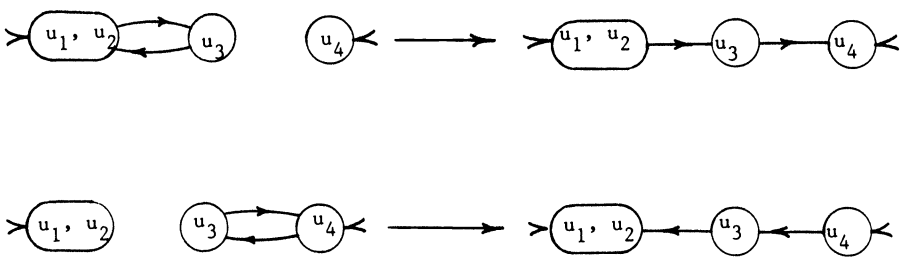


FIG. 2.4

To prove Proposition 2.6, without loss of generality, there are the following two distinct cases to be considered.

Case (2.6.i). We have that $\bar{x}(u_1, u_2) = 0$. In this case, $\bar{x}(u_3, u_2) = 1$, since $\bar{x}(u_1, u_2) + \bar{x}(u_3, u_2) \geqq 1$, and $\bar{x}$ is a vertex of $LPE(G)$. Define $\bar{x}_3$ and $\bar{x}_4$, where

$$\bar{x}_i(a) = \begin{cases} \bar{x}(a) & \text{for } a \in A - \{(u_2, u_1), (u_2, u_3), (u_3, u_1)\}, \\ \bar{x}(a) + (-1)^i\varepsilon & \text{for } a \in \{(u_2, u_3), (u_3, u_1)\}, \\ \bar{x}(a) - (-1)^i\varepsilon & \text{for } a = (u_2, u_1) \end{cases}$$

for $i = 3, 4$. Both $\bar{x}_3, \bar{x}_4 \in LPE(G)$ for ε sufficiently small if $\bar{x}(u_2, u_1), \bar{x}(u_2, u_3), \bar{x}(u_3, u_1) > 0$. Furthermore, $\bar{x} = (\bar{x}_3 + \bar{x}_4)/2$. Thus at least one of $\bar{x}(u_2, u_1), \bar{x}(u_2, u_3)$ and $\bar{x}(u_3, u_1)$ must be zero. We resolve each of the following three subcases separately.

Subcase (2.6.i.a). We have that $\bar{x}(u_2, u_1) = 0$. Delete (u_1, u_2) and (u_2, u_1) to obtain $G' = (V, A')$. x', the restriction of $\bar{x}$ to A', is a fractional vector of $LPE(G')$.

Subcase (2.6.i.b). We have that $\bar{x}(u_2, u_3) = 0$. In this case, $\bar{x}(u_2, u_1) = 1$ and $\bar{x}(u_3, u_1) = 0$. Form $G' = (V', A')$ by deleting node u_2 and all incident arcs. x', the restriction of $\bar{x}$ to A' with $x'(u_3, u_1) = 1$, is a fractional vertex of $LPE(G')$.

Subcase (2.6.i.c). We have that $\bar{x}(u_3, u_1) = 0$. If $\bar{x}(u_2, u_3) > 0$, we must have that $\bar{x}(u_2, u_3) + \bar{x}(u_2, u_1) = 1$. Form $G' = (V', A')$ by deleting node u_2 and all incident arcs; i.e., $V' = V - \{u_2\}, A' = A - \{(u_1, u_2), (u_2, u_1), (u_2, u_3), (u_3, u_2)\}$. See Fig. 2.5. Define x', where

$$x'(a) = \begin{cases} \bar{x}(a) & \text{for } a \in A' - \{(u_3, u_1)\}, \\ \bar{x}(u_2, u_1) & \text{for } a = (u_3, u_1). \end{cases}$$

Clearly, if $\bar{x}$ is a fractional vector, so is x'. We now show that x' is a vertex of $LPE(G')$. Assume not. Then we can write $x' = \sum \alpha_i x'_i + \delta$, where each x'_i is the incidence vector of a minimal equivalent subgraph of G', $\alpha_i \geqq 0$, $\sum \alpha_i = 1$, $\delta_a \geqq 0$ for all $a \in A'$. There are $\gamma \leqq \bar{x}(u_2, u_1)$ solutions x'_i with $x'_i(u_3, u_1) = 1$ (this refers to the set of such solutions with total weight γ, as described in the proof of Proposition 2.5). In each such case, define $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x'_i(a) & \text{for } a \in A' - \{(u_3, u_1)\}, \\ 1 & \text{for } a \in \{(u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

For $\bar{x}(u_2, u_1) - \gamma$ of the remaining solutions x'_i (this refers to a subset of the remaining solutions with total weight $\bar{x}(u_2, u_1) - \gamma$), perform the same transformation as above to get $\bar{x}_i$. For the remainder, define $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x'_i(a) & \text{for } a \in A', \\ 1 & \text{for } a \in \{(u_3, u_2), (u_2, u_3)\}, \\ 0 & \text{otherwise.} \end{cases}$$

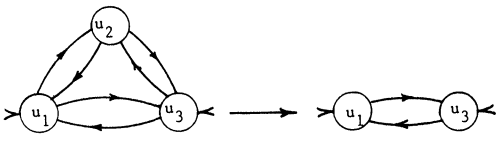


FIG. 2.5

Each $\bar{x}_i$ is the incidence vector of a minimal equivalent subgraph of G and $\bar{x} = \sum \alpha_i \bar{x}_i + \bar{\delta}$, where $\bar{\delta}_a = \delta_a$ for $a \in A'$, $\delta_a \geqq 0$ for $a \in A - A'$. Thus $\bar{x}$ is not a vertex of $LPE(G)$, contradicting our assumption. This implies that, if $\bar{x}$ is a vertex of $LPE(G)$, then x' is a vertex of $LPE(G')$, contradicting the minimality of G.

This eliminates Subcase (2.6.i.c). Since Subcases (2.6.i.a)–(2.6.i.c) have been eliminated, we cannot have that $\bar{x}(u_1, u_2) = 0$. Thus $\bar{x}(u_1, u_2) > 0$. By symmetry, we must have that $\bar{x}(u_2, u_1), \bar{x}(u_2, u_3), \bar{x}(u_3, u_2) > 0$.

Case (2.6.ii). We have that $\bar{x}(u_1, u_3) = 0$. Note that

$$\bar{x}(u_1, u_2) + \bar{x}(u_3, u_2) \geqq 1, \tag{2.3}$$

$$\bar{x}(u_2, u_1) + \bar{x}(u_2, u_3) \geqq 1. \tag{2.4}$$

Form $G' = (V, A')$, where $A' = A - \{(u_1, u_3), (u_3, u_1)\}$. Define the vector x', where

$$x'(a) = \begin{cases} \bar{x}(a) & \text{for } a \in A' - \{(u_3, u_2), (u_2, u_1)\}, \\ \bar{x}(u_2, u_1) + \bar{x}(u_3, u_1) & \text{for } a = (u_2, u_1), \\ \bar{x}(u_3, u_2) + \bar{x}(u_3, u_1) & \text{for } a = (u_3, u_2). \end{cases}$$

If x' is an integer vector, then $\bar{x}(u_1, u_2), \bar{x}(u_2, u_3), \bar{x}(u_2, u_1) + \bar{x}(u_3, u_1), \bar{x}(u_3, u_2) + \bar{x}(u_3, u_1) \in \{0, 1\}$. If any of the above is zero, we have Case (2.6.i). If all are 1, we have that $\bar{x}(u_1, u_2) = \bar{x}(u_2, u_3) = 1$ and

$$\bar{x}(u_2, u_1) + \bar{x}(u_3, u_1) = 1, \tag{2.5}$$

$$\bar{x}(u_3, u_2) + \bar{x}(u_3, u_1) = 1. \tag{2.6}$$

If $0 < \bar{x}(u_3, u_1) < 1$, define $\bar{x}_5$ and $\bar{x}_6$, where

$$\bar{x}_i(a) = \begin{cases} \bar{x}(a) & \text{for } a \in A - \{(u_2, u_1), (u_3, u_2), (u_3, u_1)\}, \\ \bar{x}(a) + (-1)^i \varepsilon & \text{for } a \in \{(u_2, u_1), (u_3, u_2)\}, \\ \bar{x}(a) - (-1)^i \varepsilon & \text{for } a = (u_3, u_1) \end{cases}$$

for $i = 5, 6$. Both $\bar{x}_5$ and $\bar{x}_6 \in LPE(G)$, and $\bar{x} = (\bar{x}_5 + \bar{x}_6)/2$. Thus $\bar{x}$ cannot be a vertex unless $\bar{x}(u_3, u_1) = 0$. Then, however, from (2.5) and (2.6), we have that $\bar{x}(u_2, u_1) = \bar{x}(u_3, u_2) = 1$; i.e., $\bar{x}$ is an integer vector. Thus we can assume that x' is a fractional vector. Now we show that x' is a vertex of $LPE(G')$.

Assume not. Then we can write $x' = \sum \alpha_i x'_i + \delta$, where $\alpha_i \geqq 0$, $\sum \alpha_i = 1$, $\delta_a \geqq 0$ for all $a \in A'$, and each x'_i is the incidence vector of a minimal equivalent subgraph of G'. All minimal equivalent subgraphs of G' have one of the following configurations, shown in Fig. 2.6. Define

$$\beta_1 = \{\textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.6(a)}\},$$

$$\beta_2 = \{\textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.6(b)}\},$$

$$\beta_3 = \{\textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.6(c)}\},$$

$$\beta_4 = \{\textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.6(d)}\},$$

$$\beta_5 = \{\textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.6(e)}\}.$$

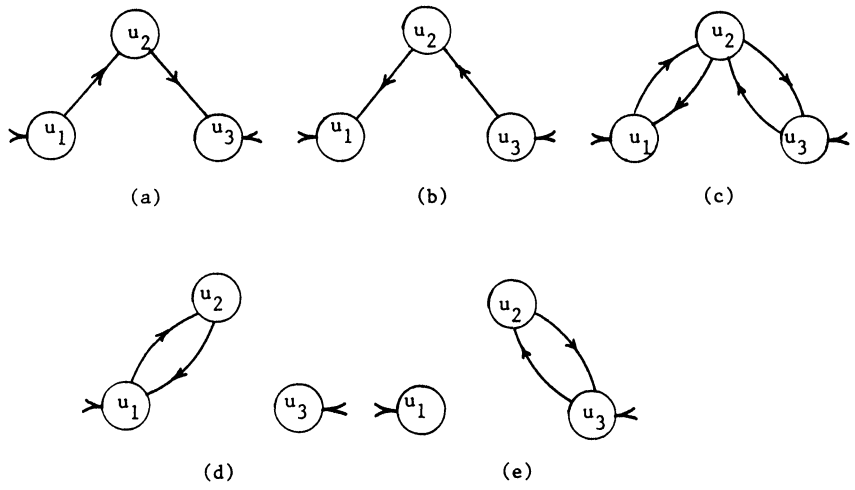


FIG. 2.6

With an argument similar to that in the proof of Proposition 2.5, we can assume that either $\beta_4 = 0$ or $\beta_5 = 0$. Note that $\beta_1 + \beta_2 + \beta_3 + \beta_4 + \beta_5 = 1$. From (2.3) and (2.4), we thus have that

$$\bar{x}(u_1, u_2) - (\beta_1 + \beta_3 + \beta_4) \geqq \beta_2 + \beta_5 - \bar{x}(u_3, u_2), \tag{2.7}$$

$$\bar{x}(u_2, u_3) - (\beta_1 + \beta_3 + \beta_5) \geqq \beta_2 + \beta_4 - \bar{x}(u_2, u_1). \tag{2.8}$$

If $\beta_3 \geqq \bar{x}(u_3, u_1)$, convert a subset of solutions x_i' with Fig. 2.6(c) of total weight $\bar{x}(u_3, u_1)$ to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_3, u_2), (u_2, u_1)\}, \\ 1 & \text{for } a = (u_3, u_1), \\ 0 & \text{otherwise.} \end{cases} \tag{2.9}$$

Each $\bar{x}_i$ is the incidence vector of a minimal equivalent subgraph of G and $\bar{x} = \sum \alpha_i \bar{x}_i + \bar{\delta}$, where $\bar{\delta}_a = \delta_a$ for $a \in A'$, $\delta_a \geqq 0$ for $a \in A - A'$. In this case, $\bar{x}$ is not a vertex of $LPE(G)$, contradicting our assumption.

Thus we have that $\beta_3 < \bar{x}(u_3, u_1)$. In this case, convert the solutions x_i' with Fig. 2.6(c) of total weight β_3 to $\bar{x}_i$, as in (2.9).

Without loss of generality, assume that $\bar{x}(u_2, u_1) \geqq \bar{x}(u_3, u_2)$. Convert a subset of solutions x_i' with Fig. 2.6(b) of total weight $\eta_1 = \max\{0, \beta_2 - \bar{x}(u_2, u_1)\}$ to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_1, u_2), (u_2, u_1), (u_2, u_3), (u_3, u_2)\}, \\ 1 & \text{for } a \in \{(u_1, u_2), (u_2, u_3), (u_3, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.7.

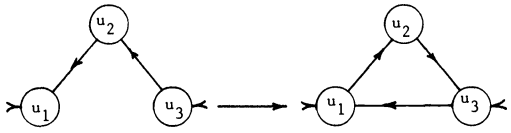


FIG. 2.7

Let $\eta_2 = \max\{0, \beta_2 - \bar{x}(u_3, u_2) - \eta_1\}$. If $\eta_2 > 0$, convert a subset of the remaining solutions x_i' with Fig. 2.6(b) of total weight η_2 to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_1, u_2), (u_3, u_2), (u_3, u_1)\}, \\ 1 & \text{for } a \in \{(u_1, u_2), (u_3, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.8.

If $\beta_5 = 0$, let $\eta_3 = \max\{0, \beta_2 + \beta_4 - \bar{x}(u_2, u_1) - \eta_1\}$. Clearly, $\eta_3 \leqq \beta_4$. If $\eta_3 > 0$, convert a subset of solutions x_i' with Fig. 2.6(d) of total weight η_3 to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_2, u_1), (u_2, u_3)\}, \\ 1 & \text{for } a = (u_2, u_3), \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.9.

If $\beta_4 = 0$, let $\eta_4 = \max\{0, \beta_2 + \beta_5 - \bar{x}(u_3, u_2) - \eta_1 - \eta_2\}$. Clearly, $\eta_4 \leqq \beta_5$. If $\eta_4 > 0$, convert a subset of solutions x_i' with Fig. 2.6(e) of total weight η_4 to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_3, u_2), (u_1, u_2)\}, \\ 1 & \text{for } a = (u_1, u_2), \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.10.

Each $\bar{x}_i$ is the incidence vector of a minimal equivalent subgraph of G. Furthermore, by (2.7),

$$\beta_1 + \beta_3 + \beta_4 + \eta_1 + \eta_2 + \eta_4 \leqq \bar{x}(u_1, u_2),$$

and, by (2.8),

$$\beta_1 + \beta_3 + \beta_5 + \eta_1 + \eta_3 \leqq \bar{x}(u_2, u_3),$$

$$\beta_3 + \eta_1 + \eta_2 \leqq \bar{x}(u_3, u_1);$$

since $\beta_3 < \bar{x}(u_3, u_1)$, $\beta_3 + \beta_2 \leqq \bar{x}(u_3, u_2) + \bar{x}(u_3, u_1)$ and $\max\{0, \beta_2 - \bar{x}(u_3, u_2)\} = \eta_1 + \eta_2$. Thus $\bar{x} = \sum \alpha_i \bar{x}_i + \bar{\delta}$, where $\bar{\delta}_a \geqq 0$ for all $a \in A$. This implies that $\bar{x}$ is not a

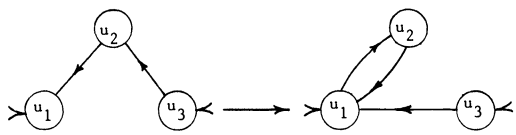


FIG. 2.8

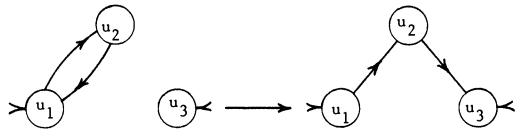


FIG. 2.9

vertex of $LPE(G)$, contradicting our assumption. This shows that, if $\bar{x}$ is a fractional vertex of $LPE(G)$, x' must be a fractional vertex of $LPE(G')$, contradicting the minimality of G.

Thus we have shown that Case (2.6.ii) can also be resolved. As a result, Fig. 2.2(d) cannot be present in G. □

PROPOSITION 2.7. *If G contains Fig. 2.2(e), there exists a minor G' of G such that $LPE(G')$ has a fractional vertex.*

Proof. With an argument similar to that in the proof of Proposition 2.6, we can show that at least one of the arcs

$$a\in\{(u_1,u_2),(u_2,u_1),(u_1,u_4),(u_4,u_1),(u_2,u_3),(u_3,u_2),(u_3,u_4),(u_4,u_3)\}$$

has $\bar{x}(a) = 0$. Without loss of generality, assume that $\bar{x}(u_1, u_2) = 0$. This implies that $\bar{x}(u_3, u_2) = 1$. Delete u_2 and all incident arcs to get $G' = (V', A')$, where $V' = V - \{u_2\}$ and $A' = A - \{(u_1, u_2), (u_2, u_1), (u_3, u_2), (u_2, u_3)\}$. Define x', where

$$x'(a)=\begin{cases}\bar{x}(a) & \text{for } a\in A'-\{(u_3,u_4),(u_4,u_1)\},\\ \bar{x}(a)+\bar{x}(u_2,u_1) & \text{for } a\in\{(u_3,u_4),(u_4,u_1)\}.\end{cases}$$

Note that, in vector $\bar{x}$, we have that

$$\bar{x}(u_2,u_1)+\bar{x}(u_2,u_3)\geqq 1, \tag{2.10}$$

$$\bar{x}(u_1,u_4)+\bar{x}(u_3,u_4)\geqq 1, \qquad \bar{x}(u_4,u_1)+\bar{x}(u_4,u_3)\geqq 1. \tag{2.11}$$

If x' is an integer vector, then $\bar{x}(u_1, u_4)$, $\bar{x}(u_4, u_3)$, $\bar{x}(u_4, u_1) + \bar{x}(u_2, u_1)$, $\bar{x}(u_3, u_4) + \bar{x}(u_2, u_1) \in \{0, 1\}$. If $0 < \bar{x}(u_2, u_1) < 1$, then $0 < \bar{x}(u_2, u_3)$, $\bar{x}(u_4, u_1)$, $\bar{x}(u_3, u_4) < 1$, and both $\bar{x}(u_1, u_4)$ and $\bar{x}(u_4, u_3)$ must be 1. Define the vectors $\bar{x}_7$ and $\bar{x}_8$, where

$$\bar{x}_i(a)=\begin{cases}\bar{x}(a) & \text{for } a\in A-\{(u_2,u_1),(u_2,u_3),(u_3,u_4),(u_4,u_1)\},\\ \bar{x}(a)+(-1)^i\varepsilon & \text{for } a\in\{(u_2,u_3),(u_3,u_4),(u_4,u_1)\},\\ \bar{x}(a)-(-1)^i\varepsilon & \text{for } a=(u_2,u_1)\end{cases}$$

for $i = 7, 8$. Both $\bar{x}_7, \bar{x}_8 \in LPE(G)$ for ε small enough and $\bar{x} = (\bar{x}_7 + \bar{x}_8)/2$. Thus we cannot have that $0 < \bar{x}(u_2, u_1) < 1$. This shows that, if x' is an integer vector, so is $\bar{x}$. Thus, if $\bar{x}$ is a fractional vector, so is x'. Now we show that x' is a vertex of $LPE(G')$.

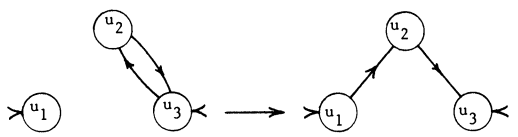


FIG. 2.10

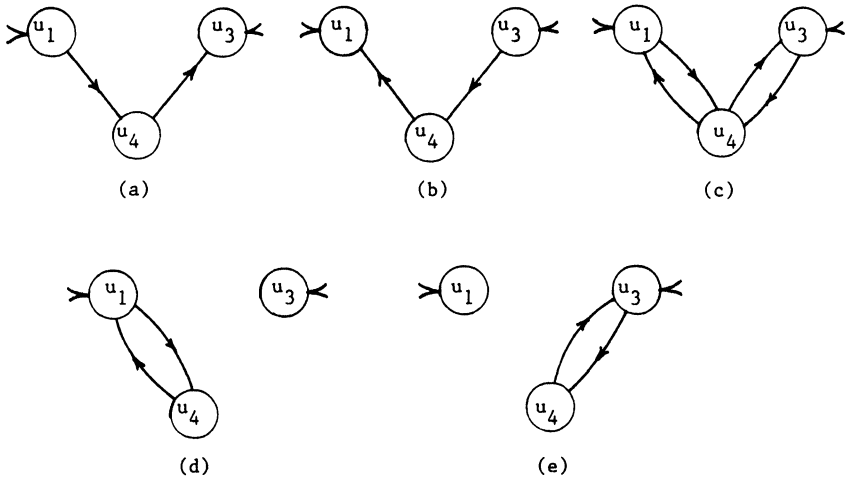


FIG. 2.11

Assume not. Then we can write $x' = \sum \alpha_i x'_i + \delta$, where $\alpha_i \geqq 0$, $\sum \alpha_i = 1$, $\delta_a \geqq 0$ for all $a \in A'$, and each x'_i is the incidence vector of a minimal equivalent subgraph of G'. All minimal equivalent subgraphs of G' have one of the following configurations in Fig. 2.11.

Define

$$\theta_1 = \{ \textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.11(a)} \},$$

$$\theta_2 = \{ \textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.11(b)} \},$$

$$\theta_3 = \{ \textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.11(c)} \},$$

$$\theta_4 = \{ \textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.11(d)} \},$$

$$\theta_5 = \{ \textstyle\sum \alpha_i \mid x'_i \text{ has Fig. 2.11(e)} \}.$$

As before, we can assume that either $\theta_4 = 0$ or $\theta_5 = 0$. Note that

$$\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 = 1.$$

From (2.11), we thus have that

$$\bar{x}(u_1, u_4) - (\theta_1 + \theta_3 + \theta_4) \geqq (\theta_2 + \theta_5) - \bar{x}(u_3, u_4), \tag{2.12}$$

$$\bar{x}(u_4, u_3) - (\theta_1 + \theta_3 + \theta_5) \geqq (\theta_2 + \theta_4) - \bar{x}(u_4, u_1). \tag{2.13}$$

If $\theta_3 \geqq \bar{x}(u_2, u_1)$, convert a subset of solutions x'_i with Fig. 2.11(c) of total weight $\bar{x}(u_2, u_1)$ to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x'_i(a) & \text{for } a \in A' - \{(u_3, u_4), (u_4, u_1)\}, \\ 1 & \text{for } a \in \{(u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.14}$$

This transformation is shown in Fig. 2.12.

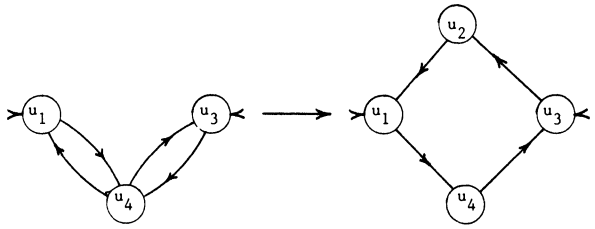


FIG. 2.12

For all other solutions x_i', form $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A', \\ 1 & \text{for } a \in \{(u_3, u_2), (u_2, u_3)\}, \\ 0 & \text{otherwise.} \end{cases}$$

Each $\bar{x}_i$ is the incidence vector of a minimal equivalent subgraph of G and $\bar{x} = \sum \alpha_i \bar{x}_i + \bar{\delta}$, where $\bar{\delta}_a \geqq 0$ for $a \in A$. In this case, $\bar{x}$ is not a vertex of $LPE(G)$, contradicting our assumption.

Thus we have that $\theta_3 < \bar{x}(u_2, u_1)$. In this case, convert all the solutions x_i' with Fig. 2.11(c) (of total weight θ_3) to $\bar{x}_i$ as in (2.14).

Without loss of generality, assume that $\bar{x}(u_4, u_1) \geqq \bar{x}(u_3, u_4)$. Convert a subset of solutions x_i' with Fig. 2.11(b) of total weight $\mu_1 = \max\{0, \theta_2 - \bar{x}(u_4, u_1)\}$ to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_4, u_3), (u_1, u_4), (u_3, u_4), (u_4, u_1)\}, \\ 1 & \text{for } a \in \{(u_1, u_4), (u_4, u_3), (u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.13.

Let $\mu_2 = \max\{0, \theta_2 - \bar{x}(u_3, u_4) - \mu_1\}$. If $\mu_2 > 0$, convert a subset of the remaining solutions x_i' with Fig. 2.11(b) of total weight μ_2 to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_3, u_4), (u_1, u_4)\}, \\ 1 & \text{for } a \in \{(u_1, u_4), (u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.14.

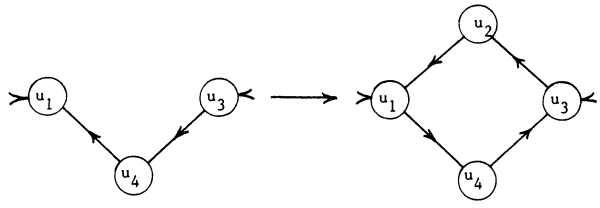


FIG. 2.13

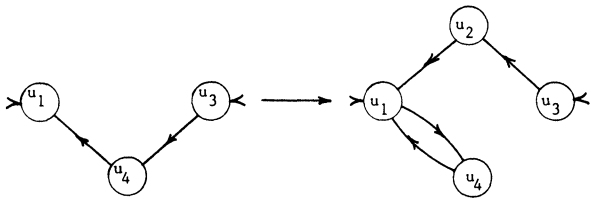


FIG. 2.14

If $\theta_5 = 0$, let $\mu_3 = \max\{0, \theta_2 + \theta_4 - \bar{x}(u_4, u_1) - \mu_1\}$. Clearly, $\mu_3 \leqq \theta_4$. If $\mu_3 > 0$, convert a subset of solutions x_i' with Fig. 2.11(d) of total weight μ_3 to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_4, u_1), (u_4, u_3)\}, \\ 1 & \text{for } a \in \{(u_4, u_3), (u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.15.

If $\theta_4 = 0$, let $\mu_4 = \max\{0, \theta_2 + \theta_5 - \bar{x}(u_3, u_4) - \mu_1 - \mu_2\}$. Clearly, $\mu_4 \leqq \theta_5$. If $\mu_4 > 0$, convert a subset of solutions x_i' with Fig. 2.11(e) of total weight μ_4 to $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A' - \{(u_1, u_4), (u_3, u_4)\}, \\ 1 & \text{for } a \in \{(u_1, u_4), (u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

This transformation is shown in Fig. 2.16.

Note that either μ_3 or μ_4 is zero, and that $\theta_3 + \mu_1 + \mu_2 + \mu_3 + \mu_4 \leqq \bar{x}(u_2, u_1)$. Let $\mu_5 = \bar{x}(u_2, u_1) - (\theta_3 + \mu_1 + \mu_2 + \mu_3 + \mu_4)$. For a subset of the remaining solutions x_i' of total weight μ_5, form $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A', \\ 1 & \text{for } a \in \{(u_3, u_2), (u_2, u_1)\}, \\ 0 & \text{otherwise.} \end{cases}$$

For the remaining solutions x_i' of total weight $1 - \bar{x}(u_2, u_1)$, form $\bar{x}_i$, where

$$\bar{x}_i(a) = \begin{cases} x_i'(a) & \text{for } a \in A', \\ 1 & \text{for } a \in \{(u_3, u_2), (u_2, u_3)\}, \\ 0 & \text{otherwise.} \end{cases}$$

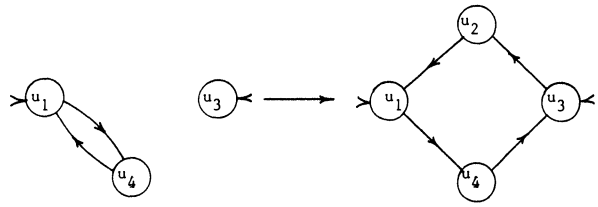


FIG. 2.15

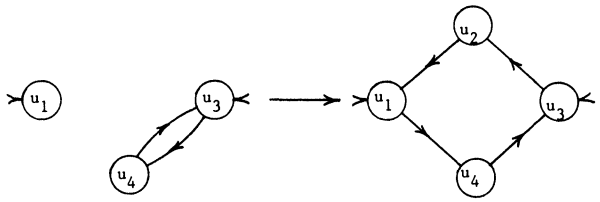


FIG. 2.16

Each $\bar{x}_i$ is the incidence vector of a minimal equivalent subgraph of G. Furthermore, by (2.12),

$$\theta_1+\theta_3+\theta_4+\mu_1+\mu_2+\mu_4 \leqq \bar{x}(u_1, u_4),$$

and, by (2.13),

$$\theta_1+\theta_3+\theta_5+\mu_1+\mu_3 \leqq \bar{x}(u_4, u_3).$$

Thus we have that $\bar{x} = \sum \alpha_i \bar{x}_i + \bar{\delta}$, where $\bar{\delta}_a \geqq 0$ for all $a \in A$. This implies that $\bar{x}$ is not a vertex of $LPE(G)$, contradicting our assumption. Thus, if $\bar{x}$ is a fractional vertex of $LPE(G)$, x' must be a fractional vertex of $LPE(G')$, contradicting the minimality of G.

This shows that Fig. 2.2(e) cannot be present in G.

Propositions 2.3–2.7 contradict the minimality of G in the proof of Theorem 2.1. This proves Theorem 2.1. □

Remark 2.1. In the case where $\tilde{G} = (V, \tilde{A})$ is any directed series-parallel graph that is strongly connected, it is a subgraph of a graph $G = (V, A)$ of the form assumed in Theorem 2.1. If C is a directed cut in G, then $\tilde{C} = C - \{A - \tilde{A}\}$ is a directed cut in $\tilde{G}$. All directed cuts in $\tilde{G}$ can be obtained in this manner, starting with all directed cuts in G. Thus $LPE(\tilde{G})$ is the intersection of $LPE(G)$ with the set $\{x \mid x_a = 0 \text{ for all } a \in A - \tilde{A}\}$. Since $LPE(G)$ has only integer vertices, so does $LPE(\tilde{G})$. This proves the following corollary.

COROLLARY 2.1. *For any strongly connected directed series-parallel graph G, we have that $LPE(G) = PE(G)$.*

From Fulkerson's results on blocking polyhedra [3], it thus follows.

COROLLARY 2.2. *For any strongly connected directed series-parallel graph G, we have that $LPC(G) = PC(G)$.*

3. Conclusions. In this paper, we give a complete characterization of $PE(G)$ and $PC(G)$ for connected series-parallel graphs. A complete characterization of $PE(G)$ is unlikely to be found for general graphs. Various families of facet-defining inequalities for this polyhedron are studied in [1]. As in the undirected case, the minimum directed cut can be found in polynomial time. We are still far from a complete description of $PC(G)$, however. In comparison to the undirected case, there is one interesting point. Undirected cuts and spanning trees form a blocking pair of clutters. Neither clutter, however has the weak max-flow min-cut property, even if the graph is a triangle. In contrast, the clutter of directed cuts has the weak max-flow min-cut property as long as the graph is series-parallel.

Acknowledgments. The author thanks the referees for a careful reading of the paper and for several valuable suggestions.

REFERENCES

[1] S. CHOPRA, *Polyhedra of the equivalent subgraph problem and some edge connectivity problems*, SIAM J. Discrete Math., 5 (1992), pp. 321–337.

[2] R. DUFFIN, *Topology of series-parallel networks*, J. Math. Anal. Appl., 10 (1965), pp. 303–318.

[3] D. R. FULKERSON, *Blocking and anti-blocking pairs of polyhedra*, Math. Programming, 1 (1971), pp. 168–194.

[4] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, San Francisco, 1979.

[5] C. L. LUCCHESI AND D. H. YOUNGER, *A min-max relation for directed graphs*, J. London Math. Soc. (2), 17 (1978), pp. 369–374.

[6] M. B. RICHEY, R. G. PARKER, AND R. L. RARDIN, *An efficiently solvable case of the minimum weight equivalent subgraph problem*, Networks, 15 (1985), pp. 217–228.

[7] P. D. SEYMOUR, *The matroids with the max-flow min-cut property*, J. Combin. Theory Ser. B, 23 (1977), pp. 189–222.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 491–496, November 1992
© 1992 Society for Industrial and Applied Mathematics
005

COVERING GRAPHS BY CYCLES*

GENGHUA FAN†

Abstract. Let G be a bridgeless graph with m edges and n vertices. It is proved that the edges of G can be covered by circuits whose total length is at most $m + (r/r-1)(n-1)$, where r is the minimum length of an even circuit (of G) of length at least 6 ($r = \infty$, if there is no such circuit). The proof suggests a polynomial-time algorithm for constructing such a cover.

Key words. covering, Eulerian subgraphs, graph algorithms, integer flows

AMS(MOS) subject classifications. 05C45, 05C70

1. Introduction. Graphs are finite, undirected, and may have loops or multiple edges. We denote the *vertex set* and *edge set* of a graph G by $V(G)$ and $E(G)$, respectively. Sometimes we identify a graph with its edge set. An edge is *contracted* if it is removed and if its ends are identified. A *component* of a graph G is a maximal connected subgraph of G; a *bridge* of G is an edge whose removal leaves a graph with more components than G. A graph without bridge is said to be *bridgeless*. A *cycle* is a graph in which every component is an Eulerian graph; a *trivial cycle* is one without edge. A *circuit* is a minimal nontrivial cycle. A *circuit decomposition* of a cycle is a set of edge-disjoint circuits whose union is the cycle. The *length* of a cycle is the number of the edges it contains. A cycle is *even* if its length is even.

A collection of cycles of a graph G *covers* G if each edge of G is in at least one of the cycles; such a collection is called a *cycle cover* of G. The *length* of a cycle cover is the sum of lengths of the cycles in the cover. The *shortest cycle cover problem* is to find a cycle cover of shortest length. Let $cc(G)$ denote the minimum length of a cycle cover of a bridgeless graph G. Itai et al. [5] conjectured that the problem of determining whether $cc(G) \leqq k$ is NP-complete. Nevertheless, polynomial-time algorithms have been developed to give good upper bounds for $cc(G)$. There are two types of upper bounds for $cc(G)$. One is in terms of $|E(G)|$, and the other in terms of both $|E(G)|$ and $|V(G)|$. We only consider the second type in this paper. If G is the graph obtained from a tree by replacing each edge by an odd number of new edges with the same ends, then $cc(G) = |E(G)| + |V(G)| - 1$. Itai and Rodeh [6] asked whether it is true that $cc(G) \leqq |E(G)| + |V(G)| - 1$ for every bridgeless graph G. In the same paper, they proved that $cc(G) \leqq |E(G)| + (2 \log |V(G)|)|V(G)|$. A breakthrough, relying on Jaeger's 8-flow theorem [7], was made by Itai et al. [5], who proved that $cc(G) \leqq |E(G)| + 6|V(G)| - 7$. This bound was improved to $|E(G)| + \frac{7}{3}(|V(G)| - 1)$ by Alon and Tarsi [1] and Bermond, Jackson, and Jaeger [2], and then to $|E(G)| + \frac{5}{4}(|V(G)| - 1)$ by Fraisse [4]. We prove the following theorem.

THEOREM 1.1. *Let G be a bridgeless graph; then $cc(G) \leqq |E(G)| + (r/r-1) \times (|V(G)| - 1)$, where r is the minimum length of an even circuit (of G) of length at least 6 ($r = \infty$, if there is no such circuit).*

An immediate consequence of Theorem 1.1 is the following theorem.

THEOREM 1.2. *Let G be a bridgeless graph; then $cc(G) \leqq |E(G)| + \frac{6}{5}(|V(G)| - 1)$.*

2. Proof of Theorem 1.1. The number of cycles used in a cycle cover is significant. It is known (see [7]) that a graph can be covered by two cycles if and only if it has a

* Received by the editors August 14, 1991; accepted for publication (in revised form) February 26, 1992.
† Department of Mathematics, Arizona State University, Tempe, Arizona 85287.

nowhere-zero 4-flow, and by three cycles if and only if it has a nowhere-zero 8-flow. The covers given in [6] use $1 + \lfloor \log n \rfloor$ cycles, and all the covers in [1], [2], [5] use four cycles, while the covers given by Fraisse [4] and by Theorem 1.1 need only three cycles. Our method is different from that used by Fraisse [4]. The proof of Theorem 1.1 involves Seymour's 6-flow theorem [9] and uses Younger's algorithm [11] for constructing a nowhere-zero 6-flow. A *flow* in a graph G with an orientation is an integer-valued function ϕ on $E(G)$ such that, for each vertex v, the sum of $\phi(e)$ over all edges e with head v is equal to the sum over all edges e with tail v. If there is a positive integer k such that $-k < \phi(e) < k$ for every $e \in E(G)$, then ϕ is called a k-flow of G. The *support* of a flow ϕ of G is defined by $S(\phi) = \{e \in E(G): \phi(e) \neq 0\}$; ϕ is called *nowhere-zero* if $S(\phi) = E(G)$. A fundamental result on flows is the following one due to Tutte [10]. It states that any flow can be reduced modulo m for any positive integer m.

LEMMA 2.1. *If G has a flow ϕ, then, for any integer $k > 0$, G has a k-flow ϕ' such that $\phi'(e) \equiv \phi(e) \pmod{k}$ for every $e \in E(G)$.*

A *circuit-chain* is a sequence $(C_1, C_2, \cdots, C_l)$ of edge-disjoint circuits such that $|V(C_i) \cap V(C_j)| = 1$ if $|i - j| = 1$ and $|V(C_i) \cap V(C_j)| = 0$ otherwise. By this definition, we have the following proposition.

PROPOSITION 2.2. *If $(C_1, C_2, \cdots, C_l)$ is a non-null circuit chain, then the number of distinct vertices of the chain is $\sum_{i=1}^{l} (|C_i| - 1) + 1$.*

In Younger's algorithm [11] for constructing a nowhere-zero 6-flow, if we choose, in each step, the cycle that contains the two ends of the "link pair" to be minimal, then the cycle is either a circuit chain or a single vertex. Therefore the following lemma essentially follows from Younger's algorithm [11].

LEMMA 2.3. *Every bridgeless graph G has a cycle F, each of whose components is a circuit chain, and a 3-flow ψ such that $E(G) \setminus E(F) \subseteq S(\psi)$.*

Remark 2.4. Lemma 2.3 can also be obtained without using Younger's algorithm. In that case, we first transfer G to a 3-edge connected graph G', then apply the method used by Seymour [9] to find a cycle F', each component of which is a circuit chain in G', and finally convert F' into the required cycle F in G.

DEFINITION 2.5. Let ϕ be a flow of G. Define $E_{\text{odd}}(\phi)$ ($E_{\text{even}}(\phi)$) to be the set of edges e with $\phi(e)$ odd (even). By the definition of a flow, $E_{\text{odd}}(\phi)$ is a cycle of G.

LEMMA 2.6. *Suppose that ϕ is a nowhere-zero 6-flow of G and $\{C_1, C_2, \cdots, C_t\}$ is a circuit decomposition of $E_{\text{odd}}(\phi)$. Then G has a 4-flow φ such that $E_{\text{even}}(\phi) \subseteq S(\varphi)$ and $|S(\varphi) \cap C_i| \geq \frac{7}{9}|C_i|$, $1 \leq i \leq t$.*

Proof. Let $C \in \{C_1, C_2, \cdots, C_t\}$ and let f be a 2-flow of G with $S(f) = C$. Set

$$A_\lambda = \{e \in C : \phi(e) + \lambda f(e) \equiv 0 \pmod{3}\}.$$

Since $\phi(e) \in \{\pm 1, \pm 3, \pm 5\}$ for $e \in C$, it is easy to check that $\{A_0, A_2, A_{-2}\}$ is a partition of C. Therefore we may choose $\theta \in \{0, 2, -2\}$ such that $|A_\theta| \geq \frac{1}{3}|C|$. Then

$$|\{e \in C : \phi(e) + \theta f(e) \not\equiv 0 \pmod{3}\}| \leq \tfrac{2}{3}|C|.$$

Since C is an arbitrary circuit in the circuit decomposition, we may apply the above arguments to each C_i, $1 \leq i \leq t$. Suppose that f_i is a 2-flow of G with $S(f_i) = C_i$. Then we have $\theta_i \in \{0, 2, -2\}$ such that

$$|\{e \in C_i : \phi(e) + \theta_i f_i(e) \not\equiv 0 \pmod{3}\}| \leq \tfrac{2}{3}|C_i|, \qquad 1 \leq i \leq t. \tag{2.1}$$

Let $\phi' = \phi + \sum_{i=1}^{t} \theta_i f_i$. Then ϕ' is a flow of G. Applying Lemma 2.1 with $k = 3$ to ϕ', we obtain a 3-flow ψ. We observe that, if $e \in E_{\text{even}}(\phi)$, then $f_i(e) = 0$ for all i, $1 \leq i \leq t$, and so $\phi'(e) = \phi(e) \in \{\pm 2, \pm 4\}$, which implies that $\psi(e)$ ($\equiv \phi'(e) \bmod 3$) is nonzero.

Thus $E_{\text{even}}(\phi) \subseteq S(\psi)$. For $E_{\text{odd}}(\phi)$ we obtain from (2.1) that

$$|S(\psi) \cap C_i| \leq \tfrac{2}{3}|C_i|, \qquad 1 \leq i \leq t. \tag{2.2}$$

In the definition of A_λ, if we replace $\phi(e)$ by $\psi(e)$ and mod 3 by mod 4, then, since $\psi(e) \in \{\pm 1, \pm 2\}$ for $e \in S(\psi) \cap C_i$, we obtain with $\mu_i \in \{1, -1, 2\}$ a partition of C such that

$$|\{e \in C_i : \psi(e) + \mu_i f_i(e) \equiv 0 \pmod 4\}| \leq \tfrac{1}{3}|S(\psi) \cap C_i|, \qquad 1 \leq i \leq t, \tag{2.3}$$

where we have used the fact that, for every $e \in C_i \backslash S(\psi)$,

$$\psi(e) + \mu_i f_i(e) = \mu_i f_i(e) \not\equiv 0 \pmod 4, \qquad 1 \leq i \leq t.$$

Let $\psi' = \psi + \sum_{i=1}^{t} \mu_i f_i$. By applying Lemma 2.1 with $k = 4$ to ψ', we obtain a 4-flow φ with $E_{\text{even}}(\phi) \subseteq S(\varphi)$, and, by (2.3),

$$|\{e \in C_i : \varphi(e) = 0\}| \leq \tfrac{1}{3}|S(\psi) \cap C_i|, \qquad 1 \leq i \leq t.$$

Using (2.2),

$$|\{e \in C_i : \varphi(e) = 0\}| \leq \tfrac{2}{9}|C_i|, \qquad 1 \leq i \leq t.$$

Therefore

$$|S(\varphi) \cap C_i| \geq \tfrac{7}{9}|C_i|, \qquad 1 \leq i \leq t,$$

which ends the proof of Lemma 2.6. □

The proof of the proposition below is easy, and the details can be found in [3].

PROPOSITION 2.7. *If G has a nowhere-zero 4-flow, then G can be covered by two cycles Z_1 and Z_2 with $|Z_1| + |Z_2| \leq |E(G)| + |V(G)| - 1$.*

Proof of Theorem 1.1. By Lemma 2.3, G has a cycle F, each of whose components is a circuit chain, and a 3-flow ψ such that $E(G)\backslash E(F) \subseteq S(\psi)$. Let f be a 2-flow of G with $S(f) = F$ and define $\phi = 2\psi + f$. Then ϕ is a nowhere-zero 6-flow of G with $E_{\text{odd}}(\phi) = F$. Decompose F into edge-disjoint circuits $C_1, C_2, \cdots, C_t$. By Lemma 2.6, we have a 4-flow φ such that $E_{\text{even}}(\phi) \subseteq S(\varphi)$ and $|S(\varphi) \cap C_i| \geq \frac{7}{9}|C_i|$, $1 \leq i \leq t$. Since $|S(\varphi) \cap C_i|$ is an integer, we have, for $|C_i| \leq 4$, that $|S(\varphi) \cap C_i| = |C_i|$, which implies that $C_i \subseteq S(\varphi)$. Therefore, if we set

$$F' = \{C_i : |C_i| \geq 5, 1 \leq i \leq t\},$$

then $E(G)\backslash E(F') \subseteq S(\varphi)$. Denote by G^* the graph obtained by contracting all the edges in F'. It is clear that the restriction of φ to $E(G^*)$ is a nowhere-zero 4-flow of G^*. By Proposition 2.7, G^* can be covered by two cycles Z_1^* and Z_2^* with

$$|Z_1^*| + |Z_2^*| \leq |E(G^*)| + |V(G^*)| - 1. \tag{2.4}$$

By adding edges in F' to Z_j^*, we may expand Z_j^* into a cycle Z_j of G, $j = 1, 2$. For any $C \in F'$, if $|Z_j \cap C| > \frac{1}{2}|C|$, then $|(Z_j \oplus C) \cap C| < \frac{1}{2}|C|$, where $Z_j \oplus C = (Z_j \cup C)\backslash (Z_j \cap C)$. We may therefore assume that Z_1 and Z_2 have been chosen such that, for every $C \in F'$,

$$|Z_j \cap C| \leq \tfrac{1}{2}|C|, \qquad j = 1, 2. \tag{2.5}$$

Set

$$P = \{C \in F' : |C| \text{ is odd}\} \quad \text{and} \quad Q = \{C \in F' : |C| \text{ is even}\}.$$

Let p and q be the numbers of circuits of P and Q, respectively. For any $C \in P$, by (2.5) $|Z_j \cap C| \leqq \frac{1}{2}(|C| - 1), j = 1, 2$, and so $|Z_1 \cap C| + |Z_2 \cap C| \leqq |C| - 1$. It follows that

$$|Z_1 \cap F'| + |Z_2 \cap F'| \leqq |E(F')| - p. \tag{2.6}$$

Since $|Z_1| + |Z_2| = |Z_1^*| + |Z_2^*| + |Z_1 \cap F'| + |Z_2 \cap F'|$, it follows from (2.4) and (2.6) that

$$\begin{aligned} |Z_1| + |Z_2| &\leqq |E(G^*) + |V(G^*)| + |E(F')| - 1 - p \\ &= |E(G)| + |V(G^*)| - 1 - p. \end{aligned}$$

Clearly, $\{Z_1, Z_2, F'\}$ is a cycle cover of G, and so

$$cc(G) \leqq |Z_1| + |Z_2| + |E(F')| \leqq |E(G)| + |V(G^*)| + |E(F')| - 1 - p. \tag{2.7}$$

We now give an upper bound for $|V(G^*)| + |E(F')|$. We may assume that G is connected. So G^* is connected. Let T^* be a spanning tree of G^*. Consider the subgraph H of G induced by $E(T^*) \cup E(F')$. Since each vertex of T^* corresponds to a component of F', the only circuits in H are those in F'. If we remove one edge from each circuit of F', then the remainder of H is a spanning tree of G. From the fact that the total number of circuits in F' is $p + q$, it follows that

$$|E(H)| - (p + q) \leqq |V(G)| - 1.$$

Since $|E(H)| = |E(T^*)| + |E(F')|$ and $|E(T^*)| = |V(G^*)| - 1$, we obtain

$$|V(G^*)| + |E(F')| \leqq |V(G)| + p + q.$$

Substituting into (2.7), we obtain

$$cc(G) \leqq |E(G)| + |V(G)| - 1 + q. \tag{2.8}$$

If $Q = \varnothing$, then $q = 0$, and the result follows from (2.8). Suppose that $Q \neq \varnothing$. For convenience, regard Q as a subgraph of G. Then each component of Q is a non-null circuit chain. It follows from Proposition 2.2 that

$$|V(Q)| \geqq \sum_{C \in Q} (|C| - 1) + 1.$$

By the definition of r and Q, $|C| \geqq r$ for all $C \in Q$, and hence

$$|V(Q)| \geqq q(r - 1) + 1, \tag{2.9}$$

which gives

$$q \leqq \frac{1}{r-1}(|V(Q)| - 1) \leqq \frac{1}{r-1}(|V(G)| - 1). \tag{2.10}$$

Applying (2.10) to (2.8) yields the fact that $cc(G) \leqq |E(G)| + (r/r - 1)(|V(G)| - 1)$ and completes our proof. □

3. Other results. In Lemma 2.3, if f is a 2-flow of G with $S(f) = F$, then $\phi = 2\psi + f$ is a nowhere-zero 6-flow of G with $E_{\text{odd}}(\phi) = F$. We may start with any given connected cycle to build up the cycle $F\ (= E_{\text{odd}}(\phi))$. In particular, we may start with a circuit. This gives us the following strengthening of Lemma 2.3.

LEMMA 3.1. *Let C be any circuit of a bridgeless graph G; then G has a nowhere-zero 6-flow ϕ such that each component of $E_{\text{odd}}(\phi)$ is a circuit chain and C is one of the components.*

If we apply Lemma 3.1 to the proof of Theorem 1.1, then we obtain a better upper bound for the number q in (2.10) as follows. If the given circuit C (in Lemma 3.1) is in Q, then, instead of (2.9), we have that $|V(Q)| \geqq (q-1)(r-1) + |V(C)|$, which gives

$$q \leqq \frac{1}{r-1}(|V(Q)| - |V(C)|) + 1 \leqq \frac{1}{r-1}(|V(G)| - |V(C)|) + 1;$$

otherwise, $C \in P$ and $|V(Q)| \leqq |V(G)| - |V(C)|$, and, from (2.9),

$$q \leqq \frac{1}{r-1}(|V(G)| - |V(C)| - 1),$$

where we assume that $Q \neq \varnothing$. In either case, $q \leqq (1/r - 1)(|V(G)| - |V(C)|) + 1$. This, together with (2.8), yields the following result.

THEOREM 3.2. *Let C be any circuit of a bridgeless graph G; then*

$$cc(G) \leqq |E(G)| + |V(G)| + \frac{1}{r-1}(|V(G)| - |V(C)|),$$

where r is the minimum length of an even circuit (of G) of length at least 6 ($r = \infty$, if there is no such circuit).

An immediate consequence of the above theorem is Corollary 3.3.

COROLLARY 3.3. *Suppose that G is a bridgeless graph. Let l be the maximum length of a circuit of G. Then*

$$cc(G) \leqq |E(G)| + |V(G)| + \frac{1}{r-1}(|V(G)| - l),$$

where r is the minimum length of an even circuit (of G) of length at least 6 ($r = \infty$, if there is no such circuit).

Another possible approach to strengthen Theorem 1.1 is to improve inequality (2.9). If each component of $E_{\text{odd}}(\phi)$ is a circuit, then we would have, instead of (2.9), that $|V(Q)| \geqq qr$, and it would follow from (2.8) that

$$(3.1) \qquad cc(G) \leqq |E(G)| + \frac{r+1}{r}|V(G)| - 1 \leqq |E(G)| + \frac{7}{6}|V(G)| - 1.$$

This leads us to the following question.

Problem 3.4. Does every bridgeless graph have a nowhere-zero 6-flow ϕ such that each component of $E_{\text{odd}}(\phi)$ is a circuit?

An affirmative answer to the above problem was given by Seymour [9, (3.2)] to 3-connected simple graphs. The condition "simple" can be dropped if we combine Seymour's method [9] with that used by Jaeger [8, Thm. 4.4]. In fact, by combination of these two methods, we can obtain the following result.

THEOREM 3.5. *If G is a graph such that $G - e$ is 2-connected for every edge e, then, for any circuit C of G, G has a 6-flow ϕ such that $C \subseteq E_{\text{odd}}(\phi)$, and each component of $E_{\text{odd}}(\phi)$ is a circuit.*

Combining the proofs of Theorem 3.2 and inequality (3.1), we have the following consequence of Theorem 3.5.

COROLLARY 3.6. *Suppose that G is a 3-connected graph. Let l be the maximum length of a circuit of G. Then*

$$cc(G) \leqq |E(G)| + |V(G)| + \frac{1}{r}(|V(G)| - l),$$

where r is the minimum length of an even circuit (of G) of length at least 6 ($r = \infty$, *if there is no such circuit*).

4. Time bounds for finding cycle covers. Throughout this section, m and n denote the numbers of edges and vertices of a graph, respectively. The cycle cover given by Itai and Rodeh [6] uses $1 + \lfloor \log n \rfloor$ cycles. The number of cycles is dramatically reduced to 4 by Itai et al. [5]. The covers given by Alon and Tarsi [1] and Bermond, Jackson, and Jaeger [2] also require four cycles. These four cycles can be found in $O(m + n^2)$ time (see [1]). (It is possible that $m \gg n$, since G may have multiple edges.) The covers given by Fraisse [4] and by Theorem 1.1 need only three cycles. Fraisse did not give a time bound for his construction, which starts with three cycles covering G. Since it requires $O(m \cdot n)$ to find such three cycles (see [5]), Fraisse's construction needs at least $O(m \cdot n)$ running time. The cover in Theorem 1.1 can be constructed in $O(m \cdot n)$. We sketch an evaluation of the complexity as follows.

(1) Younger's algorithm [11] finds the needed 6-flow in $O(m \cdot n)$ time.

(2) In the proof of Lemma 2.6, θ_i and μ_i, $1 \leqq i \leqq t$, can be determined in $O(m)$ time.

(3) Applying Lemma 2.1 to obtain the needed k-flows can be done in $O(m \cdot n)$ time (see [11, § 3]).

By the statements above, the 4-flow φ in Lemma 2.6 can be constructed in $O(m \cdot n)$ time. Clearly, finding the two cycles Z_1 and Z_2 from φ can be done in $O(m \cdot n)$. Therefore the time bound for finding the cover in Theorem 1.1 is $O(m \cdot n)$, as claimed.

REFERENCES

[1] N. ALON AND M. TARSI, *Covering multigraphs by simple circuits*, SIAM J. Alg. Discrete Meth., 6 (1985), pp. 345–350.

[2] J. C. BERMOND, B. JACKSON, AND F. JAEGER, *Shortest covering of graphs with cycles*, J. Combin. Theory Ser. B, 35 (1983), pp. 297–308.

[3] G. FAN, *Integer flows and cycle covers*, J. Combin. Theory Ser. B, 54 (1992), pp. 113–122.

[4] P. FRAISSE, *Cycle covering in bridgeless graphs*, J. Combin. Theory Ser. B, 39 (1985), pp. 146–152.

[5] A. ITAI, R. J. LIPTON, C. H. PAPADIMITRIOU, AND M. RODEH, *Covering graphs with simple circuits*, SIAM J. Comput., 10 (1981), pp. 746–750.

[6] A. ITAI AND M. RODEH, *Covering a graph by circuits*, in Automata, Languages and Programming, Lecture Notes in Computer Science 62, Springer-Verlag, Berlin, 1978, pp. 289–299.

[7] F. JAEGER, *Flows and generalized coloring theorems in graphs*, J. Combin. Theory Ser. B, 26 (1979), pp. 205–216.

[8] ———, *Nowhere-zero flow problems*, in Selected Topics in Graph Theory 3, L. W. Beineke and R. J. Wilson, eds., Academic Press, London, 1988, pp. 71–95.

[9] P. D. SEYMOUR, *Nowhere-zero 6-flows*, J. Combin. Theory Ser. B, 30 (1981), pp. 130–135.

[10] W. T. TUTTE, *A class of abelian groups*, Canadian J. Math, 8 (1956), pp. 13–28.

[11] D. H. YOUNGER, *Integer flows*, J. Graph Theory, 7 (1983), pp. 349–357.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 497–499, November 1992
© 1992 Society for Industrial and Applied Mathematics
006

HOW TO GUESS A GENERATING FUNCTION*

SEYOUM GETU†, LOUIS W. SHAPIRO†, WEN-JIN WOAN†, AND LEON C. WOODSON‡

Abstract. In this note, a new method for taking the first few terms of a sequence and making an educated guess as to the generating function of the sequence is described. The method involves a matrix factorization into lower triangular, diagonal, and upper triangular matrices (the LDU decomposition), generating functions, and solving a first-order differential equation.

Key words. generating function, Hankel matrix, differential equations, preferential arrangement, Schröder numbers

AMS(MOS) subject classification. 05A15

Analyzing a sequence. Suppose that we have determined the first few terms of a sequence and would like to know more about it. The sequence 1, 3, 10, 37, 151, 674, 3263, 17007, 94824, $\cdots$ will be used to illustrate. From this limited information, we would like to guess either a generating function or a recursion relation. Suppose that looking at differences, Sloane's handbook [S], or looking for a recursion, have not helped in identifying the sequence. Here is a new technique that often provides some insight. Start by forming a Hankel matrix from the sequence. The illustrative sequence yields that

$$H=\begin{bmatrix} 1 & 3 & 10 & 37 & 151 & \\ 3 & 10 & 37 & 151 & 674 & \\ 10 & 37 & 151 & 674 & 3263 & \vdots \\ 37 & 151 & 674 & 3263 & 17007 & \\ 151 & 674 & 3263 & 17007 & 94824 & \\ & & \cdots & & & \end{bmatrix},$$

and Gauss elimination is used to find the LDU decomposition of H as follows:

$$L=\begin{bmatrix} 1 & & & & 0 \\ 3 & 1 & & & \vdots \\ 10 & 7 & 1 & & \\ 37 & 40 & 12 & 1 & \\ 151 & 221 & 103 & 18 & 1 \\ & & \cdots & & \end{bmatrix}, \quad D=\begin{bmatrix} 1 & & & & 0 \\ & 1 & & & \\ & & 2! & & \\ & & & 3! & \vdots \\ 0 & & & & 4! \\ & & & \cdots & \end{bmatrix},$$

and $U = L^T$, the transpose of L.

Obviously, L and the original sequence convey the same information, but often L is more tractable.

Let $C_0(x)$ be the generating function for the first column, $C_1(x)$ the generating function for the second, and find $f(x)$ such that $C_0(x)f(x) = C_1(x)$.

* Received by the editors November 5, 1990; accepted for publication June 10, 1991. This research was supported by National Science Foundation grant R11-89-12667.

† Department of Mathematics, Howard University, Washington, D.C. 20059.

‡ Department of Mathematics, Morgan State University, Baltimore, Maryland 21239.

In this example, exponential generating functions work, and we solve

$$\left(1+3x+10\frac{x^2}{2!}+37\frac{x^3}{3!}+\cdots\right)f(x)=x+7\frac{x^2}{2!}+40\frac{x^3}{3!}+221\frac{x^4}{4!}+\cdots$$

to obtain $f(x) = x + x^2/2! + x^3/3! + \cdots$, leading to a reasonable guess that $f(x) = e^x - 1$.

The next step is a comparison of the first two columns. Here it seems that $C_{n+1,0} = 3C_{n,0} + C_{n,1}$, where $L = (C_{n,k})_{n,k\geq 0}$. This leads to

$$C_0'(x)=3C_0(x)+C_1(x)=3C_0(x)+f(x)C_0(x)$$

$$=3C_0(x)+(e^x-1)C_0(x).$$

The solution of this elementary differential equation is

$$C_0(x)=e^{e^x+2x-1} \quad \text{since } C_0(0)=1.$$

We have guessed our generating function. If we define $C_0(x) = \sum_{n=0}^{\infty} P_n(x^n/n!)$, then differentiation yields

$$C_0'(x)=C_0(x)(e^x+2), \quad \text{so}$$

$$p_{n+1}=2p_n+\sum_{l=0}^{n}\binom{n}{l}p_l.$$

We can even give a combinatorial interpretation for $e^{2x}e^{3^x-1}$. Since e^{e^x-1} generates the Bell numbers, we can take a set $[n]$, color some elements red and some others green, then partition the rest into disjoint nonempty blocks $B_1, B_2, \cdots, B_k$. Let G be the green elements, R the red elements, and let $B_1 \cup G, B_2 \cup G, \cdots, B_k \cup G$ be the atoms of a sublattice. The $\hat{O}$ element for this sublattice is G, while $\hat{1}$ is $[n] - R$. This process can be easily reversed. Thus the sequence 1, 3, 10, 37, 151, $\cdots$ could be the number of Boolean sublattices of the Boolean lattice of subsets of $[n]$.

We have seen one example where we started with the first nine terms of a sequence, formed the Hankel matrix, row-reduced to obtain the LDU decomposition, found a recurrence in L, guessed f, and solved a first-order differential equation. From this, we found the generating function, a recursion relation, and a combinatorial interpretation.

Here are two examples, set as exercises, to illustrate the technique.

(i) We use the sequence 1, 1, 3, 13, 75, 541, 4683, $\cdots$. We obtain that

$$L=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \\ 1 & 1 & 0 & 0 & 0 & \\ 3 & 5 & 1 & 0 & 0 & \vdots \\ 13 & 31 & 12 & 1 & 0 & \\ 75 & 233 & 133 & 22 & 1 & \\ & & \cdots & & & \end{bmatrix},$$

$$f(x)=x+3x^2/2!+13x^3/3!+75x^4/4!+\cdots.$$

A reasonable guess is that $C_0(x) - 1 = f(x)$. The differential equation then becomes

$$C_0'(x)=C_0(x)+2C_0(x)f(x),$$

so $C_0(x) = 1/(2 - e^x)$. These numbers can arise as the number of preferential arrangements.

(ii) We use the sequence 1, 2, 6, 22, 90, 394, 1806, $\cdots$. In this case, ordinary generating functions work better, as shown below:

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 & \\ 2 & 1 & 0 & 0 & \vdots \\ 6 & 5 & 1 & 0 & \\ 22 & 23 & 8 & 1 & \\ & \cdots & & & \end{bmatrix},$$

$$f = x + 3x^2 + 11x^3 + 45x^4 + \cdots = \frac{C_0(x) - 1}{2},$$

$$C_{n+1,0} = 2C_{n,0} + 2C_{n,1} \quad \text{so}$$

$$C_0(x) = 1 + 2x[C_0(x) + C_1(x)] = 1 + 2x\left[C_0(x) + C_0(x)\left(\frac{C_0(x) - 1}{2}\right)\right].$$

Now we solve a quadratic equation instead of a differential equation and obtain that

$$C_0(x) = \frac{1 - x - \sqrt{1 - 6x + x^2}}{2x},$$

which generates the (double) Schröder numbers.

If we have the stronger condition that the $C_n(x) = C_0(x)(f(x))^n$ or $C_n(x) = C_0(x)(f(x)^n)/n!$ for all n, then we obtain a group structure for the lower triangular matrices leading variously to the umbral calculus [R], paths and continued fraction expansions [F], [GJ], and combinatorics of orthogonal polynomials [V]. A brief introduction is given in [GSWW].

For a researcher discovering integer sequences, this technique can be very useful at the initial work stage. It is hard to judge how many sequences on which this would work, but the list does include factorials, derangement numbers, telephone numbers (the number of elements in S_n such that $x^2 = e$), Bernoulli numbers, number of permutations with no double descents, Bell numbers, both even and odd factorials, Euler numbers, numbers of preferential arrangements, secant numbers, Catalan numbers, Motzkin numbers, Schröder numbers (little and big), Delannoy numbers, central binomial coefficients, directed animals (single source), central trinomial coefficients, and some polynomial sequences such as Chebyshev Legendre and Hermite sequences.

REFERENCES

[C] L. COMTET, *Advanced Combinatorics*, Reidel, Boston, MA, 1974.

[F] P. FLAJOLET, *Combinatorial aspects of continued fractions*, Discrete Math., 32 (1980), pp. 125–161.

[GJ] I. GOULDEN AND D. JACKSON, *Combinatorial Enumeration*, John Wiley, New York, 1983.

[GSWW] S. GETU, L. SHAPIRO, W-J WOAN, AND L. WOODSON, *The Riordan group*, Discrete Appl. Math., 34 (1991), pp. 229–239.

[R] S. ROMAN, *The Umbral Calculus*, Academic Press, New York, 1984.

[S] N. J. A. SLOANE, *A Handbook of Integer Sequences*, Academic Press, New York, 1973.

[V] G. VIENNOT, *Une théorie combinatoire des polynômes orthogonaux généraux*, Notes, Université du Quebec à Montréal, 1983.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 500–523, November 1992
© 1992 Society for Industrial and Applied Mathematics
007

SOME RESULTS ON LIU'S CONJECTURE*

KWANG S. HONG† AND JOSEPH Y.-T. LEUNG‡

Abstract. In 1972, Liu claimed that for any task systems to be scheduled on $m \geqq 1$ identical processors, the ratio of the optimal nonpreemptive schedule length versus the optimal preemptive schedule length is bounded above by $2m/(m+1)$. Furthermore, Liu showed that the bound is the best possible by giving a task system achieving the ratio. His upper bound proof was later found to be incorrect, and his claim has since remained a conjecture. In this paper, it is shown that Liu's bound is valid for the unit execution time (UET) and tree-structured task systems. For two processors, it is shown that some other classes of task systems satisfy the 4/3 bound. Other results comparing the lengths of optimal list, nonpreemptive schedules, and preemptive schedules are also given.

Key words. schedule length, list scheduling, nonpreemptive scheduling, preemptive scheduling, task system, multiprocessor system

AMS(MOS) subject classifications. 90B35, 68R05

1. Introduction. In 1972, Liu [9] claimed that for any task systems to be scheduled on $m \geqq 1$ identical processors, the ratio of the optimal nonpreemptive schedule length versus the optimal preemptive schedule length is bounded above by $2m/(m+1)$. Furthermore, he showed that the bound is the best possible by giving a task system achieving the ratio. His upper bound proof was later found to be incorrect, and his claim has since remained a conjecture. In this paper, we show that Liu's bound is valid for the unit execution time (UET) and tree-structured task systems. For two processors, we show that some other classes of task systems satisfy the 4/3 bound. Other results comparing the lengths of optimal list, nonpreemptive, and preemptive schedules are also given.

The problem to be considered involves scheduling a task system on a set $P = \{P_1, P_2, \cdots, P_m\}$ of $m \geqq 1$ identical processors so as to minimize the schedule length. A task system $\tau = (TS, G, e)$ consists of (1) a set of n tasks $TS = \{T_1, T_2, \cdots, T_n\}$, (2) a directed acyclic graph $G = (TS, A)$ describing the precedence constraints between the tasks in TS such that $(T_i, T_j) \in A$ implies that T_i must be completed before T_j can start, and (3) a function $e : TS \rightarrow \{\text{nonnegative integers}\}$ giving the execution time of the tasks, where $e(T_i)$ denotes the execution time of T_i. If S is a schedule for τ on P, the length of S, denoted by ω_S, is the total time taken to execute all tasks in τ. We call a schedule with the minimum schedule length an *optimal* schedule.

Preemptive, nonpreemptive, and list are three scheduling disciplines commonly studied in the literature [1], [4]. For a given task system τ to be scheduled on $m \geqq 1$ identical processors, let ω_P, ω_N, and ω_L denote the lengths of optimal preemptive, nonpreemptive, and list schedules, respectively. Trivially, we have that $\omega_P \leqq \omega_N \leqq \omega_L$. It is known [1], [4] that $\omega_L/\omega_N \leqq \omega_L/\omega_P \leqq 2 - 1/m$. Furthermore, there are task systems for which both ratios approach $2 - 1/m$ asymptotically [1], [4]. Liu [9] gave a task system for which $\omega_N/\omega_P = 2m/(m+1)$. It consists of a set of $m+1$ independent tasks, each of which has execution time m. As mentioned before, it has been conjectured that $2m/(m+1)$ is also an upper bound. In this paper, we show that Liu's bound is valid

* Received by the editors January 18, 1990; accepted for publication (in revised form) August 7, 1991. This research was supported in part by Office of Naval Research grant N0001-1-87-K-0833 and in part by a grant from Texas Instruments, Inc.

† Computer Science Program, University of Texas at Dallas, Richardson, Texas 75083.

‡ Department of Computer Science and Engineering, University of Nebraska–Lincoln, Lincoln, Nebraska 68588-0115.

for UET and tree-structured task systems. Furthermore, we show that $\omega_L = \omega_N$, and hence $\omega_L/\omega_P = \omega_N/\omega_P \leqq 2m/(m+1)$ for these two classes. Liu's task system shows that the bound is the best possible for these two classes. For arbitrary precedence constraints and arbitrary execution times, it is not known whether Liu's bound is valid, even for $m = 2$. We show, for two-processor systems, that the 4/3 bound is valid for two special classes. The first class allows arbitrary execution times but restricts the precedence constraints to be *simple* (defined later in the paper). The second class allows arbitrary precedence constraints but restricts the execution times to be drawn from the set $\{1, 2, 3, 4\}$.

A UET task system is one in which each task has execution time 1 unit. Finding optimal nonpreemptive schedules for UET task systems is NP-hard if m is arbitrary [11]. Coffman and Graham [2] gave an optimal algorithm for two processors. Lam and Sethi [8] used the Coffman–Graham Algorithm as a heuristic and showed that, for any UET task systems, $\omega_{CG}/\omega_N \leqq 2 - 2/m$, where ω_{CG} is the length of the schedule produced by the Coffman–Graham Algorithm. Furthermore, they showed that the bound can be approached asymptotically. We follow the analysis of Lam and Sethi [8] to show that $\omega_{CG}/\omega_P \leqq 2m/(m+1)$.

Let $\tau = (TS, G, e)$ be a task system. A *chain* from a task T to a task T' is a sequence of tasks $T_1, T_2, \cdots, T_l$ such that $T = T_1$, $T' = T_l$, and (T_i, T_{i+1}) is an arc in G for all $1 \leqq i < l$. Task $T(T')$ is a *predecessor* (*successor*) of task $T'(T)$ if there is a chain from T to T'. Furthermore, if $l = 2$, then $T(T')$ is an *immediate* predecessor (successor) of $T'(T)$. The *length* of a chain is the total execution time of the tasks on the chain. *Initial* tasks are those with no predecessors, and *terminal* tasks are those with no successors. The *level* of a task T is the length of the longest chain from T to a terminal task. The precedence graph G is an *in-forest* (*out-forest*) if each task has at most one immediate successor (predecessor). An *in-tree* (*out-tree*) is a special case of in-forest (out-forest) in which there is exactly one task, called the *root*, with no immediate successor (predecessor). An *opposing forest* is a collection of in-trees and out-trees. For the purpose of minimizing schedule length, the cases of in-trees, out-trees, in-forests, and out-forests are equivalent. Thus any results for in-trees also apply to out-trees, in-forests, and out-forests. In this paper, we concentrate on in-trees only. Unless stated otherwise, we refer to an in-tree simply as a tree.

A list-scheduling algorithm that has been studied in the literature is the so-called *critical path* algorithm [4]. In the critical path algorithm, a list of tasks is formed by assigning higher priorities to tasks at higher level; ties are broken arbitrarily. A schedule obtained by the critical path algorithm is called a CP-*schedule*, and its length is denoted by ω_{CP}. Kaufman [6] showed that, for any tree-structured task system to be scheduled on m processors, $\omega_P \leqq \omega_N \leqq \omega_{CP} \leqq \omega_P + k - k/m$, where k is the maximum execution time of all tasks. Kunde [7] showed that, for any tree-structured task systems to be scheduled on m processors, $\omega_{CP}/\omega_N \leqq 2m/(m+1)$. Moreover, the bound is the best possible. We strengthen Kunde's analysis to show that $\omega_{CP}/\omega_P \leqq 2m/(m+1)$.

Let $(T_1, T_2, \cdots, T_l)$ be a chain in τ. A task T not on the chain is a *loop* task of the chain if T is a successor of some task T_i and a predecessor of some task T_j, where $j > i + 1$. τ is *simple* if there is a longest chain in τ that does not have any loop tasks. Clearly, simple task systems include in-forests, out-forests, and opposing forests. In this paper, we give an algorithm to construct a nonpreemptive schedule S for a simple task system on two processors such that $\omega_S/\omega_P \leqq 4/3$.

Muntz and Coffman [10] gave an algorithm to construct optimal preemptive schedules for arbitrary task systems on two processors. A schedule constructed by the Muntz–Coffman Algorithm is called an MC-*schedule*, and its length is denoted by ω_{MC}. One approach to prove the 4/3 bound for two processors is to devise an algorithm to convert

an MC-schedule into a nonpreemptive schedule S such that $\omega_S/\omega_{MC} \leqq 4/3$. We attempted this approach, but we could not obtain an algorithm for arbitrary execution times. We can, however, give an algorithm for task systems whose execution times are drawn from the set $\{1, 2, 3, 4\}$. Goyal [3] extended the Coffman–Graham Algorithm to schedule task systems with execution times 1 or k units on two processors. He showed that $\omega_G/\omega_N \leqq 4/3$ for $k = 2$ and $(3k - 1)/2k$ for $k > 2$, where ω_G denotes the length of the schedule produced by his algorithm. We note that our algorithm compares favorably with Goyal's algorithm for $k \leqq 4$.

We now introduce notation that is used throughout this paper. If S is a schedule, we let $s(T)$ and $f(T)$ denote the *starting* and *finishing* times of task T in S, respectively. The symbol $r(T)$ denotes the first time-instant in S at which T becomes ready. The paper is organized as follows. In the next two sections, we give the results for UET and tree-structured task systems, respectively. In § 4 we concentrate on the case of two processors. Finally, we draw some concluding remarks in the last section.

2. UET task systems. In this section we assume that τ is a UET task system, and we show that $\omega_{CG}/\omega_P \leqq 2m/(m + 1)$. The next theorem can be easily proved by an interchange argument.

THEOREM 1. *For any* UET *task system to be scheduled on m processors, we have that* $\omega_L = \omega_N$.

The Coffman–Graham Algorithm is a special critical path algorithm, where the priorities assigned to two tasks at the same level are determined by the priorities assigned to their immediate successors. For each $T \in \tau$, the Coffman–Graham Algorithm assigns to T an integer $\alpha(T)$, which represents the priority of T. Let $IS(T)$ denote the set of immediate successors of T and let $N(T)$ denote the decreasing sequence of integers obtained from the set $\{\alpha(T') \mid T' \in IS(T)\}$. The Coffman–Graham Algorithm assigns priorities to tasks as follows.

Coffman–Graham Algorithm

1. Choose an arbitrary terminal task T and define $\alpha(T)$ to be 1.
2. Suppose that the integers $1, 2, \cdots, j - 1$ have been assigned. Let R be the set of tasks, all of whose immediate successors have been labeled. Let T^* be in R such that $N(T^*)$ is lexicographically less than or equal to $N(T)$ for all T in R. Define $\alpha(T^*)$ to be j.
3. Repeat the above step until all tasks have been labeled.
4. Construct a list schedule using the list $(T_n, \cdots, T_1)$, where $\alpha(T_i) = i$ for all $1 \leqq i \leqq n$.

A schedule constructed by the Coffman–Graham Algorithm is called a CG-*schedule*, and its length is denoted by ω_{CG}. To prove that CG-schedules are optimal on two processors, Coffman and Graham [2] divided the schedules into *segments*. These segments have the important property that all tasks in one segment precede every task in the next segment. Thus, in any schedules whatsoever, we must finish all tasks in one segment before we can start any tasks in the next segment. On two processors, the Coffman–Graham Algorithm schedules the tasks in each segment optimally. In [8] the segments are called *blocks*. One or more consecutive blocks are merged together, along with some extra tasks that do not belong to any blocks, to form segments. These segments also have the property that all tasks in one segment precede every task in the next segment. Thus we can bound the ratio ω_{CG}/ω_N by bounding the ratio of the lengths of a segment in a CG-schedule and an optimal nonpreemptive schedule. In [8] it is shown that the ratio is no more than $2 - 2/m$. We follow the same technique and show that the ratio of the

lengths of a segment in a CG-schedule and an optimal preemptive schedule is no more than $2m/(m+1)$. This implies that $\omega_{CG}/\omega_P \leqq 2m/(m+1)$.

Let X be a block as defined in [8]. A time unit in X during which all processors are executing tasks is called a *full* column of X. Time units in X that are not full columns are called *partial* columns. The next two lemmas are immediate from the proofs in [8].

LEMMA 1. *Let $X_i, \cdots, X_{i-k}$, $k \geqq 0$, be the blocks in a segment W and let there be p partial columns in these blocks. Then there is a chain of length p in W.*

LEMMA 2. *Let f and p be the numbers of full and partial columns in a segment W, respectively. If E is the number of tasks in W, then $E \geqq mf + 2p - 1$.*

Using the above two lemmas, we can derive a lower bound for the optimal preemptive schedule length for the tasks in a segment, as given in Lemma 3. Lemma 4 is the key to proving that Liu's bound is valid for UET task systems.

LEMMA 3. *Let f and p be the numbers of full and partial columns in a segment W, respectively. If ω^* denotes the length of an optimal preemptive schedule for the tasks in W, then we have that $\omega^* \geqq \max\{p, (mf+2p-1)/m\}$.*

Proof. The proof is immediate from Lemmas 1 and 2. □

LEMMA 4. *Let ω and ω^* be the lengths of a segment W in a* CG*-schedule and an optimal preemptive schedule, respectively. Then we have that $\omega/\omega^* \leqq 2m/(m+1)$.*

Proof. Let there be p partial columns and f full columns in W. Then we have that $\omega = f + p$. We consider the following two cases.

Case 1. We have that $f \geqq p$.

Since $\omega = f + p$ and $f \geqq p$, we have that $f \geqq \omega/2$. Letting E be the number of tasks in W, we have that $E \geqq mf + p = (m-1)f + \omega \geqq (m-1)\omega/2 + \omega = (m+1)\omega/2$. Since $\omega^* \geqq E/m$, we have that $\omega/\omega^* \leqq 2m/(m+1)$.

Case 2. We have that $f < p$.

Since f and p are integers, we have that $f - p + 1 \leqq 0$. Thus $(m+1)\omega = (m+1)(f+p) = (mf+2p-1) + mp + (f-p+1) \leqq m\omega^* + m\omega^* + (f-p+1)$ (by Lemma 3) $\leqq 2m\omega^*$. Hence we have that $\omega/\omega^* \leqq 2m/(m+1)$. □

THEOREM 2. *For any* UET *task system to be scheduled on m processors, we have that $\omega_N/\omega_P = \omega_L/\omega_P \leqq \omega_{CG}/\omega_P \leqq 2m/(m+1)$.*

Proof. The proof is immediate from Theorem 1 and Lemma 4. □

3. Tree-structured task systems. In this section, we assume that τ is a tree-structured task system; i.e., its precedence graph is an in-tree. We show that, for any tree-structured task systems to be scheduled on m processors, $\omega_L = \omega_N$ and $\omega_{CP}/\omega_P \leqq 2m/(m+1)$.

THEOREM 3. *For any tree-structured task systems to be scheduled on m processors, we have that $\omega_L = \omega_N$.*

Proof. Let S be a nonpreemptive schedule for a tree-structured task system on m processors. The theorem is proved if we can show that S can be transformed into a list schedule S' with no increase in schedule length. To this end, we define the following three transformations. By SWAP (t, P_k, P_l), we mean swapping the tasks executed at or after time t on processor P_k with those on P_l.

Type I *transformation*. Suppose that T_i is executed on P_k. If T_i is ready at $t' < s(T_i)$ and P_k is idle in the interval $(t', s(T_i))$, then reassign T_i to start at t'.

Type II *transformation*. Suppose that T_i is executed on P_k and its immediate successor T_j is executed on P_l, where $k \neq l$. If $f(T_i) = s(T_j)$ and P_l is idle immediately before $s(T_j)$, then perform SWAP $(f(T_i), P_k, P_l)$. Note that, after this transformation, any task that starts at $f(T_i)$ on P_l cannot be the immediate successor of T_i.

Type III *transformation*. Suppose that T_i is executed on P_k and P_k is busy immediately before $s(T_i)$. If T_i is ready at $t' < s(T_i)$ and there is a processor P_l such that P_l is idle in the interval $(t', s(T_i))$, then perform SWAP $(s(T_i), P_k, P_l)$.

We apply the above transformations exhaustively to S. The resulting schedule S' cannot be longer than S, since none of the transformations can increase the schedule length. In the following, we show that S' is a list schedule; i.e., there is no task T_i ready at $t_1 < s(T_i)$ but not assigned to a processor P_l that is idle at t_1. By way of contradiction, let T_i be such a task and P_l be the idle processor. Observe that P_l cannot be idle from t_1 until the end of the schedule; otherwise, we can apply a Type III transformation to T_i. Let (t_1, t_2) be the maximal interval during which P_l is idle and T_j be the task that starts at t_2 on P_l. Since no Type II transformation can be applied to T_j, all immediate predecessors of T_j have finished earlier than t_2. This implies that a Type I transformation can be applied to T_j, contradicting our assumption that it cannot. □

Before we show that $\omega_{CP}/\omega_P \leqq 2m/(m+1)$, we first derive some properties of list schedules for tree-structured task systems. Recall that for a schedule S and a task T, $r(T)$ denotes the first time-instant in S at which T becomes ready. T is called a *waiting* task if $r(T) < s(T)$. If S is a list schedule and T is a waiting task, then we can conclude that all processors are busy in S during the interval $(0, s(T))$. This is shown in Lemma 6. First, we need the following lemma.

LEMMA 5. *Let S be a list schedule. For any two time-instants $t_1 < t_2$, the number of busy processors in S at t_1 is no less than that at t_2.*

Proof. In S a processor is left idle at t_1 only when there are no tasks ready at t_1. The lemma follows immediately from the observation that, in a tree-structured task system, a completed task can make at most one task ready. □

LEMMA 6. *Let S be a list schedule and T be a waiting task. Then S has no idle time in the interval $(0, s(T))$.*

Proof. Since S is a list schedule and T is a waiting task, all processors are busy in the interval $(r(T), s(T))$. By Lemma 5, all processors are busy in the interval $(0, r(T))$, also. Hence S has no idle time in the interval $(0, s(T))$. □

A useful concept of list schedules is the notion of a *maximal contiguous* chain. Let S be a list schedule. A chain $(T_1, T_2, \cdots, T_l)$ is called a maximal contiguous chain if (1) either $s(T_1) = 0$ or T_1 is a waiting task, (2) $f(T_i) = s(T_{i+1})$ for each $1 \leqq i < l$, and (3) $f(T_l) = \omega_S$. It is clear that a list schedule must have a maximal contiguous chain. The next lemma shows that if the length of a maximal contiguous chain is no more than half of the schedule length, then $\omega_S/\omega_P \leqq 2m/(m+1)$.

LEMMA 7. *Let S be a list schedule and l_c be the length of a maximal contiguous chain. If $l_c \leqq \omega_S/2$, then $\omega_S/\omega_P \leqq 2m/(m+1)$.*

Proof. Let T_1 be the first task of the maximal contiguous chain and t be the starting time of T_1. From the definition of maximal contiguous chains, we have that $t = \omega_S - l_c$. Since $l_c \leqq \omega_S/2$, $t \geqq \omega_S/2 > 0$, and hence T_1 is a waiting task. By Lemma 6, S has no idle time in the interval $(0, t)$. Letting E be the total execution time of all tasks, we have that $E \geqq mt + l_c = (m-1)t + \omega_S \geqq (m-1)\omega_S/2 + \omega_S = (m+1)\omega_S/2$. Since $\omega_P \geqq E/m$, we have that $\omega_S/\omega_P \leqq 2m/(m+1)$. □

Lemma 7 shows that Liu's bound is valid for any list schedule S with $l_c \leqq \omega_S/2$. To show that Liu's bound is also valid for $l_c > \omega_S/2$, we consider CP-schedules. CP-schedules have the property that if T is a waiting task, then any tasks started earlier than T have level at least that of T. This property was first proved by Kunde [7]. In the following, we use $l(T)$ to denote the level of T.

LEMMA 8 (see [7, Lemma 1]). *Let S be a* CP*-schedule and T be a waiting task. Then, for any task T' with $s(T') < s(T)$, $l(T') \geqq l(T)$.*

In a preemptive schedule, a task may be preempted with the remaining portion executed at a later time. Therefore there is a need to indicate the level of a portion of a task. If T is at level $l(T)$ and it has executed a portion $x < e(T)$, then the level of the

remaining portion of T is $l(T) - x$. We use E_l to denote the total execution time of those portions of tasks whose levels are at least l. Thus E_0 equals the total execution time of all tasks. The next two lemmas are instrumental in proving the bound.

LEMMA 9. *Let S be a* CP*-schedule, T be a waiting task, and t be its starting time. Then those portions of tasks executed before t have levels at least $l(T) - t$.*

Proof. Let T' be a task with $s(T') < t$. By Lemma 8, we have that $l(T') \geqq l(T)$. During the interval $(0, t)$, at most t units of T' can be executed. Thus any portions of T' executed in the interval $(0, t)$ have levels at least $l(T) - t$. □

LEMMA 10. *It holds that $\omega_P \geqq E_l/m + l$.*

Proof. Let S be an optimal preemptive schedule and t be the last instant in S such that a portion of a task at level l is executed. Clearly, we have that $t \geqq E_l/m$. Since it takes at least l time units to finish the remaining tasks at level less than l, we have that $\omega_P \geqq E_l/m + l$. □

LEMMA 11. *Let S be a* CP*-schedule and l_c be the length of a maximal contiguous chain. If $l_c > \omega_{CP}/2$, then we have that $\omega_{CP}/\omega_P \leqq 2m/(m+1)$.*

Proof. Let T_1 be the first task of the maximal contiguous chain and t be the starting time of T_1. If $t = 0$, then $\omega_{CP} = l_c = \omega_P$, and hence the lemma is proved. Thus we may assume $t > 0$, and, consequently, T_1 is a waiting task. By Lemma 6, S has no idle time in the interval $(0, t)$. By Lemma 9, those portions of tasks executed before t must have levels at least $l_c - t$. Since T_1 is at level l_c and T_1 is executed after t, we have that $E_{l_c - t} \geqq mt + t$. By Lemma 10, we have that $\omega_P \geqq (m+1)t/m + (l_c - t) = (t + ml_c)/m = (\omega_{CP} + (m-1)l_c)/m \geqq (\omega_{CP} + (m-1)\omega_{CP}/2)/m = (m+1)\omega_{CP}/(2m)$. Hence we have that $\omega_{CP}/\omega_P \leqq 2m/(m+1)$. □

THEOREM 4. *For any tree-structured task systems to be scheduled on m processors, we have that $\omega_N/\omega_P = \omega_L/\omega_P \leqq \omega_{CP}/\omega_P \leqq 2m/(m+1)$.*

Proof. The proof is immediate from Theorem 3 and Lemmas 7 and 11. □

4. Two-processor systems. In this section, we concentrate on two-processor systems, showing that the 4/3 bound holds for two special classes of task systems. The case of simple task systems is shown in § 4.1, and the case of restricted execution times is shown in § 4.2. First, we introduce notation that is used throughout this section. If S is a (partial) schedule for τ, then the *concurrency* of S, denoted by C_S, is defined to be the total amount of time during which both processors are busy in S. I_S denotes the total idle time in S, and E_S denotes the total execution time of the tasks assigned in S. Thus $\omega_S = C_S + I_S$ and $E_S = 2C_S + I_S$. If S is understood, then we let C, I, and E denote C_S, I_S, and E_S, respectively. For a given set of tasks R, $E(R)$ denotes the total execution time of all tasks in R. Thus $E(TS)$ denotes the total execution time of all tasks in $\tau = (TS, G, e)$.

In proving the 4/3 bound, we need only consider those τ such that its optimal preemptive schedule has no idle time. Otherwise, we can construct τ' from τ by adding enough independent tasks to fill the idle intervals in S_P. If the 4/3 bound holds for τ', then it also holds for τ. Since optimal preemptive schedules can be constructed by the Muntz–Coffman Algorithm, any τ can be converted in polynomial time into τ' satisfying this property. Throughout this section, we assume that τ satisfies this property, and hence $\omega_P = E(TS)/2$.

We can further restrict our attention to those τ with the length of the longest chain more than $E(TS)/3$. We show in Theorem 5 that, if the length of the longest chain is no more than $E(TS)/3$, then any list schedule S for τ satisfies $\omega_S/\omega_P \leqq 4/3$. The next two lemmas are instrumental in proving Theorem 5.

LEMMA 12. *Let S be a nonpreemptive schedule for τ. If $I \leqq \omega_S/2$, or, equivalently, $I \leqq C$, then $\omega_S/\omega_P \leqq 4/3$.*

Proof. Since $\omega_S = I + C$, $I \leqq \omega_S/2$ if and only if $I \leqq C$. If $I \leqq \omega_S/2$, then $E(TS) = 2\omega_S - I \geqq 3\omega_S/2$. Since $\omega_P \geqq E(TS)/2$, we have that $\omega_P \geqq 3\omega_S/4$, and hence $\omega_S/\omega_P \leqq 4/3$. □

LEMMA 13 (see [1], [4]). *Let S be a list schedule for τ. If l denotes the length of the longest chain in τ, then we have that $I \leqq l$.*

THEOREM 5. *Let S be a list schedule for τ and l be the length of the longest chain in τ. If $l \leqq E(TS)/3$, then $\omega_S/\omega_P \leqq 4/3$.*

Proof. By Lemma 13, we have that $I \leqq l \leqq E(TS)/3$. Since $\omega_S = (E(TS) + I)/2 \geqq (3I + I)/2 = 2I$, we have that $I \leqq \omega_S/2$. By Lemma 12, we have that $\omega_S/\omega_P \leqq 4/3$. □

4.1 Simple task systems. In this section, we show that the 4/3 bound holds for simple task systems. Recall that $\tau = (TS, G, e)$ is a simple task system if there is a longest chain $\mathrm{LC} = (T_1, T_2, \cdots, T_l)$ such that LC has no loop tasks. The tasks in $TS - \mathrm{LC}$ can be partitioned into three disjoint sets: PRED, SUCC, and REM, where PRED (SUCC) is the set of tasks that are predecessors (successors) of some tasks in LC, and REM is the set of all remaining tasks. Let T_p, T_r, and T_s be arbitrary tasks in PRED, REM, and SUCC, respectively. It is clear that T_p cannot be a successor of T_r or T_s, and T_r cannot be a successor of T_s. Thus, if S_1, S_2, and S_3 are valid schedules for the tasks in PRED, REM, and SUCC, respectively, then $S_1 \| S_2 \| S_3$ is a valid schedule for the tasks in PRED ∪ REM ∪ SUCC. As is seen in Theorem 6, the proof that the 4/3 bound holds consists of constructing a schedule S, where the tasks in PRED ∪ REM ∪ SUCC are scheduled on one processor and the tasks in LC are scheduled on the other. It will be shown that $C \geqq I$, and hence the 4/3 bound holds by Lemma 12.

Before we prove Theorem 6, we must introduce the following notation. Let PRED be partitioned into $A_1, A_2, \cdots, A_l$ such that, for each $1 \leqq i \leqq l$, A_i is the set of tasks that are predecessors of T_i but not T_{i-1}. Similarly, SUCC is partitioned into $B_1, B_2, \cdots, B_l$ such that, for each $1 \leqq i \leqq l$, B_i is the set of tasks that are successors of T_i but not T_{i+1}. Since T_1 has no predecessors and T_l has no successors, we have that $A_1 = B_l = \varnothing$. Observe that T_i cannot start until T_{i-1} and all tasks in A_i have finished. Similarly, T_{i+1} and all tasks in B_i cannot start until T_i has finished.

THEOREM 6. *Let $\tau = (TS, G, e)$ be a simple task system to be scheduled on two processors. Then we have that $\omega_N/\omega_P \leqq 4/3$.*

Proof. Let $\mathrm{LC} = (T_1, T_2, \cdots, T_l)$ be the longest chain that has no loop tasks and lc be its length. From the discussions in § 4, we may assume that $\mathrm{lc} > E(TS)/3$ and $\omega_P = E(TS)/2$. We construct a nonpreemptive schedule S for τ as follows. Starting at time 0, the tasks in $A_1, A_2, \cdots, A_l$, REM, $B_1, B_2, \cdots$, and B_l are successively scheduled on P_2. The tasks in each set are scheduled such that precedence constraints are observed. The tasks on the chain LC are scheduled on P_1 as follows. T_1 is scheduled to start at time 0. For each $2 \leqq i \leqq l$, T_i is scheduled to start at time $t = \max\{f(T_{i-1}), f(A_i)\}$.

We show that S is a valid schedule. It is clear that there are no precedence violations between the tasks scheduled on P_2. Furthermore, there are no precedence violations between a task in LC and a task in $\cup_{i=1}^{l} A_i \cup$ REM. Thus all we must show is that there are no precedence violations between a task in LC and a task in $\cup_{i=1}^{l} B_i$. We show by contradiction that, for each $1 \leqq i \leqq l$, $f(T_i) \leqq s(B_i)$. Suppose that i is the smallest index such that $f(T_i) > s(B_i)$. Let t' be such that $(t', f(T_i))$ is the maximal interval during which P_1 is busy. Then t' must be the starting time of some task T_x in LC and also the finishing time of A_x. Let $F = \cup_{j=x+1}^{l} A_j \cup \mathrm{REM} \cup \cup_{j=1}^{i-1} B_j$. Since $f(T_i) > s(B_i)$, we have that $\sum_{j=x}^{i} e(T_j) > E(F)$. However, the tasks $T_x, T_{x+1}, \cdots, T_i$ can execute concurrently only with the tasks in F. Thus any schedules for τ on two processors must have

some idle times, contradicting our assumption that $\omega_P = E(TS)/2$. Therefore S is a valid schedule.

Since $f(T_l) \leqq s(B_l) \leqq f(B_l)$, every task on LC must be executed concurrently with some tasks on P_2. Thus $C = \text{lc} > E(TS)/3$. Since $E(TS) = 2C + I$, we have that $I < C$. By Lemma 12, we have that $\omega_S/\omega_P \leqq 4/3$, and hence $\omega_N/\omega_P \leqq 4/3$. □

4.2. Restricted execution times. In this section, we show that the 4/3 bound holds for task systems with execution times drawn from the set $\{1, 2, 3, 4\}$. The idea is to convert the MC-schedule S_{MC} into a nonpreemptive schedule S such that $I \leqq C$. By Lemma 12, we have that $\omega_S/\omega_P \leqq 4/3$. The conversion is done in two steps. First, S_{MC} is converted into an *extended* schedule S', where a task can be executed concurrently by two processors. S' has the property that the intervals of execution of two *preempted* tasks do not overlap; i.e., they are either disjoint, or one is included in the other. The algorithms for this conversion are given in § 4.2.1. Second, S' is converted into a nonpreemptive schedule S by rescheduling the preempted tasks and the tasks that are executed concurrently by two processors. The algorithms for this conversion are given in § 4.2.2. Before we provide the details, we must state the Muntz–Coffman Algorithm and show an important property of MC-schedules that is used in later sections.

Muntz–Coffman Algorithm

Assign one processor each to the tasks at the highest level. If there is a tie among y tasks (because they are at the same level) for the last $x (x < y)$ processors, then assign x/y of a processor to each of these y tasks. Whenever either of the two events described below occurs, reassign the processors to the unexecuted portion of the task system according to the above rule. These are the following:

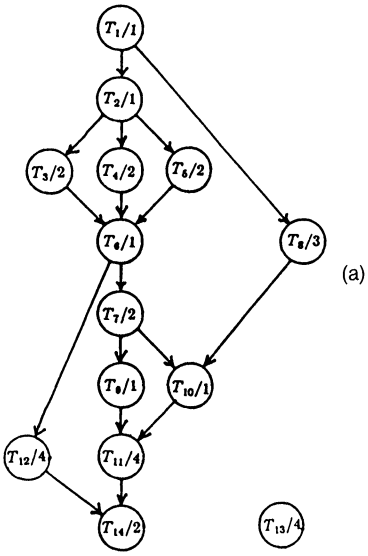
1. A task is finished;
2. We reach a point where, if we were to continue the present assignment, we would be executing some tasks (in the unexecuted portion of the task system) at a lower level at a faster rate than other tasks at a higher level.

Figure 1 shows the MC-schedule for a task system τ. Note that the Muntz–Coffman Algorithm allows a processor to be shared by several tasks. The next theorem shows that the finishing time of each task in a MC-schedule must be an integral multiple of 1/2; it holds even if the MC-schedule has some idle times. Due to space limitation, we omit the proof here; it can be found in [5].

THEOREM 7. *Let S_{MC} be the* MC*-schedule for a task system with integer execution times. Then $f(T)$ is an integral multiple of* 1/2 *for each task T.*

4.2.1. Conversion. In this section, we show how to construct an *extended* schedule, which is used later for constructing a nonpreemptive schedule. An extended schedule is not a valid schedule, in the sense that some tasks can be executed concurrently on both processors; they are called *double* tasks, while the others are called *regular* tasks. Like regular tasks, double tasks can be further divided into preempted and nonpreempted double tasks, depending on whether it is preempted. The construction is done in two steps. First, an extended schedule is constructed from the MC-schedule. Second, the schedule obtained is examined to see if all nonpreempted double tasks satisfy a certain condition (specified later). Those tasks not satisfied with the condition will be rescheduled, and the rescheduling process may involve other tasks or portions of other tasks. As is seen in the next section, the condition put on the nonpreempted double tasks is to ensure that the 4/3 bound holds for the final nonpreemptive schedule.

We now introduce some notation that is used throughout this section. Let S be a schedule with length ω_S. An idle interval (a, b) on a processor $\hat{P}$ is called an *idle slot* if



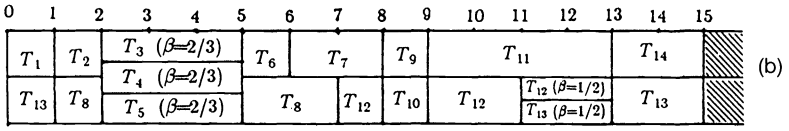


FIG. 1. *An* MC-*schedule for a task system*: (a) *a task system* τ, $T/x \equiv$ *Task* T *with execution time* x; (b) *the* MC-*schedule for* τ.

it is not properly contained in any other idle intervals on $\hat{P}$. S is called a *normalized* schedule if, for each idle slot (a, b) in S, two tasks start at b whenever $b < \omega_S$. Clearly, every list schedule is a normalized schedule. Let S be a normalized schedule and b_1, b_2, $\cdots$, b_k be the sequence of times such that $b_1 = 0$, $b_k = \omega_S$, and b_i is the right end of an idle slot for each $1 < i < k$. Then, for each $1 \leqq i < k$, the partial schedule of S in the interval (b_i, b_{i+1}) is called the ith *segment* of S. Note that each segment has exactly one idle slot. By interchanging processors if necessary, we can make all idle slots to be on P_2. In the algorithm given below, the schedule produced at each iteration is a normalized schedule, and we assume that all idle slots are on P_2. For each $1 \leqq i \leqq n$, we let $\hat{s}(T_i)$ and $\hat{f}(T_i)$ denote the starting and finishing times of T_i in the MC-schedule, respectively. Algorithm A, below, constructs the first phase of an extended schedule.

Algorithm A

1. Construct a list of tasks, $L = (T_1, T_2, \cdots, T_n)$ such that $\hat{f}(T_i) \leqq \hat{f}(T_{i+1})$ for each $1 \leqq i < n$; ties are broken by smaller starting times.

2. $i \leftarrow 0$.

3. $i \leftarrow i + 1$. If $i > n$, then STOP.

4. Let l be the length of the current schedule and t be the earliest time that T_i can be assigned to the current schedule without violating any precedence constraints. T_i will be assigned to use up as much idle time as possible, starting at t. The scheduling is done by one of the following rules.

Rule 1. Suppose that T_i can be nonpreemptively scheduled starting at t. If it can finish no later than $\hat{f}(T_i)$, then nonpreemptively schedule it on P_2 starting at t and go to step 3. (Note that T_i is a nonpreempted regular task if it is scheduled by Rule 1.)

Rule 2. Suppose that T_i can be nonpreemptively scheduled starting at t, but it cannot finish by $\hat{f}(T_i)$. Then nonpreemptively schedule a portion of it on P_2 in the interval $(t, \hat{f}(T_i))$ and the remaining portion on P_1 starting at l. Go to step 3. (Note that T_i is a nonpreempted double task if it is scheduled by Rule 2.)

Rule 3. Suppose that T_i can only be preemptively scheduled starting at t. If it can finish by l, then preemptively schedule it on P_2 starting at t and go to step 3. (Note that T_i is a preempted regular task if it is scheduled by Rule 3.)

Rule 4. Suppose that T_i can only be preemptively scheduled starting at t, but it cannot finish by l. Then preemptively schedule it on P_2 in the interval (t, l). Schedule half of the remaining portion of it on P_2 and the other half on P_1, both starting at l. (Note that T_i is a preempted double task if it is scheduled by Rule 4.) Now examine Rule 5.

Rule 5. Let R be the set of tasks executed in the interval (t, l) before T_i is assigned and r be the total amount of R executed in the interval. If $r < e(T_i)$, then reschedule all of R on P_1 and reschedule T_i by Rule 1 or Rule 2, starting at t. Go to step 3.

A schedule constructed by Algorithm A is called an A-*schedule*, denoted by S_A, and its length is dentoed by ω_A. We use $s_A(T_i)$ and $f_A(T_i)$ to denote the starting and finishing times of T_i in S_A, respectively. Figure 2 shows the A-schedule obtained from the MC-schedule in Fig. 1. Tasks T_1, T_2, T_3, and T_4 are scheduled by Rule 1, and T_5 by Rule 2. Figure 2(a) shows the schedule obtained after T_5 is scheduled. Tasks T_6 and T_8 are scheduled by Rules 1 and 4, respectively. Note that since the condition of Rule 5 is not satisfied, T_8 is not rescheduled. Figure 2(b) shows the schedule obtained after T_8 is scheduled. Task T_7 is scheduled by Rule 2, while T_9 and T_{10} are scheduled by Rule 1. Task T_{12} is initially scheduled by Rule 4, and the schedule is shown in Fig. 2(c). Since the condition of Rule 5 is satisfied, T_{12} is rescheduled with the resulting schedule, shown in Fig. 2(d). Tasks T_{11} and T_{13} are scheduled by Rules 2 and 4, respectively. Since the condition of Rule 5 is not satisfied, T_{13} is not rescheduled. Finally, T_{14} is scheduled by Rule 2 and the final schedule is shown in Fig. 2(e). In Fig. 2(e), T_5, T_7, T_{11}, and T_{14} are nonpreempted double tasks, and T_8 and T_{13} are preempted double tasks. Note that a nonpreempted double task is executed in two intervals, with one interval contained in the other. We call the smaller interval the *minor* part, and the larger one the *major* part of the task.

THEOREM 8. *Let S_A be the* A-*schedule for τ. Then we have that*

1. *Precedence constraints in τ are observed*;

2. *If T_i and T_j are two preempted tasks in S_A, then their execution intervals are either disjoint, or one is properly contained in the other. Furthermore, if the execution interval of T_j is properly contained in that of T_i, then T_i is never executed in the execution interval of T_j*;

3. *If (t, t') is the execution interval of a preempted task T_i, and r is the total amount of other tasks executed in the same interval, then $r \geqq e(T_i)$.*

Proof. The list L constructed in step 1 of Algorithm A has the property that precedence constraints in τ are observed. That is, if T_i is a predecessor of T_j in τ, then T_i precedes T_j in L. Since the tasks are scheduled in the order of L, property 1 holds. When a preempted task is scheduled by Algorithm A, it fills consecutive idle slots. Thus property

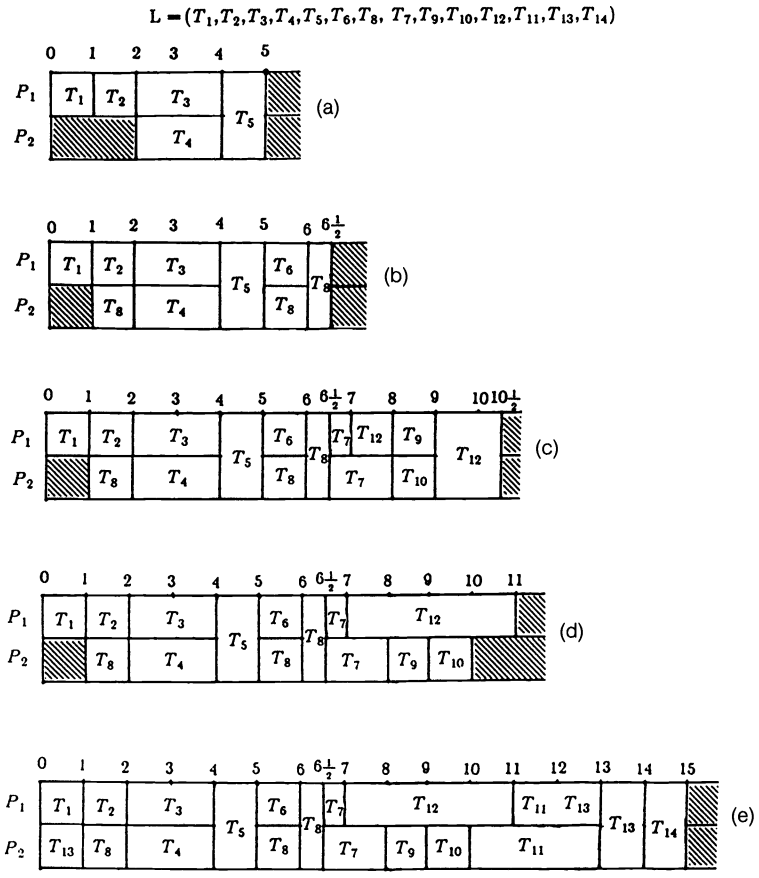


FIG. 2. *The* A-*schedule constructed from the* MC-*schedule in Fig.* 1: (a) *after* T_5 *is scheduled*; (b) *after* T_8 *is scheduled*; (c) *after* T_{12} *is scheduled by Rule* 4; (d) *after* T_{12} *is rescheduled by Rule* 5; (e) *the final* A-*schedule*.

2 holds. Finally, we observe that a preempted task T_i is scheduled by either Rule 3 or Rule 4. In both cases, we have that $r \geqq e(T_i)$. Thus property 3 holds. □

We show in Theorem 9 that there are no idle slots in an A-schedule and that each preempted portion of a preempted task has integer length. The next lemma, whose proof is omitted here but can be found in [5], is instrumental in proving Theorem 9.

LEMMA 14. *For each* $1 \leqq i \leqq n$, *let* $S_A(i)$ *be the schedule obtained after the first* i *tasks have been scheduled by Algorithm* A *and let* $\omega_A(i)$ *be its length. For each* $1 \leqq j \leqq i$, *let* $s_i(T_j)$ *and* $f_i(T_j)$ *denote the starting and finishing times of* T_j *in* $S_A(i)$, *respectively. Then we have, for each* $1 \leqq i \leqq n$, *that*

1. $\omega_A(i) \leqq \hat{f}(T_i)$;
2. $s_i(T_j)$ *and* $f_i(T_j)$ *are integral multiples of* 1/2 *for each* $1 \leqq j \leqq i$;
3. *Each idle slot in* $S_A(i)$ *has integer length*;
4. *For each* $1 \leqq j < i$, *if* $f_i(T_j) > \hat{f}(T_j)$, *then there is a task* T_r *such that the interval* $(s_i(T_j), f_i(T_j))$ *is contained in the interval* $(s_i(T_r), f_i(T_r))$ *and* $s_i(T_r) < \hat{f}(T_j)$.

THEOREM 9. *Let* S_A *be the* A-*schedule for* τ. *Then we have that*

1. S_A *does not have any idle slots*;
2. *If* T_i *is a preempted task in* S_A, *then each preempted portion of* T_i *has integer length*;

3. *Let T_i be a preempted task in S_A, and a and b be the first and last instants at which T_i coexecutes with another task, say T_j and T_k, respectively. Then the level of T_j at a and the level of T_k at b are integers.*

Proof. By Lemma 14, $\omega_A = \omega_{MC}$. Since S_{MC} does not have any idle slots, S_A cannot have any idle slots. Thus property 1 holds. Properties 2 and 3 follow from property 3 in Lemma 14. □

As is seen in the next section, the technique to reschedule the nonpreempted double tasks is different from that for the preempted tasks. The properties in Theorem 8 and properties 1 and 3 in Theorem 9 are sufficient to guarantee that the preempted tasks can be rescheduled such that the resulting schedule has concurrency no less than its total idle time. For nonpreempted double tasks, we need the schedule to satisfy additional properties. Let T_i be a nonpreempted double task with its minor part in the interval (a_i, b_i). It is desired that there is an interval $IT_i = (l_i, r_i)$ such that IT_i contains (a_i, b_i), $r_i - l_i \geqq 3(b_i - a_i)$, and no other nonpreempted double tasks or preempted tasks are executed in IT_i. Furthermore, for any two nonpreempted double tasks T_i and T_j, we need $IT_i \cap IT_j = \varnothing$. Algorithm B, given below, reschedules the nonpreempted double tasks such that these properties are satisfied, along with the properties in Theorem 8 and properties 1 and 3 in Theorem 9. Note that the RESCHEDULE procedure in Algorithm B is not given; it is given in the proof of Theorem 10. Also, the *continue* statement in Algorithm B causes the control flow to skip all subsequent statements in step 2.

Algorithm B

1. Let $T_1, T_2, \cdots, T_n$ be the n tasks such that $f_A(T_i) \leqq f_A(T_{i+1})$ for each $1 \leqq i < n$. *boundary* $\leftarrow \omega_A$. For each i from n to 1, if T_i is a nonpreempted double task, then perform step 2.

2. Let the minor part of T_i be in the interval (a_i, b_i). Let r_i be the earliest time such that $r_i \geqq b_i$ and r_i is either (a) *boundary*, (b) the starting time of the minor part of a double task, or (c) the starting or restarting time of a preempted task. If $b_i - a_i \leqq (r_i - b_i)/2$, then $\{$*boundary* $\leftarrow a_i$; continue$\}$. $l_i \leftarrow r_i - 3(b_i - a_i)$. If no other nonpreempted double tasks or preempted tasks are executed in the interval (l_i, r_i), then $\{$*boundary* $\leftarrow l_i$; continue$\}$. RESCHEDULE(T_i, r_i).

A schedule constructed by Algorithm B is called a B-*schedule*, denoted by S_B, and its length is denoted by ω_B. Figure 3 shows the B-schedule obtained from the A-schedule in Fig. 2. In the A-schedule shown in Fig. 2, T_5, T_7, T_{11}, and T_{14} are the nonpreempted double tasks. Only T_{14} must be rescheduled; the other tasks already have the conditions satisfied.

THEOREM 10. *Let S_B be the* B-*schedule obtained from S_A. Then the properties in Theorem* 8 *and properties* 1 *and* 3 *in Theorem* 9 *hold for S_B. Furthermore, if T_i is a nonpreempted double task in S_B such that its minor part is in the interval (a_i, b_i), then there is an interval $IT_i = (l_i, r_i)$ such that*

1. *IT_i contains (a_i, b_i) and $r_i - l_i \geqq 3(b_i - a_i)$;*
2. *No other preempted tasks or nonpreempted double tasks are executed in IT_i;*
3. *For any other nonpreempted double task T_j, we have that $IT_i \cap IT_j = \varnothing$.*

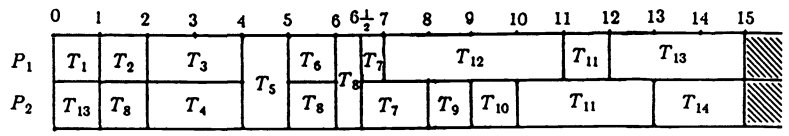


FIG. 3. *The* B-*schedule constructed from the* A-*schedule in Fig.* 2.

Proof. Let T_i be a nonpreempted double task in S_A. If T_i does not cause the RESCHEDULE procedure to be executed, then it is clear that the properties stated in Theorem 10 hold. Thus we may assume that RESCHEDULE(T_i, r_i) is executed in Algorithm B. Let (a_i, b_i) and (c_i, d_i) be the intervals of the minor and major parts of T_i in S_A, respectively. Without loss of generality, we may assume that the major part is executed on P_2. Let l_i and r_i be as computed in Algorithm B. By Algorithm B, there are no preempted or nonpreempted double tasks, besides T_i, executed in the interval (b_i, r_i). Furthermore, there must be some preempted or nonpreempted double tasks, besides T_i, executed in the interval (l_i, a_i). Let $x = b_i - a_i$ and $t = c_i - x$. Let U be the set of tasks, besides T_i, executed in the interval (t, a_i), and let $V \subseteq U$ be the set of preempted tasks in U. Note that if $T \in V$ and if a portion of T is executed totally outside the interval (t, a_i), then that portion of T is not counted toward V. Since T_i is a nonpreempted double task, any $T \in V$ must finish by a_i. We first remove all tasks in U from the schedule. Let $t_1 \leqq t$ and $t_2 \leqq t$ be the starting times of the idle slots on P_1 and P_2, respectively. We then try to assign the minor part of T_i in the interval (t, c_i) on P_2 (so that it becomes a nonpreempted regular task) and reschedule the tasks in U in the remaining idle slots so that the conditions of the theorem are satisfied. If this is impossible, then we leave T_i as it is and try to reschedule the tasks in U so that the conditions of the theorem are satisfied. We have the following two cases to consider.

Case I. Some predecessors of T_i are in U.

If $t_1 < t_2$, then we interchange P_1 with P_2 for the interval $(0, t_2)$, and, after the swap, we let t_1 and t_2 be the starting times of the idle slots on P_1 and P_2, respectively. Thus we always have that $t_1 \geqq t_2$. Let T_k and T_j be the tasks started at t_2 and t_1, respectively. (If $t_1 = t_2$, then we let T_k be the one that finished last.) Let W be the set of all predecessors of T_i in U. For each $T_l \in W$, $f_A(T_l) > t$ and $\hat{f}(T_l) \leqq t$, and hence $f_A(T_l) > \hat{f}(T_l)$. Thus, by Lemma 14, there is a T_q such that the interval $(s_A(T_l), f_A(T_l))$ is contained in the interval $(s_A(T_q), f_A(T_q))$, and $s_A(T_q) < \hat{f}(T_l) \leqq t$. Since T_k and T_j are the only tasks in U that start at or earlier than t, T_q must be either T_k or T_j. We now show that it is impossible for T_q to be T_j. Suppose that T_q is T_j. Then, since T_q is scheduled by Rule 5 of Algorithm A, the tasks in W must have been scheduled before T_j. Furthermore, each $T_l \in W$ must finish by $\hat{f}(T_l) \leqq t$ before T_j is scheduled. This means, however, that the tasks in W are scheduled in the interval (t_1, t) on P_1 before T_j is scheduled. Therefore it is impossible for T_j to start at t_1. Hence T_q must be T_k, and we have that $f_A(T_j) \leqq f_A(T_k)$.

The tasks in W are rescheduled by Rule 5 of Algorithm A because of T_k. Let t' be the latest finishing time for the tasks in W. Clearly, we have that $t' > t$. Let $X \supseteq W$ be the set of tasks executed in the interval (t_1, t') on P_1. We schedule the minor part of T_i in the interval (t, c_i) on P_2, and the tasks in X are scheduled on P_1 and P_2 as early as possible (this process unwinds the rescheduling by Rule 5 in Algorithm A). Figure 4(a) shows the schedule after the tasks in X are scheduled. Let t^* be the latest finishing time for the tasks in X and h be the total amount of tasks in X executed on P_2. Since all tasks in X should finish by t, we have that $t^* < t$, and, since $t' > t$, we have that $t^* + h \geqq t' > t$. Let $U' = U - X - \{T_k\}$. The tasks in U' and T_k are scheduled as shown in Fig. 4(b). Note that the tasks in U' are scheduled in the same relative order as in S_A. Observe that the finishing time of T_k must be larger than t. Finally, we interchange P_1 with P_2 at time t and thereafter. The final schedule is shown in Fig. 4(c).

Since T_k is preempted, it is not a nonpreempted double task. Thus no new nonpreempted double tasks are created in the process, and hence the properties in Theorem 10 hold. Properties 1 and 2 in Theorem 8 clearly hold. Since the double amount of T_k is $t - t^*$ and since $h > t - t^*$, property 3 in Theorem 8 also holds. Property 1 in Theorem

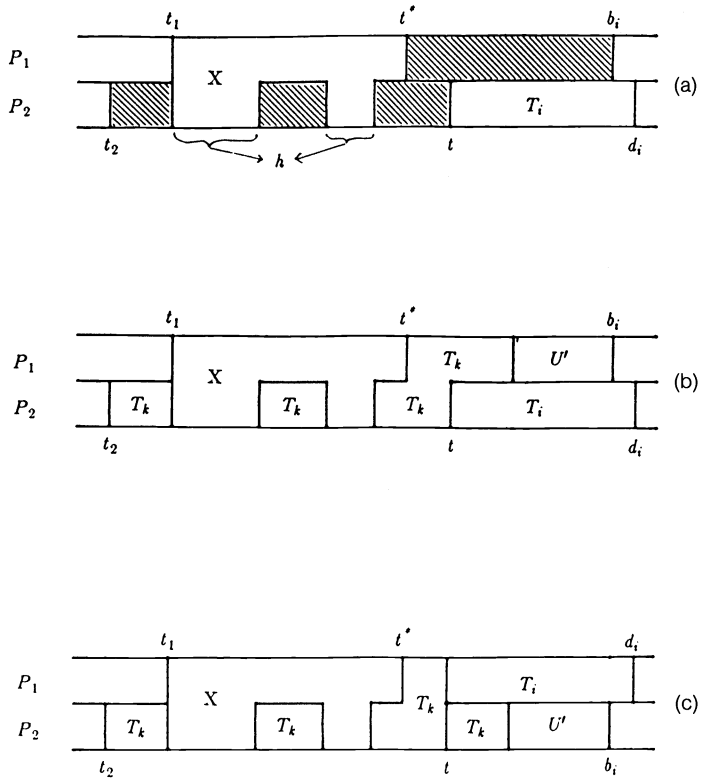


FIG. 4. *Illustrating Case* I *in Theorem* 10: (a) *after the tasks in X are scheduled*; (b) *after the tasks in* $U' \cup \{T_k\}$ *are scheduled*; (c) *interchanging* P_1 *with* P_2 *at* t *and thereafter.*

9 clearly holds. Since $d_i - b_i$ and $e(T_i)$ are integers and since each $T \in U'$ has integer execution time, the level of T_i when T is finished must be integer. Similarly, the level of T_i when T_k is finished must also be integer. Thus property 3 in Theorem 9 also holds.

Case II. No predecessors of T_i are in U.

We consider two cases depending on whether $t_1 \geqq t_2$.

Case 1. We have that $t_1 \geqq t_2$.

In this case, we schedule the minor part of T_i in the interval (t, c_i) on P_2 and the tasks in U in the remaining idle slots. Let $T_k \in U$ be the task that starts at t_2 on P_2. Then we have that $t < f_A(T_k) \leqq c_i$. Letting $d = f_A(T_k) - t$, we have that $d \leqq x$.

Suppose that T_k is not a preempted task; i.e., T_k is not in V. We schedule a portion of T_k in the interval (t_2, t) on P_2. Then the tasks in V, the remaining portion of T_k, and the tasks in $U' = U - V - \{T_k\}$ are scheduled in that order on P_1, starting at t_1. Note that the tasks in V are scheduled in the same relative order as in S_A and that the tasks in U' are scheduled so that precedence constraints are observed. Let S' denote this schedule. Clearly, properties 1 and 2 in Theorem 8 and property 1 in Theorem 9 hold. Property 3 in Theorem 8 also holds, since each task in V is a preempted regular task. Since $t_1 - t_2$ and the lengths of the tasks in V are integers, property 3 in Theorem 9 is also satisfied. Let the portion of T_k on P_1 be scheduled in the interval (t', t^*), where t' is the latest finishing time for the tasks in V and $t^* = t' + d$. As shown in Fig. 5, there are three possibilities for S'. In Fig. 5(a), we have that $t^* \leqq t$. In this case, T_k is a nonpreempted double task, and the length of its minor part is d. Since $d \leqq x \leqq (b_i - t)/2 \leqq (b_i - t^*)/$

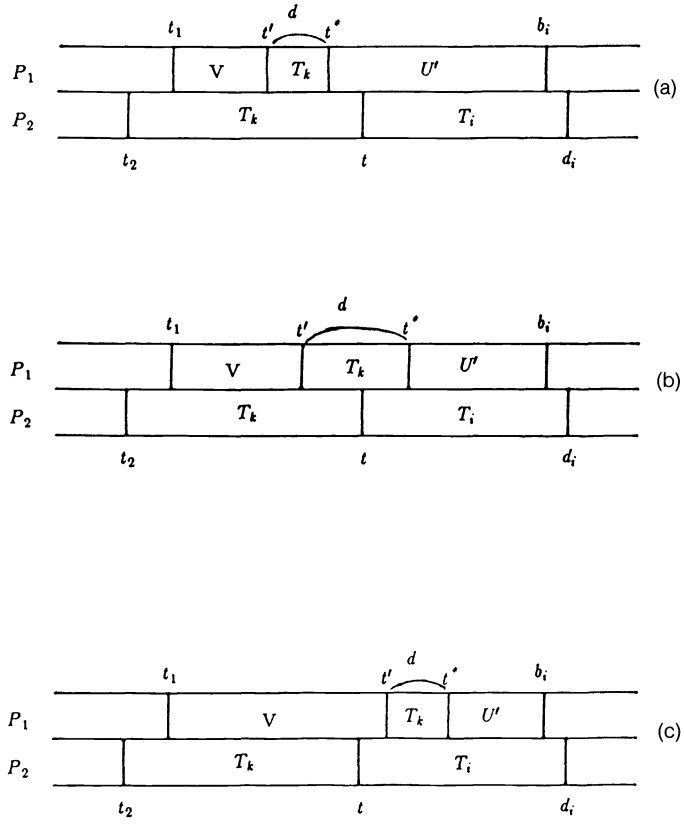


FIG. 5. *Illustrating Case* II *in Theorem* 10 ($t_1 \geqq t_2$): (a) $t^* \leqq t$; (b) $t' \leqq t < t^*$; (c) $t < t'$.

2 and since there are no preempted tasks or nonpreempted double tasks in the interval (t^*, b_i), the properties in Theorem 10 are satisfied. In Fig. 5(b), we have that $t' \leqq t < t^*$. In this case, we interchange P_1 with P_2 at t and thereafter. The resulting schedule is shown in Fig. 6(a). Clearly, the length of the minor part of T_k is less than d. Hence the properties in Theorem 10 hold, as in the previous case. Finally, in Fig. 5(c), we have that $t < t'$. Let V_1 and V_2 be the tasks executed on P_1 in the intervals (t_1, t) and (t, t'), respectively. Then at most one task in V, say $\hat{T}$, can be split into two parts, with one part in V_1 and the other in V_2. We now interchange P_1 with P_2 at time t and thereafter, and swap T_k with the tasks in V_2. The resulting schedule is shown in Fig. 6(b). Note that T_k is a nonpreempted regular task. Since no new nonpreempted double tasks are created, the properties in Theorem 10 hold. Since $d_i - b_i$ and the lengths of the tasks in V_2, besides $\hat{T}$, are integers, the level of T_i must be integer when each task in V_2 finishes. As explained above, the level of T_k is integer when each task in V_1 finishes. Thus property 3 in Theorem 9 also holds. In all three cases, we set *boundary* to be t_1.

If T_k is a preempted task (i.e., $T_k \in V$), then we let $V' = V - \{T_k\}$. Since the execution intervals of preempted tasks do not overlap, the tasks in V' must start after T_k is finished. Therefore no tasks in V' can be scheduled in the interval $(t_1, f_A(T_k))$. We consider two cases, depending on which rule was used to schedule T_k in Algorithm A. Suppose that T_k was scheduled by Rule 4 of Algorithm A. We schedule a portion of T_k in the interval (t_2, t) on P_2. The remaining portion of T_k, the tasks in V' and the tasks

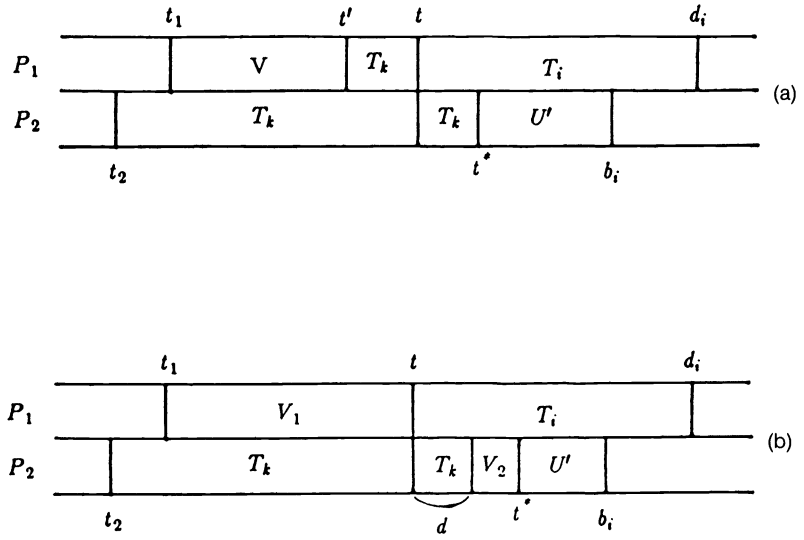


FIG. 6. *Transforming the schedules in Fig.* 5: (a) *the schedule obtained from the schedule in Fig.* 5(b); (b) *the schedule obtained from the schedule in Fig.* 5(c).

in $U - V$, are scheduled in that order on P_1, starting at t_1. Note that the tasks in V' are scheduled in the same relative order as in S_A and that the tasks in $U - V$ are scheduled so that precedence constraints are observed. Observe that T_k must finish after time t and that its double amount is decreased after the rescheduling. Since no new nonpreempted double tasks are created, the properties in Theorem 10 hold. Clearly, the properties in Theorem 8 and property 1 in Theorem 9 are satisfied. Since $d_i - b_i$ and the lengths of the tasks in $U - \{T_k\}$ are integers, the level of T_i must be integer when each task in U finishes. Thus property 3 in Theorem 9 also holds.

If T_k was scheduled by Rule 3 of Algorithm A, then we let $Y \subseteq U$ be the set of tasks executed on P_1 during the interval $(t_1, f_A(T_k))$ and y be the total execution time of the tasks in Y. Clearly, we have that $y \geqq f_A(T_k) - t_1$. Furthermore, since T_k is a preempted task, and since the execution intervals of preempted tasks do not overlap, the tasks in Y cannot be preempted tasks. We now schedule a portion of T_k in the interval (t_2, t) on P_2 and the remaining portion in the interval $(t_1, t_1 + d)$ on P_1. The tasks in $U - \{T_k\}$ are then scheduled on P_1, starting at $t_1 + d$. These tasks are scheduled in such a way that the task with the smallest finishing time is scheduled first. Thus the tasks in V' start after all tasks in Y are finished. Letting t' be the earliest starting time for the tasks in V', we have that $t' \geqq t_1 + d + y \geqq f_A(T_k) + d$. Hence there are no overlappings of executions of preempted tasks. Since T_k is a preempted task, no new nonpreempted double tasks are created, and hence the properties in Theorem 10 are satisfied. Clearly, properties 1 and 2 in Theorem 8 and property 1 in Theorem 9 hold. Since $d_i - b_i$ and the execution times of the tasks in $U - \{T_k\}$ are integers, property 3 in Theorem 9 is also satisfied. Let r be the total amount of tasks, besides T_k, executed in the interval $(s_A(T_k), \hat{t})$, where $\hat{t} = \max\{t, t_1 + d\}$. If $r \geqq e(T_k)$, then property 3 in Theorem 8 also holds, and we are done. Otherwise, we reschedule T_k to become a nonpreempted double task. We show that the properties in Theorem 10 hold after the rescheduling.

If $t_1 + d > t$, then we interchange P_1 with P_2 at t and thereafter. Furthermore, we do not reschedule the portion of T_k in the interval $(t, t_1 + d)$ on P_2. Let d' be the length of time during which T_k is doubly executed. Clearly, we have that $d' = \min\{d, t - t_1\}$.

Since $d + (t - t_1) = (f_A(T_k) - t) + (t - t_1) = f_A(T_k) - t_1$, we have that $d' \leqq (f_A(T_k) - t_1)/2$. (Note that if $\alpha = \min\{\beta, \gamma\}$, then $\alpha \leqq (\beta + \gamma)/2$.) Let $t^* = t_1 + d'$. Observe that there are no preempted tasks or nonpreempted double tasks, besides T_k, executed in the interval (t^*, t'). Since $t' \geqq t_1 + d + y$ and $t^* = t_1 + d'$, we have that $t' - t^* \geqq y$. Now $d' \leqq (f_A(T_k) - t_1)/2 \leqq y/2 \leqq (t' - t^*)/2$. We reschedule the tasks in the interval $(s_A(T_k), t_1)$. Let Z be the set of tasks, besides T_k, executed in the interval $(s_A(T_k), t_1)$ and z be the total amount of tasks in Z executed in the same interval. Then we have that $z < t^* - s_A(T_k)$; otherwise, we have that $r \geqq e(T_k)$. Furthermore, we have that $z > t_1 - s_A(T_k)$ since T_k is not doubly executed in the interval $(s_A(T_k), t_1)$. Now we schedule the tasks in Z on P_1 from $s_A(T_k)$ to $s_A(T_k) + z$, and T_k in the intervals $(s_A(T_k), t)$ and $(s_A(T_k) + z, t^*)$ on P_2 and P_1, respectively. Note that, after the rescheduling, T_k becomes a nonpreempted double task with its minor part executed in the interval $(s_A(T_k) + z, t^*)$ on P_1. Since $t_1 - s_A(T_k) < z < t^* - s_A(T_k)$, we have that $t_1 < s_A(T_k) + z < t^*$. Therefore the length of the minor part of T_k is less than d'. Since $d' \leqq (t' - t^*)/2$, the properties in Theorem 10 are satisfied.

Case 2. We have that $t_1 < t_2$.

Let $T_k \in U$ be the task scheduled on P_1 at t_1. If $t - t_2 \leqq x$ or $f_A(T_k) \leqq c_i$, then we interchange P_1 with P_2 for the interval $(0, t_2)$. After the interchange, we reschedule the tasks in U in the same manner as in Case 1. If T_k becomes a nonpreempted double task after the rescheduling, it is easy to see that the length of the minor part of T_k is no more than x. Thus we can use the same argument as in Case 1.

Now suppose that $t - t_2 > x$ and $f_A(T_k) > c_i$. Let X be the set of tasks executed in the interval $(f_A(T_k), a_i)$ on P_1. We want to show that T_k and the tasks in X cannot be preempted tasks. Observe that T_i is scheduled after the tasks in $X \cup \{T_k\}$ have been scheduled. Thus, if any task in $X \cup \{T_k\}$ is a preempted task, then it must have been scheduled by Rule 4 of Algorithm A, and hence its last portion must be executed concurrently by both processors. This contradicts the fact that T_i is executed on P_2 in the interval (c_i, a_i). Let $T_j \in U$ be the task scheduled on P_2 at t_2, and let Y be the set of tasks executed on P_2 in the interval $(f_A(T_j), c_i)$. Observe that $f_A(T_j) > t$ and that there must be some preempted tasks in $Y \cup \{T_j\}$. We consider two cases depending on whether T_j is a preempted task. If T_j is a preempted task, then we schedule the minor part of T_i on P_2 in the interval (t, c_i). A portion of T_j is scheduled in the interval (t_2, t) on P_2, and the remaining portion is scheduled on P_1, starting at $f_A(T_k)$. Finally, the tasks in Y follow T_j on P_1, and they are followed by the tasks in X. Since no new nonpreempted double tasks are created, the properties in Theorem 10 are satisfied. It is easy to see that the properties in Theorem 8, and also properties 1 and 3 in Theorem 9, hold.

If T_j is not a preempted task, then we have that $V \subseteq Y$. In this case, it may not be necessary to reschedule T_i. First, we swap the task T_j with the tasks in V, so that they are executed before T_j. Since $e(T_j) > t - t_2 > x$, T_j must be executing at time t after the swap. Let t^* be the new starting time of T_j. If $t - t^* \leqq x$, then we interchange P_1 with P_2 for the interval $(0, t^*)$ and reschedule the tasks in $U' \cup \{T_i\}$ in the same manner as in Case 1, where $U' = U - V$. On the other hand, if $t - t^* > x$, then we leave all other tasks unchanged. Thus T_i is still a nonpreempted double task with its minor and major parts in the intervals (a_i, b_i) and (c_i, d_i), respectively. Observe that there are no preempted tasks or nonpreempted double tasks, besides T_i, executed in the interval (t^*, b_i). Furthermore, $b_i - t^* > 3x$. Thus the properties in Theorem 10 are satisfied. It is easy to see that the properties in Theorem 8 and properties 1 and 3 in Theorem 9 hold. In this case, we set *boundary* to be t^*. □

4.2.2. Insertion. In this section we show how a B-schedule can be converted into a nonpreemptive schedule S such that $\omega_S/\omega_P \leqq 4/3$. A B-schedule can have four types

of tasks—preempted regular, preempted double, nonpreempted regular, and nonpreempted double. Preempted tasks (both regular and double) are rescheduled by Algorithm D, given later. Nonpreempted double tasks are rescheduled by Algorithm C, given below. Recall that a nonpreempted double task has two parts—minor and major, with the minor contained in the major. The algorithm to reschedule the nonpreempted double task is simply to expand the execution of the major part by the length of the minor, creating an idle interval with length twice that of the minor.

Algorithm C

1. Let S_1 be the given B-schedule and ω_1 be its length. Let $T_1, T_2, \cdots, T_k$ be the nonpreempted double tasks in S_1. For each i from 1 to k, perform step 2.

2. Without loss of generality, assume that the minor and major parts of T_i are executed on P_1 and P_2, respectively. Let t be the finishing time of the minor part of T_i, and let x be its length. Shift the tasks executed in the interval (t, ω_i) on both processors to the right by x. Remove the minor part of T_i from P_1 and assign it to the interval $(t, t + x)$ on P_2. Let S_{i+1} be the new schedule and ω_{i+1} be its length.

A schedule constructed by Algorithm C is called a C-*schedule*, denoted by S_C, and its length is denoted by ω_C. Figure 7 shows the C-schedule obtained from the B-schedule in Fig. 3. Note that there are only three nonpreempted double tasks in the B-schedule in Fig. 3, namely, T_5, T_7, and T_{11}.

THEOREM 11. *Let S_C be a* C-*schedule and (a_i, b_i) be an idle slot in S_C. Then there is an interval $IT_i = (l_i, r_i)$ such that*

1. *IT_i contains the interval (a_i, b_i), and $r_i - l_i \geqq 2(b_i - a_i)$;*
2. *No preempted tasks are executed in IT_i;*
3. *For any other idle slot (a_j, b_j), we have that $IT_i \cap IT_j = \varnothing$.*

Proof. By Algorithm C, the idle slot (a_i, b_i) is created because of rescheduling a nonpreempted double task T_i. The minor part of T_i has length $(b_i - a_i)/2$. The theorem follows immediately from Theorem 10. □

THEOREM 12. *Let τ be a task system such that its* B-*schedule has no preempted tasks. Then we have that $\omega_N/\omega_P \leqq 4/3$.*

Proof. Since the B-schedule for τ has no preempted tasks, its C-schedule has no preempted tasks, also. Thus S_C is a valid nonpreemptive schedule for τ. By Theorem 11, the concurrency of S_C is no less than its total idle time. Thus, by Lemma 12, we have that $\omega_N/\omega_P \leqq \omega_C/\omega_P \leqq 4/3$. □

We now consider how to reschedule the preempted tasks. Observe that the execution intervals of preempted tasks in a C-schedule do not overlap. That is, if T_i and T_j are two preempted tasks in S_C, then the execution intervals of T_i and T_j are either disjoint, or one is properly contained in the other. Furthermore, if the execution interval of T_j is properly contained in that of T_i, then T_i is never executed in the execution interval of T_j. The rescheduling proceeds from the preempted task in the *innermost* level to the *outermost* one. Suppose that we are considering T_i. We divide the current schedule into three portions—the left portion from the beginning of the schedule to the starting time of T_i, the middle portion from the starting time of T_i to the finishing time of T_i, and

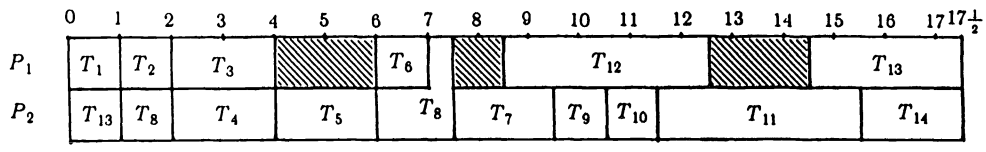


FIG. 7. *The* C-*schedule constructed from the* B-*schedule in Fig.* 3.

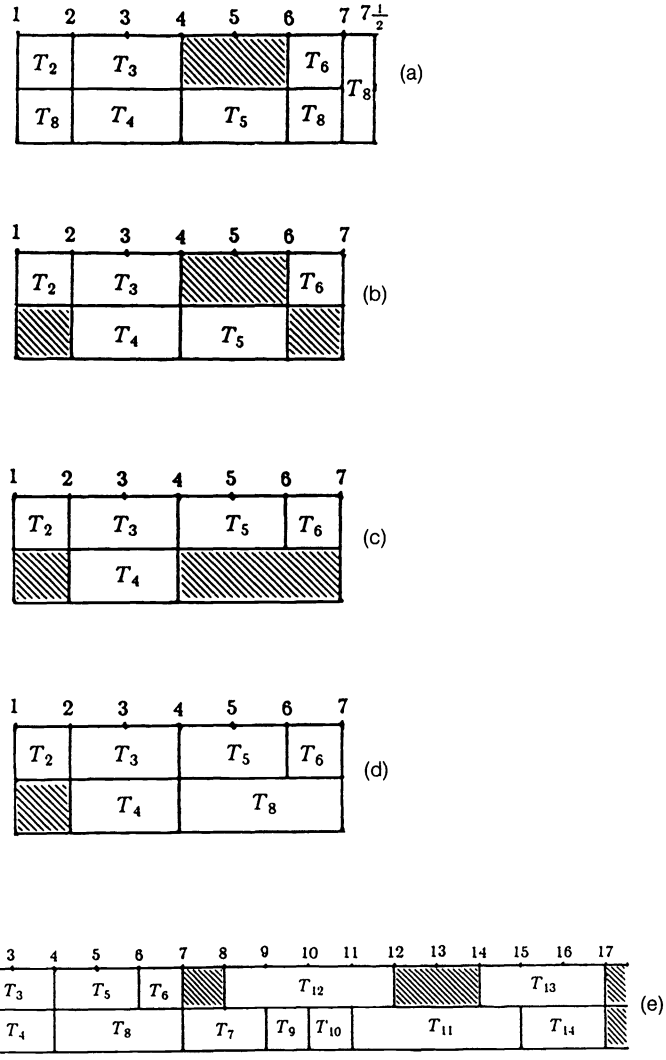


FIG. 8. *The schedule obtained after inserting* T_8: (a) *the schedule* S_M; (b) *the schedule* S_{M1}; (c) *the schedule* S_{M2}; (d) *the schedule* S_{M3}; (e) *the schedule after concatenation.*

the right portion from the finishing time of T_i to the end of the schedule. The middle portion is rescheduled, and the resulting schedule is concatenated with the left and right portions again. To reschedule the middle portion, we first remove all of T_i from the schedule. Then the schedule is rearranged to form a *proper* schedule (defined later). Finally, the whole piece of T_i is inserted into the proper schedule. This is done in such a way that the resulting schedule has concurrency no less than its total idle time. Thus, after all preempted tasks have been rescheduled, the final schedule has the property that its concurrency is no less than its total idle time, and hence the 4/3 bound holds by Lemma 12. The algorithm is given below. Note that there are three subroutines used in the algorithm, but they are not given there; they are given later when we characterize the schedules obtained at each iteration of the algorithm.

Algorithm D

1. Let S_1 be the given C-schedule and ω_1 be its length. $i \leftarrow 1$. While there is a preempted task, perform step 2.

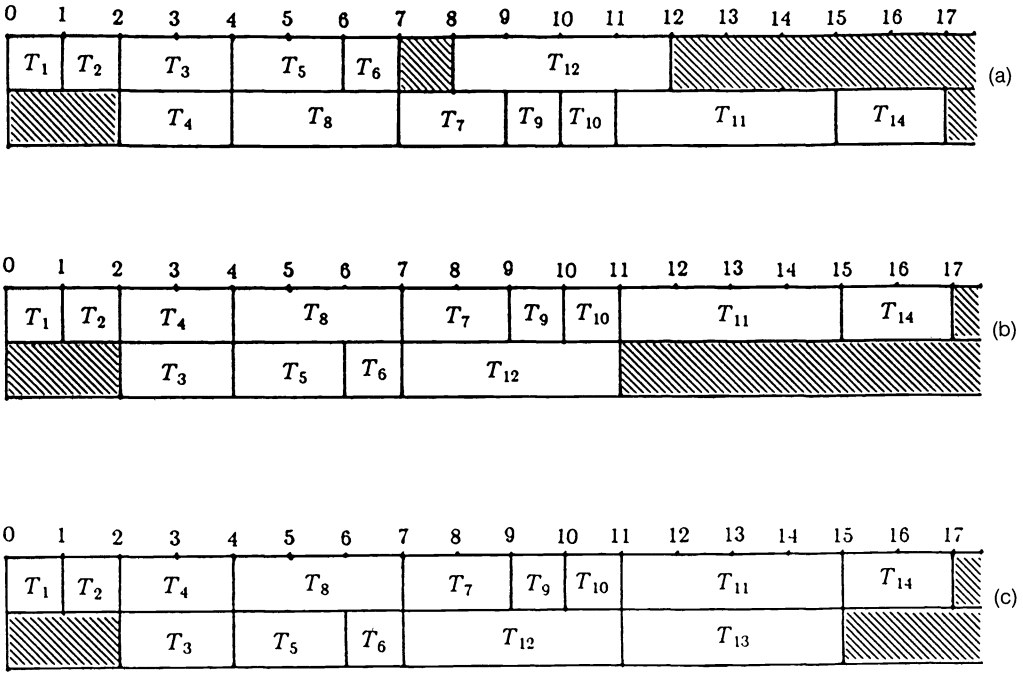


FIG. 9. *The final schedule* S_D: (a) *the schedule* S_{M1}; (b) *the schedule* S_{M2}; (c) *the schedule* S_{M3}.

2. Let T_j be a preempted task in the innermost level, and let t_1 and t_2 be the starting and finishing times of T_j in S_i, respectively. Divide S_i into three partial schedules S_L, S_M, and S_R such that they consist of the intervals $(0, t_1)$, (t_1, t_2), and (t_2, ω_i), respectively. Let S_{M1} be the schedule obtained by removing T_j from S_M. $S_{M2} \leftarrow$ REARRANGE(S_{M1}). $S_{M3} \leftarrow$ INSERT(T_j, S_{M2}). $S_{i+1} \leftarrow$ CONCATENATE(S_L, S_{M3}, S_R). Let ω_{i+1} be the length of S_{i+1}. $i \leftarrow i + 1$.

A schedule constructed by Algorithm D is called a D-*schedule*, denoted by S_D, and its length is denoted by ω_D. Figures 8 and 9 show the schedules obtained after inserting T_8 and T_{13}, respectively. Since T_8 and T_{13} are the only preempted tasks, the schedule shown in Fig. 9(c) is the final D-schedule.

Before we give the REARRANGE procedure, we must define additional notation. Let S be a nonpreemptive schedule, and let (a_1, b_1), (a_2, b_2), $\cdots$, (a_k, b_k) be the sequence of all idle slots in S. S is called a *proper* schedule if (1) there are no idle slots on P_1, (2) for each $1 \leqq i < k$, two tasks start at b_i, and (3) if $b_k < \omega_S$, then there is a task T that starts in the interval (a_k, b_k). In the INSERT procedure given later, we assume that the schedule in hand is a proper schedule. Thus the REARRANGE procedure must generate a proper schedule. If S is a proper schedule, an interval (t_1, t_2) on P_2 is called a *task block* if it is a maximal interval during which tasks are executing. Thus t_1 is either the beginning of the schedule or the right end of an idle slot, and t_2 is either the end of the schedule or the left end of an idle slot. The proof of the next theorem gives the REARRANGE procedure.

THEOREM 13. *Let* S_M *be the middle subschedule when the preempted task* T_j *is considered for insertion in Algorithm* D. *Let* T_1 *and* T_2 (*if they exist*) *be two tasks such that* T_1 *starts at the beginning and* T_2 *finishes at the end of* S_M. *Then there is a proper schedule* S_{M2} *such that* T_1 *starts at the beginning and* T_2 *finishes at the end of* S_{M2}.

Proof. Let S_{M1} be obtained from S_M by removing T_j. We obtain S_{M2} from S_{M1} as follows. By scanning S_{M1} from left to right, we shift the tasks in S_{M1} as far left as possible.

Let S_{M2} denote this new schedule and let ω_{M2} be its length. By interchanging processors within each segment if necessary, we can make all idle slots to be on P_2. Clearly, S_{M2} is a normalized schedule, and T_1 starts at the beginning of S_{M2}. If T_2 finishes at the end of S_{M2}, then S_{M2} is the desired schedule. Otherwise, we shift T_2 right until it finishes at ω_{M2}. Note that T_2 is in the last segment with no tasks following it. Furthermore, T_2 is executed on P_2. The shifting of T_2 shifts the idle slot in the last segment to an earlier position. Let (a, b) be this new idle slot. If there is a task that starts in (a, b), then we have the desired schedule. Otherwise, we shift the task that finishes at a on P_2 to the right until its finishing time becomes b. Shifting this task again shifts the idle slot to an earlier position. We repeat the above process until we obtain the desired schedule. Note that the shifting process must stop at the beginning of the last segment, since two tasks start at that point. □

LEMMA 15. *Let τ be a task system such that each task has integer execution time. If S is a proper schedule for τ, then each idle slot in S has integer length.*

Proof. The proof is immediate from the definition of proper schedules. □

THEOREM 14. *Let S_{M2} be the schedule as in Algorithm* D. *Then each idle slot in S_{M2} has integer length.*

Proof. Since S_{M2} is a proper schedule, the theorem is proved by showing that each task executed in S_{M2} has integer execution time. This follows immediately from property 3 in Theorem 9. □

Our next goal is to give the INSERT procedure. Let T_j be the task to be inserted into the schedule S_{M2}. We show in Theorem 15 that, if $e(T_j) \leqq E_{S_{M2}}$, $e(T_j) \leqq 4$, and $I_{S_{M2}} \leqq C_{S_{M2}} + e(T_j)$, then there is a nonpreemptive schedule S_{M3} for T_j and the tasks in S_{M2} such that $I_{S_{M3}} \leqq C_{S_{M3}}$. We then show in Theorem 16 that the conditions $e(T_j) \leqq E_{S_{M2}}$ and $I_{S_{M2}} \leqq C_{S_{M2}} + e(T_j)$ hold at each iteration of Algorithm D. To show the above results, we must introduce additional notation. Let $\tau = (TS, G, e)$ be a task system. $\tau' = (TS', G', e')$ is a *restriction* of τ if (1) TS' is a subset of TS, (2) G' is a subgraph of G induced by TS', and (3) e' is a restriction of e to TS'. Lemma 17 is instrumental in proving Theorem 15. First, we must prove the following lemma.

LEMMA 16. *Let $\tau_1 = (TS_1, G_1, e_1)$ be a restriction of $\tau_2 = (TS_1 \cup \{T\}, G_2, e_2)$ such that T is independent with every task in TS_1. Let S_1 (S_2) be a nonpreemptive schedule for τ_1 (τ_2), and let ω_1 (ω_2) be its length. Let C_1 (C_2) and I_1 (I_2) be the concurrency and total idle time of S_1 (S_2), respectively. Let $d = (C_1 + 2e(T) - I_1)/3$. If $\omega_1 - \omega_2 \leqq d$, then $I_2 \leqq C_2$.*

Proof. Clearly, we have that $E(TS_2) = E(TS_1) + e(T)$, $\omega_1 = C_1 + I_1$, $\omega_2 = C_2 + I_2$, $E(TS_1) = 2C_1 + I_1$, and $E(TS_2) = 2C_2 + I_2$. Thus $\omega_2 = C_2 + I_2 = E(TS_2) - C_2 = E(TS_1) + e(T) - C_2$, and $\omega_1 = C_1 + I_1 = E(TS_1) - C_1$. Since $\omega_2 - \omega_1 \leqq d$, we have that $C_2 \geqq e(T) + C_1 - d = E(TS_2)/3$. Therefore $I_2 = E(TS_2) - 2C_2 \leqq C_2$. □

LEMMA 17. *Let $\tau_1 = (TS_1, G_1, e_1)$ be a restriction of $\tau_2 = (TS_1 \cup \{T\}, G_2, e_2)$ such that T is independent with every task in TS_1. If $e(T) \leqq E(TS_1)$ and if there is a proper schedule S_1 for τ_1 such that $I_1 \leqq e(T)$, then there is a nonpreemptive schedule S_2 for τ_2 such that $I_2 \leqq C_2$.*

Proof. If $e(T) \geqq E(TS_1)/2$, we can obtain S_2 by scheduling all tasks in TS_1 on P_1 and T on P_2. Then $C_2 = e(T)$ and $I_2 = E(TS_1) - e(T)$. Since $E(TS_1) \leqq 2e(T)$, we have that $I_2 \leqq e(T) = C_2$. Thus we may assume that $e(T) < E(TS_1)/2$. Let S_1 be a proper schedule for τ_1, and let C_1 and I_1 be its concurrency and total idle time, respectively. Then we have that $I_1 \leqq e(T)$. Since $E(TS_1) = 2C_1 + I_1$ and $2e(T) < E(TS_1)$, we have that

$$(*) \qquad 2e(T) < 2C_1 + I_1.$$

Let $d = (C_1 + 2e(T) - I_1)/3$. By Lemma 16, if T can be inserted into S_1 such that the schedule length is increased by at most d, then the resulting schedule has concurrency not less than its total idle time. Thus, if $e(T) \leqq d$, then we can append T at the end of S_1, and the length of the resulting schedule is increased by at most d. Thus we may assume that $e(T) > d = (C_1 + 2e(T) - I_1)/3$ or, equivalently,

$$(**) \qquad e(T) > C_1 - I_1.$$

We now show that T can still be inserted into S_1 such that the resulting schedule has length increased by at most d under conditions $(*)$ and $(**)$. Let (a_1, b_1), $(a_2, b_2), \cdots, (a_k, b_k)$ be all idle slots in S_1. Let i be the smallest index such that $b_i \geqq e(T) - d$, and let D be the total amount of tasks in the interval $(0, b_i)$ on P_2. We consider the following two cases.

Case A. We have that $D \leqq d$.

In this case, we obtain S_2 from S_1, as follows. All tasks in the interval $(0, b_i)$ on P_2 are moved to P_1. If the new idle slot is not large enough for T, then the idle slot is expanded such that its length becomes $e(T)$. Schedule T in this idle slot. Since $D \leqq d$ and since $b_i \geqq e(T) - d$, or, equivalently, $e(T) \leqq b_i + d$, the schedule length is increased by at most d.

Case B. We have that $D > d$.

Let D' be the total amount of tasks in the interval (b_i, ω_{S_1}) on P_2. Note that $C_1 = D + D'$. Thus $D' = C_1 - D < C_1 - d = (2C_1 - 2e(T) + I_1)/3 \leqq (2C_1 - 2I_1 + I_1)/3$ (since $I_1 \leqq e(T)$) $= (2C_1 - I_1)/3 < (C_1 + e(T))/3$ (by $(**)$) $\leqq (C_1 + 2e(T) - I_1)/3$ (since $I_1 \leqq e(T)$) $= d$. Therefore, if we can show that $\omega_{S_1} - a_i \geqq e(T) - d$, then we can insert T into the interval (a_i, ω_{S_1}) just as we did in Case A. We prove the contradiction that $\omega_{S_1} - a_i \geqq e(T) - d$. If $\omega_{S_1} - a_i < e(T) - d$, then $a_i > \omega_{S_1} - e(T) + d$. Since i is the smallest index such that $b_i \geqq e(T) - d$, we have that $b_{i-1} < e(T) - d$. Note that the interval (b_{i-1}, a_i) on P_2 is a task block with length $a_i - b_{i-1}$. However, $a_i - b_{i-1} > (\omega_{S_1} - e(T) + d) - (e(T) - d) = \omega_{S_1} - 2e(T) + 2d = C_1 + I_1 - 2e(T) + 2(C_1 + 2e(T) - I_1)/3 = (5C_1 - 2e(T) + I_1)/3 > (5C_1 - (2C_1 + I_1) + I_1)/3$ (by $(*)$) $= C_1$. This is the contradiction we seek, since the task block has length greater than C_1. □

We are now ready to give the INSERT procedure. The proof of the following theorem describes the procedure.

THEOREM 15. *Let T_j be the task to be inserted into S_{M2} in Algorithm* D. *Let C_{M2} and I_{M2} denote the concurrency and total idle time in S_{M2}, respectively. If $e(T_j) \leqq E_{S_{M2}}$, $e(T_j) \leqq 4$, and $I_{M2} \leqq C_{M2} + e(T_j)$, then there is a nonpreemptive schedule S_{M3} for the tasks in S_{M2} and T_j such that $I_{M3} \leqq C_{M3}$, where C_{M3} and I_{M3} denote the concurrency and total idle time in S_{M3}, respectively.*

Proof. Note that S_{M2} is a proper schedule by Theorem 13 and that each idle slot in S_{M2} has integer length by Theorem 14. Furthermore, T_j is independent with every task in S_{M2}. If $I_{M2} \leqq e(T_j)$, then the theorem holds by Lemma 17. Thus we may assume that $e(T_j) < I_{M2}$. Since $e(T_j) \leqq 4$, we consider the following four cases.

Case (i). We have that $e(T_j) = 1$.

In this case, we have that $I_{M2} \leqq C_{M2} + 1$. Since each idle slot has length at least 1 unit, we can obtain S_{M3} by inserting T_j into any idle slot in S_{M2}. Then we have that $C_{M3} = C_{M2} + 1$ and $I_{M3} = I_{M2} - 1$. Since $I_{M2} \leqq C_{M2} + 1$, we have that $I_{M3} \leqq C_{M3}$.

Case (ii). We have that $e(T_j) = 2$.

In this case, we have that $I_{M2} \leqq C_{M2} + 2$. If there is an idle slot in S_{M2} with length at least 2 units, then we can obtain S_{M3} by the same method as in Case (i). Thus we may assume that each idle slot has length 1 unit. Since C_{M2} equals the sum of the lengths of all task blocks, C_{M2} is at least the number of task blocks. Note that the number of idle

slots is at most one more than the number of task blocks. Thus we have $I_{M2} \leqq C_{M2} + 1$. To obtain S_{M3}, we expand an arbitrary idle slot (a, b) in S_{M2} by 1 unit. This is done by right shifting all tasks executed after b by 1 unit. Then T_j is inserted into the new idle slot, which has a length of 2 units. Clearly, we have that $C_{M3} = C_{M2} + 1$ and $I_{M3} = I_{M2}$. Since $I_{M2} \leqq C_{M2} + 1$, we have that $I_{M3} \leqq C_{M3}$.

Case (iii). We have that $e(T_j) = 3$.

In this case, we have that $I_{M2} \leqq C_{M2} + 3$. If there is an idle slot in S_{M2} with length at least 3 units, then we can obtain S_{M3} as in Case (i). On the other hand, if there is an idle slot in S_{M2} with length 2 units, then we can expand the idle slot by 1 unit and insert T_j as in Case (ii). Thus we may assume that each idle slot has length 1 unit. We consider the following two cases. If there is a task block with length 1 unit, then we can move the task in the task block to P_1, thereby creating a new idle slot with length 3 units or more. We can now schedule T_j in this idle slot. Clearly, $C_{M3} = C_{M2} + 2$ and $I_{M3} = I_{M2} - 1$. Since $I_{M2} \leqq C_{M2} + 3$, we have that $I_{M3} \leqq C_{M3}$. On the other hand, if each task block has length 2 units or more, then C_{M2} must be at least twice the number of task blocks. Since I_{M2} is at most one more than the number of task blocks, we have that $C_{M2} \geqq 2(I_{M2} - 1)$, or, equivalently, $C_{M2} + 2 \geqq 2I_{M2}$. Since $I_{M2} > e(T_j) = 3$, we have that $C_{M2} > I_{M2} + 1$. We can expand an arbitrary idle slot by 2 units and insert T_j into this new idle slot. Clearly, we have that $C_{M3} = C_{M2} + 1$ and $I_{M3} = I_{M2} + 1$. Since $C_{M2} > I_{M2} + 1$, we have that $I_{M3} \leqq C_{M3}$.

Case (iv). We have that $e(T_j) = 4$.

In this case, we have that $I_{M2} \leqq C_{M2} + 4$. If there is an idle slot in S_{M2} with length 4 units or more, then we can obtain S_{M3} as in Case (i). Similarly, if there is an idle slot with length 3 units, then we can expand the idle slot by 1 unit as in Case (ii). Thus we may assume that each idle slot has length 2 units or less. If there is a task block with length 1 unit between two idle slots, then we can move the task in the task block to P_1, thereby creating an idle slot with length at least 4 units. T_j is inserted into the new idle slot. Clearly, we have that $C_{M3} = C_{M2} + 3$ and $I_{M3} = I_{M2} - 2$. Since $I_{M2} \leqq C_{M2} + 4$, we have that $I_{M3} \leqq C_{M3}$. Thus we may assume that each task block between two idle slots has length more than 1 unit. Therefore C_{M2} must be at least twice the number of task blocks between two idle slots. The number of task blocks between two idle slots is exactly one less than the number of idle slots. Since each idle slot has length 2 units or less, I_{M2} is no more than twice the number of idle slots. Consequently, we have that $C_{M2} \geqq I_{M2} - 2$ or, equivalently, $I_{M2} \leqq C_{M2} + 2$. If there is an idle slot with length 2 units, then we can expand the idle slot by 2 units and insert T_j into this new idle slot. Clearly, we have that $C_{M3} = C_{M2} + 2$ and $I_{M3} = I_{M2}$. Since $I_{M2} \leqq C_{M2} + 2$, we have that $I_{M3} \leqq C_{M3}$. On the other hand, if each idle slot has length 1 unit, then I_{M2} is exactly the number of idle slots. Consequently, we have that $C_{M2} \geqq 2I_{M2} - 2$. Since $I_{M2} > e(T_j) = 4$, we have that $C_{M2} > I_{M2} + 2$. We can now expand an arbitrary idle slot by 3 units and insert T_j into a new idle slot. Clearly, we have that $C_{M3} = C_{M2} + 1$ and $I_{M3} = I_{M2} + 2$. Since $C_{M2} > I_{M2} + 2$, we have that $I_{M3} \leqq C_{M3}$. □

THEOREM 16. *Let T_j be the task to be inserted into S_{M2} in Algorithm* D. *Let C_{M2} and I_{M2} denote the concurrency and total idle time of S_{M2}, respectively. Then we have that $e(T_j) \leqq E_{S_{M2}}$ and $I_{M2} \leqq C_{M2} + e(T_j)$.*

Proof. Let T_j be a preempted task in the innermost level in the C-schedule. By property 3 in Theorem 8, we have that $e(T_j) \leqq E_{S_{M2}}$. Let C_M and I_M denote the concurrency and total idle time of S_M, respectively. Let $e'(T_j)$ denote the double amount of T_j in S_M. From the nature of Algorithm C, we have that $I_M \leqq C_M - e'(T_j) - (e(T_j) - 2e'(T_j))$. Let C_{M1} and I_{M1} denote the concurrency and total idle time of S_{M1}, respectively. Since S_{M1} is obtained from S_M by removing T_j, we have that $I_{M1} = I_M +$

$(e(T_j) - 2e'(T_j))$ and $C_{M1} = C_M - e'(T_j) - (e(T_j) - 2e'(T_j))$. Thus we have that $I_{M1} \leqq C_{M1} + e(T_j)$. Since S_{M2} is obtained by rearranging S_{M1}, we have that $C_{M2} \geqq C_{M1}$ and $I_{M2} \leqq I_{M1}$. Thus we have that $I_{M2} \leqq C_{M2} + e(T_j)$. If T_j is not in the innermost level, then, by induction on the level of nestedness, we can show that $e(T_j) \leqq E_{S_{M2}}$ and $I_{M2} \leqq C_{M2} + e(T_j)$. □

We now describe the procedure CONCATENATE. The purpose of CONCATENATE is to concatenate together the subschedules S_L, S_{M3}, and S_R to form S_{i+1}. In Algorithm D, the schedule S_i is divided into three subschedules S_L, S_M, and S_R. It is possible that there is a task T_1 executed in both S_L and S_M. Similarly, there may be a task T_2 executed in both S_M and S_R. When S_i is divided into three portions, T_1 and T_2 are cut at the boundaries. The CONCATENATE procedure must ensure that T_1 and T_2 are not preempted at the boundaries in S_{i+1}. By the REARRANGE and INSERT procedures, T_1 and T_2 can always be rearranged so that T_1 starts at the beginning and T_2 finishes at the end of S_{M3}. Thus, by interchanging processors in S_L if necessary, we can arrange T_1 to be executed on the same processor in S_L as in S_{M3}. Similarly, T_2 can also be arranged so that it is executed on the same processor in S_R as in S_{M3}. Thus T_1 and T_2 are not preempted at the boundaries in S_{i+1}.

THEOREM 17. *Let τ be a task system such that $e(T_j) \in \{1, 2, 3, 4\}$ for each T_j. Then we have that $\omega_N/\omega_P \leqq \omega_D/\omega_P \leqq 4/3$.*

Proof. The proof is immediate from Theorems 15 and 16 and Lemma 12. □

5. Conclusion. In this paper, we have shown that Liu's bound is valid for UET and tree-structured task systems. For two processors, we show that the 4/3 bound holds for simple task systems, as well as task systems with execution times drawn from the set $\{1, 2, 3, 4\}$. We note that all of our proofs are constructive. Thus they can be used as algorithms with a guaranteed performance bound. For future research, it is clearly desirable to settle Liu's conjecture for arbitrary task systems on arbitrary number of processors. However, we feel that the problem might be too difficult to solve. A less ambitious goal is to settle the 4/3 bound for arbitrary task systems on two processors. In this regard, we note that all of our results in § 4.2, except Algorithm D, are also applicable to arbitrary execution times. A more powerful insertion technique is needed if our technique proves to be successful.

REFERENCES

[1] E. G. COFFMAN, JR., ED., *Computer and Job-Shop Scheduling Theory*, John Wiley, New York, 1976.

[2] E. G. COFFMAN, JR. AND R. L. GRAHAM, *Optimal scheduling for two-processor systems*, Acta Inform., 1 (1972), pp. 200–213.

[3] D. K. GOYAL, *Non-preemptive scheduling of unequal execution time tasks on two identical processors*, Tech. Report CS-77-039, Computer Science Department, Wash. State University, Pullman, WA, 1977.

[4] R. L. GRAHAM, *Bounds on multiprocessing timing anomalies*, SIAM J. Appl. Math., 17 (1969), pp. 416–429.

[5] K. S. HONG, *On Some Scheduling Problems*, Ph.D. thesis, Northwestern University, Evanston, IL, 1989.

[6] M. T. KAUFMAN, *An almost-optimal algorithm for the assembly-line scheduling problem*, IEEE Trans. Comput., C-23 (1974), pp. 1169–1174.

[7] M. KUNDE, *Nonpreemptive* LP*-scheduling on homogeneous multiprocessor systems*, SIAM J. Comput., 10 (1981), pp. 151–173.

[8] S. LAM AND R. SETHI, *Worst case analysis of two scheduling algorithms*, SIAM J. Comput., 6 (1977), pp. 518–536.

[9] C. L. LIU, *Optimal scheduling on multiprocessor computing systems*, in Proc. 13th Ann. Sympos. on Switching and Automata Theory, IEEE Computer Society, 1972, pp. 155–160.

[10] R. R. MUNTZ AND E. G. COFFMAN, JR., *Optimal preemptive scheduling on two-processor systems*, IEEE Trans. Comput., C-18 (1969), pp. 1014–1020.

[11] J. D. ULLMAN, NP*-complete scheduling problems*, J. Comput. Systems Sci., 10 (1975) pp. 151–173.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 524–538, November 1992
© 1992 Society for Industrial and Applied Mathematics
008

2-COMPETITION GRAPHS*

GARTH ISAAK†, SUH-RYUNG KIM‡[2], TERRY A. MCKEE§, F. R. MCMORRIS¶, AND FRED S. ROBERTS[1,2]

Abstract. If $D = (V, A)$ is a digraph, its p-competition graph for p a positive integer has vertex set V and an edge between x and y if and only if there are distinct vertices $a_1, \cdots, a_p$ in D with (x, a_i) and (y, a_i) arcs of D for each $i = 1, \cdots, p$. This notion generalizes the notion of ordinary competition graph, which has been widely studied and is the special case where $p = 1$. Results about the case where $p = 2$ are obtained. In particular, the paper addresses the question of which complete bipartite graphs are 2-competition graphs. This problem is formulated as the following combinatorial problem: Given disjoint sets A and B such that $|A \cup B| = n$, when can one find n subsets of $A \cup B$ so that every a in A and b in B are together contained in at least two of the subsets and so that the intersection of every pair of subsets contains at most one element from A and at most one element from B?

Key words. competition graphs, edge clique coverings, food webs, complete bipartite graphs, set coverings

AMS(MOS) subject classifications. 05C90, 05C99, 05D05, 92D40

1. Introduction. Suppose that $D = (V, A)$ is a digraph, loops allowed. (For all undefined graph theory terminology, see [1], [9].) If p is a positive integer, the *p-competition graph* corresponding to D, $C_p(D)$, is defined to have vertex set V and to have an edge between x and y in V if and only if, for some distinct $a_1, \cdots, a_p$ in V, $(x, a_1), (y, a_1), (x, a_2), (y, a_2), \cdots, (x, a_p), (y, a_p)$ are in A. This concept was introduced in [5] as a generalization of the special case where $p = 1$, which has been studied by many authors.[3] The 1-competition graphs were motivated by a problem in ecology and have applications to a variety of fields, as summarized in [8]. The p-competition graphs have a similar motivation and similar applications to other fields. The ecological motivation is as follows: The vertices of D are considered species in an ecosystem, and there is an arc from species x to species a if x preys on a. Then x and y are joined by an edge in the p-competition graph if and only if they have at least p common prey. The literature of 1-competition graphs, otherwise known as *competition graphs*, is summarized in [4], [6], and [8]. In this paper, we study the special case where $p = 2$.

It is easy to reduce the study of p-competition graphs to a combinatorial problem that itself is of interest. Suppose that G is a graph and that $F = \{S_1, \cdots, S_r\}$ is a family of subsets of the vertex set of G, repetitions allowed. We say that F is a *p-edge clique covering*, or p-ECC, if, for every set of p distinct subscripts $i_1, i_2, \cdots, i_p$, $T = S_{i_1} \cap S_{i_2} \cap \cdots \cap S_{i_p}$ is either empty or induces a clique of G, and the collection of sets of the

* Received by the editors January 26, 1990; accepted for publication (in revised form) July 25, 1991.

† Department of Mathematics and Computer Science, Dartmouth College, Hanover, New Hampshire 03755.

‡ Department of Mathematics, St. Johns University, Staten Island, New York 10301.

§ Department of Mathematics and Statistics, Wright State University, Dayton, Ohio 45435. This author's research was supported by Office of Naval Research grant N00014-88-K-0163.

¶ Department of Mathematics, University of Louisville, Louisville, Kentucky 40292. This author's research was supported by Office of Naval Research grant N00014-89-J-1643.

[1] Department of Mathematics and Rutgers Center for Operations Research, Rutgers University, New Brunswick, New Jersey 08903.

[2] This author's research was supported by Air Force Office of Scientific Research grants AFOSR-89-0066, AFOSR-89-0512, and AFOSR-90-0008.

[3] Those who are familiar with the competition graphs literature will note that we do not assume that the digraph D is acyclic, as is often assumed in that literature.

form T covers all edges of G. Let $\theta_e^p(G)$ be the smallest r for which there is a p-ECC F. (A 1-ECC is an ordinary edge clique covering. Edge clique coverings have played a central role in the theory of competition graphs; cf. [10].)

THEOREM 1 (see [5]). *A graph G with n vertices is a p-competition graph if and only if $\theta_e^p(G) \leqq n$.*

Proof. Suppose that $G = C_p(D)$, where $D = (V, A)$, and let $V(G) = \{v_1, \cdots, v_n\}$. For each i, let $S_i = \{v_j : (v_j, v_i) \in A\}$. It is easy to verify that the family of S_i is a p-ECC. Conversely, suppose that G and a p-ECC $F = \{S_1, \cdots, S_r\}$, $r \leqq n$, are given. Now define $D = (V, A)$ on $V = V(G)$ by letting $(v_i, v_j) \in A$ if and only if $v_i \in S_j$. It is easy to verify that $G = C_p(D)$. □

COROLLARY. *A graph G with n vertices is a p-competition graph if and only if G has a p-ECC consisting of n sets.*

Proof. Suppose that F is a p-ECC of $r < n$ sets. Since repetitions are allowed in F, we can add $n - r$ copies of the empty set to F to obtain a p-ECC of size n. □

Kim et al. [5] obtain a number of results about p-competition graphs in general; for example, they extend the basic results about ordinary competition graphs obtained in [2], [7], and [11]. They also obtain a variety of results about 2-competition graphs. For instance, they show that all trees are 2-competition graphs, all unicyclic graphs are 2-competition graphs except the 4-cycle C_4, and all chordal graphs are 2-competition graphs. In this paper, we study the question: What complete bipartite graphs are 2-competition graphs?

A graph $G = K_{m,x}$ is a *complete bipartite graph* if the vertices are partitioned into a pair of disjoint sets A and B of m and x vertices, respectively, and there is an edge between two vertices if and only if they are in different sets. By virtue of the corollary to Theorem 1, the question of whether $K_{m,x}$ is a 2-competition graph is reduced to the combinatorial question: If $|A \cup B| = m + x = n$, are there n subsets of $A \cup B$ (not necessarily distinct) so that (i) for all $a \in A$ and $b \in B$, a and b are contained in at least two sets, and (ii) each pair of elements from A appears together in at most one set, and similarly for each pair of elements from B? In § 2 we study this question for general m and x, showing that for fixed m, there are real numbers $a(m) < b(m) < c(m)$ so that $K_{m,x}$ is not a 2-competition graph for $x \in [a(m), b(m)]$ and $K_{m,x}$ is a 2-competition graph for $x \geqq c(m)$. In § 3 we answer the question entirely for the special case where $m = 2$. In §§ 4 and 5 we consider the special cases where $m = 3$ and $m = x$. Finally, § 6 gives closing remarks and open questions.

2. Fixed m and arbitrary x. In this section, we study $K_{m,x}$ for arbitrary m and x. We show that for fixed m, $K_{m,x}$ is a 2-competition graph for all x sufficiently large. However, when m is sufficiently large (at least 24), we show that there is an interval of intermediate values of x for which $K_{m,x}$ is not a 2-competition graph. We do not yet have evidence to dispute the conjecture that, for all $x \geqq m \geqq 2$, if $K_{m,x}$ is a 2-competition graph, then so is $K_{m,x+1}$. Our results in the next section do prove this conjecture for $m = 2$ (though we do not have a direct proof).

THEOREM 2. *For every $m \geqq 1$, $K_{m,x}$ is a 2-competition graph for all x sufficiently large.*

Proof. Given m and x, let y be an integer such that $x \leqq y^{2m}$, let C be the set of all $(2m)$-tuples $c = (c_1, \cdots, c_{2m})$ with entries from $\{1, 2, \cdots, y\}$, and let C' be the set of all $(2m-1)$-tuples $d = (d_1, \cdots, d_{2m-1})$ with entries from $\{1, 2, \cdots, y\}$. Let B be any subset of C with $|B| = x$. Note that $|C| = y^{2m}$ and $|C'| = y^{2m-1}$. Let $G = K_{m,x}$ have one independent set $\{u_1, \cdots, u_m\}$ and the other independent set B. Build a 2-ECC for G as follows. Given $1 \leqq i \leqq 2m$ and c in B, define $c/i = (c_1, \cdots, c_{i-1}, c_{i+1}, \cdots, c_{2m})$.

Note that c/i is in C'. Given $1 \leqq i \leqq 2m$ and given d in C', let

$$S_d^i = \{u_{\lceil i/2 \rceil}\} \cup \{c \in B: c/i = d\}.$$

Thus the family of all such sets has $2my^{2m-1}$ members. To see why they form a 2-ECC, first fix u_i. Then any c in B appears with u_i in $S_{c/(2i-1)}^{2i-1}$ and $S_{c/2i}^{2i}$. Also, u_i and u_j never appear together if $i \neq j$. Finally, consider $c \neq c'$ in B. Then there is some i, so that $c_i \neq c_i'$. It follows that when $j \neq i$, then, for any d in C', either c or c' is not in S_d^j. We next show that c and c' appear together in at most one S_d^i.

Case 1. For some $j \neq i$, $c_j \neq c_j'$. Here for every d in C', either c or c' is not in S_d^i.

Case 2. For all $j \neq i$, $c_j = c_j'$. Here $c/i = c'/i$ and c and c' both appear in the set $S_{c/i}^i = S_{c'/i}^i$. However, whenever d in C' is different from $c/i = c'/i$, neither c nor c' is in S_d^i.

We conclude that G is a 2-competition graph as long as the number of sets in the family is at most the number of vertices of G; i.e.,

$$2my^{2m-1} \leqq m + x$$

or

$$2my^{2m-1} - m \leqq x. \tag{1}$$

Thus we have shown that $K_{m,x}$ is a 2-competition graph whenever

$$2my^{2m-1} - m \leqq x \leqq y^{2m}, \tag{2}$$

i.e., whenever x belongs to the interval

$$I_y = [2my^{2m-1} - m, y^{2m}].$$

Note that $I_y \neq \varnothing$ if $y \geqq 2m$. Note also that, if y is sufficiently large, say $y \geqq Y$ (where $Y \geqq 2m$), then

$$2m(y+1)^{2m-1} \leqq y^{2m},$$

and therefore

$$2m(y+1)^{2m-1} - m \leqq y^{2m}.$$

Thus, for all $y \geqq Y$, the intervals I_y and I_{y+1} overlap. It follows that, for all $x \geqq 2mY^{2m-1} - m$, $K_{m,x}$ is a 2-competition graph. □

COROLLARY. *$K_{m,y^{2m}}$ is a 2-competition graph whenever $m \geqq 1$ and $y \geqq 2m$.*

Proof. By the proof, $K_{m,x}$ is a 2-competition graph as long as (2) holds. However, (2) holds if $x = y^{2m}$ and $y \geqq 2m$. □

We now introduce the following notation, which we use throughout this section. Let $A = \{a_1, \cdots, a_m\}$ and $B = \{b_1, \cdots, b_x\}$. Let $S_1, \cdots, S_t$ be a 2-ECC for $K_{m,x}$ with bipartition A and B. Suppose that v_j is the number of sets S_i containing b_j and $v = \min v_j$. (In calculating v_j, if a set S_i and a set S_k are the same for $i \neq k$, we count them both.)

LEMMA 3. *It holds that*

$$v \geqq \frac{1 + \sqrt{1 + 8m}}{2}.$$

Proof. Given j, note that, for each i, there are two distinct subscripts $\alpha(i)$ and $\beta(i)$, so that sets $S_{\alpha(i)}$ and $S_{\beta(i)}$ both contain a_i and b_j. The pairs $\{S_{\alpha(i)}, S_{\beta(i)}\}$ are all different

because, if $\{S_{\alpha(i)}, S_{\beta(i)}\} = \{S_{\alpha(k)}, S_{\beta(k)}\}$ for $i \neq k$, then a_i and a_k are in two sets together, which is impossible. It follows that

$$\binom{v_j}{2} \geqq m,$$

and so $\binom{v}{2} \geqq m$. Thus $v^2 - v - 2m \geqq 0$. Using the quadratic formula and the fact that

$$\frac{1-\sqrt{1+8m}}{2} < 0 \leqq v,$$

we conclude that $v \geqq (1 + \sqrt{1+8m})/2$. □

Remark. Suppose that u_i is the number of sets S_j containing a_i (again, with multiple counting as for v_j) and $u = \min u_i$. By symmetry,

$$u \geqq \frac{1+\sqrt{1+8x}}{2}.$$

LEMMA 4. *If $S_1, \cdots, S_t$ is a* 2-ECC *for $K_{m,x}$, then $t \geqq v^2x/(v + x - 1)$.*

Proof. We may assume that all of the S_i are nonempty and contain an element of B. Otherwise, we remove the empty sets and those not containing elements of B, and we still have a 2-ECC with the same v; the result follows for the original 2-ECC from the result for the new 2-ECC. Let $S_i' = S_i \cap B$. Thus we may assume that all of the S_i' are nonempty.

We next note that since each b_j is in at least v sets S_i', we have that

$$\sum_{i=1}^{t} |S_i'| \geqq xv. \tag{3}$$

Note that no pair of elements from B is together in more than one set S_i', and so

$$\sum_{i=1}^{t} \binom{|S_i'|}{2} \leqq \binom{x}{2}. \tag{4}$$

Using the Cauchy–Schwartz inequality and (4), we have that

$$\frac{1}{t}\left[\sum_{i=1}^{t} |S_i'|\right]^2 - \sum_{i=1}^{t} |S_i'| \leqq \sum_{i=1}^{t} |S_i'|^2 - \sum_{i=1}^{t} |S_i'| \leqq x^2 - x. \tag{5}$$

Since no S_i' is empty, $\sum_{i=1}^{t} |S_i'| \geqq t$. Then, by (5) and (3),

$$tx(x-1) \geqq \sum_{i=1}^{t} |S_i'| \left(\sum_{i=1}^{t} |S_i'| - t\right) \geqq xv(xv - t);$$

so $t \geqq v^2x/(v + x - 1)$. □

Remark. By symmetry, if u is defined as in the Remark after Lemma 3, we have that $t \geqq u^2m/(u + m - 1)$.

Remark. It follows from Lemma 4 and Theorem 1 that $K_{m,x}$ is not a 2-competition graph if

$$m + x < \frac{v^2x}{v+x-1}. \tag{6}$$

By symmetry, the same conclusion holds if

$$m+x<\frac{u^2m}{u+m-1}.$$

THEOREM 5. *For fixed $m \geqq 24$, $K_{m,x}$ is not a 2-competition graph if*

$$x\in\left[\frac{m+1}{2}-\frac{\sqrt{m^2+4m+1-2m\sqrt{1+8m}}}{2},\frac{m+1}{2}+\frac{\sqrt{m^2+4m+1-2m\sqrt{1+8m}}}{2}\right].$$

Proof. Let $\gamma = \sqrt{1+8m}$. By Lemma 3, $v \geqq (1+\gamma)/2$. Since $x \geqq 1$ (by tacit assumption) and $v \geqq 1$,

$$f_x(v) = \frac{v^2x}{v+x-1}$$

is increasing in v for fixed x. (This is easy to check by taking the first derivative.) It follows that, since $v \geqq (1+\gamma)/2$,

$$\frac{\left[\frac{1+\gamma}{2}\right]^2 x}{\frac{1+\gamma}{2}+x-1} \leqq \frac{v^2x}{v+x-1}.$$

Hence, if we can show that

$$m+x<\frac{\left[\frac{1+\gamma}{2}\right]^2 x}{\frac{1+\gamma}{2}+x-1}, \tag{7}$$

then (6) follows. However, since $x \geqq 1$ and since

$$\left[\frac{1+\gamma}{2}\right]^2 = \frac{1+\gamma}{2}+2m,$$

we see by cross-multiplying that (7) holds if and only if

$$F(x) = x^2+(-1-m)x+\left[\frac{\gamma-1}{2}\right]m<0.$$

Thus, for given m, this holds if x is between the roots of the quadratic $F(x)$, namely,

$$\frac{1+m\pm\sqrt{m^2+2m+1-2(\gamma-1)m}}{2} = \frac{m+1}{2}\pm\frac{\sqrt{m^2+4m+1-2m\sqrt{1+8m}}}{2}.$$

This proves the desired result. Note that the hypothesis $m \geqq 24$ is needed for

$$m^2+4m+1-2m\sqrt{1+8m}$$

to be nonnegative, and hence for

$$\sqrt{m^2+4m+1-2m\sqrt{1+8m}}$$

to be defined (to give a real number). For $m < 24$, the square root is undefined, and there are no real roots of the quadratic $F(x)$; hence $F(x) < 0$ is never the case. □

COROLLARY 1. *For fixed m large, $K_{m,x}$ is not a 2-competition graph for*

$$x \in \left(\frac{\sqrt{1+8m}}{2} + \frac{3}{2}, m - \frac{\sqrt{1+8m}}{2} - 1 \right).$$

Proof. Consider

$$g(m) = [m^2 + 4m + 1 - 2m\sqrt{1+8m}]^{1/2} = m[1 + 4/m + 1/m^2 - \sqrt{4/m^2 + 32/m}]^{1/2}.$$

Using the binomial theorem and the notation $o(1)$ for terms that go to zero as m goes to ∞, we find that

$$\begin{aligned} g(m) &= m\left\{ 1 + \tfrac{1}{2}[4/m + 1/m^2 - \sqrt{4/m^2 + 32/m}] + \frac{\frac{1}{2}(-\frac{1}{2})}{2}[4/m + 1/m^2 \right. \\ &\qquad \left. - \sqrt{4/m^2 + 32/m}]^2 \right\} + o(1) \\ &= m + \tfrac{1}{2}[4 + 1/m - \sqrt{4 + 32m}] - \tfrac{1}{8}[4/\sqrt{m} + 1/m^{3/2} - \sqrt{4/m + 32}]^2 + o(1) \\ &\sim m - \sqrt{1+8m} - 2. \end{aligned}$$

Thus

$$\frac{m+1}{2} - \frac{g(m)}{2} \sim \frac{m}{2} + \frac{1}{2} - \left[\frac{m}{2} - \frac{\sqrt{1+8m}}{2} - 1 \right] = \frac{\sqrt{1+8m}}{2} + \frac{3}{2},$$

and

$$\frac{m+1}{2} + \frac{g(m)}{2} \sim \frac{m}{2} + \frac{1}{2} + \left[\frac{m}{2} - \frac{\sqrt{1+8m}}{2} - 1 \right] = m - \frac{\sqrt{1+8m}}{2} - \frac{1}{2}. \qquad \square$$

COROLLARY 2. *For fixed m large, $K_{m,x}$ is not a 2-competition graph for*

$$x \in (2 + m + \sqrt{1 + 2(m+1)}, \tfrac{1}{2}(m-2)^2).$$

Proof. By symmetry, Corollary 1 holds with m and x reversed. Thus, for fixed x large, $K_{m,x}$ is not a 2-competition graph for

$$m \in \left(\frac{\sqrt{1+8x}}{2} + \frac{3}{2}, x - \frac{\sqrt{1+8x}}{2} - \frac{1}{2} \right).$$

Since $\sqrt{1+8x} < \sqrt{8x} + 1$, we have that $K_{m,x}$ is not a 2-competition graph for fixed x large and

$$m \in \left(\frac{\sqrt{8x}+1}{2} + \frac{3}{2}, x - \left(\frac{\sqrt{8x}+1}{2} \right) - \frac{1}{2} \right),$$

i.e., for

$$m \in (\sqrt{2x} + 2, x - \sqrt{2x} - 1). \tag{8}$$

Note that

$$\begin{aligned} m > \sqrt{2x} + 2 &\leftrightarrow m - 2 > \sqrt{2x} \\ &\leftrightarrow x < \tfrac{1}{2}(m-2)^2. \end{aligned} \tag{9}$$

(The second equivalence follows, since $m - 2 > 0$.)

Let $z = \sqrt{x}$ (using the positive square root). Then, using the quadratic formula and the fact that $z > 0$, we have that

$$
\begin{aligned}
m < x - \sqrt{2x} - 1 &\leftrightarrow m < z^2 - \sqrt{2}z - 1 \\
&\leftrightarrow z^2 - \sqrt{2}z - (m+1) > 0 \\
&\leftrightarrow z > \tfrac{1}{2}[\sqrt{2} + \sqrt{2 + 4(m+1)}] \\
&\leftrightarrow x > 2 + m + \sqrt{1 + 2(m+1)}.
\end{aligned} \tag{10}
$$

Since $K_{m,x}$ is not a 2-competition graph for m in the interval given in (8), it follows from (9) and (10) that $K_{m,x}$ is not a 2-competition graph for

$$x \in (2 + m + \sqrt{1 + 2(m+1)}, \tfrac{1}{2}(m-2)^2). \qquad \square$$

Independently, Jacobson [3] obtained results that can be stated as follows.

THEOREM 6 (see [3]). *For fixed m large,*

(a) *$K_{m,x}$ is not a 2-competition graph if $x \in [m, (2+\sqrt{3})m)$;*

(b) *$K_{m,x}$ is a 2-competition graph if $x \in [16m^2, +\infty)$.*

Proof. (a) Jacobson proves that $K_{m,x}$ is not a 2-competition graph for sufficiently large x, $c > 2 - \sqrt{3}$, and $m = cx$, i.e., for sufficiently large m and $x < m/(2-\sqrt{3}) = (2+\sqrt{3})m$.

(b) Let $\alpha(t)$ be the smallest prime power that is at least as large as t. Since $2^r < t \leqq 2^{r+1}$ for some r, $\alpha(t) \leqq 2t$. Jacobson proves that $\theta_e^p \leqq mp(\alpha(\sqrt{x}))$ whenever $\alpha(\sqrt{x}) \geqq pm/(p-1)$. We show that if $x \geqq 16m^2$, then $2m(\alpha(\sqrt{x})) \leqq m + x$, which by Theorem 1 shows that $K_{m,x}$ is a 2-competition graph. (Note that $\alpha(\sqrt{x}) \geqq \alpha(4m) \geqq 4m > 2m$; so Jacobson's result applies.) If $x \geqq 16m^2$, we have

$$2m\alpha(\sqrt{x}) \leqq 2m(2\sqrt{x}) \leqq x < m + x. \qquad \square$$

Combining part (a) with Corollary 2 gives us that $K_{m,x}$ is not a 2-competition graph for $x \in [m, \frac{1}{2}(m-2)^2)$ when m is a fixed large number. We are not sure for what values of $x \in [\frac{1}{2}(m-2)^2, 16m^2]$ the graph $K_{m,x}$ is a 2-competition graph. Using better bounds on α (for example, those in Jacobson's paper), the constant 16 in $16m^2$ can be improved somewhat (to some value greater than or equal to 4).

3. $\mathbf{K_{2,x}}$. In this section, we study the values of x for which $K_{2,x}$ is a 2-competition graph. Suppose that $K_{2,x}$ has one independent set $\{a, b\}$ and a second independent set $B = \{a_1, \cdots, a_x\}$, and suppose that $S_1, \cdots, S_r$ is a 2-ECC for $K_{2,x}$. Let r_a be the number of sets S_j that contain a and suppose similarly for r_b. Let s_a be the largest size of a set $B \cap S_j$ for a set S_j containing a and suppose similarly for s_b. We start with a simple lemma.

LEMMA 7. *Suppose that $S_1, \cdots, S_r$ is a* 2-ECC *for $K_{2,x}$. Then*

(a) $r \geqq r_a + r_b - 1$;

(b) $r_a \geqq s_a + 1$, $r_b \geqq s_b + 1$; *and*

(c) *If $s_a = 1$, then $r_a \geqq 2x$; if $s_b = 1$, then $r_b \geqq 2x$.*

Proof. (a) The elements a and b cannot be in more than one set together.

(b) Start with a set S_j containing a such that $|B \cap S_j| = s_a$. Each element of $B \cap S_j$ appears in another set S_k, $k \neq j$, containing a, and no two elements of $B \cap S_j$ can appear together in more than one set of the 2-ECC. The same is true for b.

(c) Each element of B appears twice in a set containing a and twice in a set containing b. $\square$

Continuing with the above notation, suppose that $K_{2,x}$ is a 2-competition graph and that $r = 2 + x$. Let S_i be any set containing a and b, if there is such a set (there can be at most one), and let it be any set containing a otherwise. Let $N = |B - S_i|$.

LEMMA 8. *If* $S_1, \cdots, S_{2+x}$ *is a* 2-ECC *for* $K_{2,x}$, $1 \leqq x < 15$, *and* N *is defined as above, then*

(a) $N \leqq (3 + 3x)/(15 - x)$,

(b) $x \leqq 2N + 1$, *and*

(c) $x \geqq 7$ *or* $x \leqq 3$.

Proof. (a) We can assume that each set S_j contains either a or b, since otherwise we may replace S_j by $\varnothing$ and still have a 2-ECC. Every element a_k of $B - S_i$ appears in at least two sets $S_j, j \neq i$, with a, and in at least two sets $S_j, j \neq i$, with b. If a_k is in more than two sets containing a, or more than two sets containing b, it can be deleted from one of these sets without changing the fact that we have a 2-ECC. Thus, by iterating the argument, it follows that we can assume that a_k appears in exactly two of each kind of set. Moreover, since a and b appear together at most in S_i, these four sets containing a and a_k and b and a_k have distinct subscripts. Thus every element a_k of $B - S_i$ appears in exactly four sets S_j.

Let $T_j = S_j \cap (B - S_i)$. Suppose that every $T_j, j \neq i$, is empty. Then, if $B - S_i$ is not empty, there is a vertex in B that is in none of the sets in the 2-ECC. This vertex is an isolated vertex of $K_{2,x}$, which is a contradiction. Thus $B - S_i$ must be empty, N must be 0, and so (a) follows trivially. Thus assume that some $T_j, j \neq i$, is nonempty, and therefore $B - S_i \neq \varnothing$ and $N > 0$. Without loss of generality, relabel the sets so that T_j is nonempty if and only if $j \leqq q$. Thus, since every element a_k of $B - S_i$ is in exactly four sets T_j, we have that

$$\sum_{j=1}^{q} |T_j| = 4N. \tag{11}$$

Moreover, since $T_i = \varnothing$, we have that $q < 2 + x$.

Now every pair of elements in $B - S_i$ appears in at most one T_j. Hence

$$\sum_{j=1}^{q} \binom{|T_j|}{2} \leqq \binom{N}{2}.$$

Thus

$$\sum_{j=1}^{q} |T_j|^2 - \sum_{j=1}^{q} |T_j| \leqq N(N-1).$$

Using (11), we have that

$$\sum_{j=1}^{q} |T_j|^2 \leqq N^2 + 3N. \tag{12}$$

By the Cauchy–Schwartz inequality,

$$\left[\sum_{j=1}^{q} |T_j|\right]^2 \leqq q \sum_{j=1}^{q} |T_j|^2,$$

so (since $q \geqq 1$) (12) implies that

$$\frac{1}{q}\left[\sum_{j=1}^{q} |T_j|\right]^2 \leqq N^2 + 3N.$$

Using (11), the observation that $q < 2 + x$, and the facts that $N > 0$ and $x < 15$, we have that

$$\frac{1}{q}(4N)^2 \leqq N^2 + 3N,$$

$$\frac{1}{q} \leqq \frac{1}{16} + \frac{3}{16N},$$

$$\frac{3}{16N} \geqq \frac{1}{1+x} - \frac{1}{16} = \frac{15 - x}{16(1+x)},$$

$$N \leqq \frac{3 + 3x}{15 - x}.$$

(b) Note that $|B \cap S_i| = x - N$. Now every element of $B \cap S_i$ appears in another set S_j together with a and in another set S_j together with b. Since a and b do not appear together in any sets other than S_i and since two elements of B can appear together in at most one set, it follows that all of these additional sets S_j have distinct subscripts. Hence, counting S_i, the 2-ECC has at least $2(x - N) + 1$ sets. Thus $2x - 2N + 1 \leqq 2 + x$ or $x \leqq 2N + 1$.

(c) By parts (a) and (b),

$$\frac{3 + 3x}{15 - x} \geqq \frac{x - 1}{2},$$

so $(x - 7)(x - 3) \geqq 0$. □

THEOREM 9. *$K_{2,x}$ is a 2-competition graph if and only if $x = 1$ or $x \geqq 9$.*

Proof. It is useful to consider three separate cases: (a) $x = 1$, (b) $2 \leqq x \leqq 8$, and (c) $x \geqq 9$.

(a) $x = 1$. Then it is trivial to show by Theorem 1 that $K_{2,x}$ is a 2-competition graph.

(b) $2 \leqq x \leqq 8$. Let us use the notation defined before Lemmas 7 and 8, taking $r = 2 + x$. If $s_a = 1$, then, since r_b must be at least 2, parts (a) and (c) of Lemma 7 imply that

$$2 + x = r \geqq r_a + r_b - 1 \geqq 2x + 2 - 1,$$

which is a contradiction. Thus we may assume that $s_a \geqq 2$ and, similarly, that $s_b \geqq 2$. Hence, by Lemma 7(b), $r_a \geqq 3$ and $r_b \geqq 3$.

By Lemma 8(c), $x \leqq 3$ or $x \geqq 7$. If $x = 2$, then by Lemma 7(a),

$$4 = 2 + x = r \geqq 3 + 3 - 1 = 5,$$

which is a contradiction.

Next, suppose that $x = 3$. Then, if $r_a \geqq 4$ or $r_b \geqq 4$, we have that

$$5 = 2 + x \geqq 4 + 3 - 1 = 6,$$

which is a contradiction. Thus $r_a = r_b = 3$ and, by Lemma 7(b), $s_a = s_b = 2$. Thus the sets containing a must, in their intersections with B, be the sets $\{a_1, a_2\}$, $\{a_2, a_3\}$, and $\{a_1, a_3\}$. The same is true for the sets containing b in their intersections with B. However, now two elements of B appear together in more than one set, which is a contradiction.

We now consider the case where $x = 7$. By Lemma 8(a), $N \leqq \frac{24}{8} = 3$. If $N \leqq 2$, then, by Lemma 8(b), $x \leqq 5$, which is a contradiction. Thus we have that $N = 3$. Using

the notation of the proof of Lemma 8(a), we have $q < 2 + x$ sets T_j, which are subsets of a 3-element set $B - S_i$ and whose cardinalities total $4N = 12$. Moreover, any two of these sets have at most one element in common, since two elements of B appear in at most one set in common. Thus, if some T_j is all of $B - S_i$, all of the other T_j must be 1-element sets. It follows by (11) that $q \geqq 10$, which contradicts $q < 2 + x$. If all T_j are 1-element sets, then $q = 12 \geqq 2 + x$, again a contradiction. Even if some T_j's have two elements, at most three of these sets can have two elements, and then we need six more sets to get a total sum of cardinalities of 12. Hence $q \geqq 9 \geqq 2 + x$, and again there is a contradiction.

Finally, consider $x = 8$. By Lemma 8(a), $N \leqq \frac{27}{7}$, so $N \leqq 3$. By Lemma 8(b), however, $x \leqq 2N + 1 \leqq 7$, and we have a contradiction.

(c) $x \geqq 9$. We construct a 2-ECC E^x for $K_{2,x}$ recursively. E^x will consist of the $2 + x$ sets R^x, $K_1^x, \cdots, K_p^x$, $L_1^x, \cdots, L_q^x$, where

$$p = \begin{cases} x/2+1 & \text{if } x \text{ is even,} \\ (x+1)/2 & \text{if } x \text{ is odd;} \end{cases}$$

$$q = \begin{cases} x/2 & \text{if } x \text{ is even,} \\ (x+1)/2 & \text{if } x \text{ is odd.} \end{cases}$$

The set R^x will contain a, b, and some elements of B; the sets K_i^x will contain a and some elements of B; and the sets L_i^x will contain b and some elements of B. For $x = 9$, the sets are as follows:

$$R^9 = \{a, b, a_1, a_3, a_5, a_7, a_9\},$$

$$K_1^9 = \{a, a_1, a_2\}, \qquad L_1^9 = \{b, a_2, a_3, a_6\},$$

$$K_2^9 = \{a, a_3, a_4\}, \qquad L_2^9 = \{b, a_2, a_4, a_7\},$$

$$K_3^9 = \{a, a_4, a_5, a_6\}, \qquad L_3^9 = \{b, a_1, a_4, a_8\},$$

$$K_4^9 = \{a, a_2, a_8, a_9\}, \qquad L_4^9 = \{b, a_5, a_8\},$$

$$K_5^9 = \{a, a_6, a_7, a_8\}, \qquad L_5^9 = \{b, a_6, a_9\}.$$

That these sets form a 2-ECC for $K_{2,9}$ is easy to verify.

We now extend this definition recursively. If x is even, $x \geqq 10$, let

$$K_{x/2-4}^x = K_{x/2-4}^{x-1} \cup \{a_x\}, \qquad L_{x/2-1}^x = L_{x/2-1}^{x-1} \cup \{a_x\},$$

$$L_{x/2}^x = L_{x/2}^{x-1} \cup \{a_x\}, \qquad K_{x/2+1}^x = \{a, a_x\},$$

and, otherwise, let $R^x = R^{x-1}$, $K_i^x = K_i^{x-1}$, $L_i^x = L_i^{x-1}$. If x is odd, $x \geqq 11$, let

$$R^x = R^{x-1} \cup \{a_x\}, \quad K_{(x+1)/2}^x = K_{(x+1)/2}^{x-1} \cup \{a_x\}, \quad L_{(x+1)/2}^x = \{b, a_x\},$$

and, otherwise, let $K_i^x = K_i^{x-1}$, $L_i^x = L_i^{x-1}$.

Observe the following: (i) If $a_y \in R^x$, K_i^x, or L_i^x, then $y \leqq x$; (ii) If $a_y \in R^x$, then y is odd; and (iii) If $a_x \in K_i^x$ for x odd, $x \geqq 11$, then $i = (x + 1)/2$. Observation (iii) follows, since, by construction, if $i \neq (x + 1)/2$, $K_i^x = K_i^{x-1}$, and, by (i), a_x does not belong to K_i^{x-1}.

To see that we have defined a 2-ECC when $x > 9$, let us first observe that, for all $y \leqq x$, a and a_y appear in common in at least two of the sets, and b and a_y appear in common in at least two of the sets. This is because, if x is even, a and a_x appear in common in $K_{x/2-4}^x$ and $K_{x/2+1}^x$, and b and a_x appear in common in $L_{x/2-1}^x$ and

$L^x_{x/2}$; if x is odd, then a and a_x appear in common in R^x and $K^x_{(x+1)/2}$, and b and a_x appear in common in R^x and $L^x_{(x+1)/2}$. The result follows because K^z_i is a subset of K^{z+1}_i, L^z_i is a subset of L^{z+1}_i, and R^z is a subset of R^{z+1}.

Note also that a and b appear in common only in R^x. Thus it suffices to show that, if $y < z \leqq x$, then a_y and a_z appear in common in at most one set of E^x. We prove this by induction on x. It is true for $x = 9$. Assume that it is true for $x' < x$. Suppose that $y < z$. If $z < x$, then it is true for x, because, in going from E^{x-1} to E^x, neither a_y nor a_z is added to any set, and the inductive hypothesis can be applied. Thus it suffices to show this for $z = x$.

We first assume that x is odd, $x \geqq 11$. Note that a_x appears only in R^x, $K^x_{(x+1)/2}$, and $L^x_{(x+1)/2}$, and a_y is not in the last of these sets. Also, if a_y is in R^x, then, by observation (ii), y is odd. However, since x is odd and $x - 1$ is even and greater than or equal to 10,

$$K^x_{(x+1)/2} = K^{x-1}_{(x+1)/2} \cup \{a_x\} = K^{x-1}_{(x-1)/2+1} \cup \{a_x\} = \{a, a_{x-1}, a_x\}.$$

Since y is odd and $y < x$, we have that $y \neq x - 1$.

Next, suppose that x is even, $x \geqq 10$. Here a_x appears in only four sets. The case where $x = 10$ is a special case. In this case,

$$L^x_{x/2-1} = L^{x-1}_{x/2-1} \cup \{a_x\} = L^9_4 \cup \{a_{10}\} = \{b, a_5, a_8, a_{10}\},$$

$$L^x_{x/2} = L^{x-1}_{x/2} \cup \{a_x\} = L^9_5 \cup \{a_{10}\} = \{b, a_6, a_9, a_{10}\},$$

$$K^x_{x/2-4} = K^{x-1}_{x/2-4} \cup \{a_x\} = K^9_1 \cup \{a_{10}\} = \{a, a_1, a_2, a_{10}\},$$

$$K^x_{x/2+1} = \{a, a_x\} = \{a, a_{10}\}.$$

Thus, clearly, if $y < 10$, a_y appears in at most one of these sets.

The case where $x = 12$ is also a special case. In this case,

$$L^x_{x/2-1} = L^{11}_5 \cup \{a_{12}\} = L^{10}_5 \cup \{a_{12}\} = \{b, a_6, a_9, a_{10}, a_{12}\},$$

$$L^x_{x/2} = L^{11}_6 \cup \{a_{12}\} = \{b, a_{11}, a_{12}\},$$

$$K^x_{x/2-4} = K^{11}_2 \cup \{a_{12}\} = K^{10}_2 \cup \{a_{12}\} = K^9_2 \cup \{a_{12}\} = \{a, a_3, a_4, a_{12}\},$$

$$K^x_{x/2+1} = \{a, a_{12}\}.$$

Thus, if $y < 12$, a_y appears in at most one of these sets.

Finally, suppose that x is even and $x \geqq 14$. Then

$$\begin{aligned} L^x_{x/2-1} &= L^{x-1}_{x/2-1} \cup \{a_x\} \\ &= L^{x-2}_{(x-2)/2} \cup \{a_x\} \\ &= L^{x-3}_{(x-2)/2} \cup \{a_{x-2}\} \cup \{a_x\} \\ &= \{b, a_{x-3}, a_{x-2}, a_x\}, \end{aligned}$$

which holds, since $x - 3 \geqq 11$. Also,

$$\begin{aligned} L^x_{x/2} &= L^{x-1}_{x/2} \cup \{a_x\} \\ &= \{b, a_{x-1}, a_x\}, \\ K^x_{x/2-4} &= K^{x-1}_{x/2-4} \cup \{a_x\} \\ &= K^{x-2}_{(x-2)/2-3} \cup \{a_x\} \\ &= K^{x-3}_{(x-2)/2-3} \cup \{a_x\}, \end{aligned}$$

and

$$K^x_{x/2+1} = \{a, a_x\}.$$

To prove that a_y and a_x are not in common in more than one of these four sets, it therefore suffices to show that $a_y \in K^{x-3}_{(x-2)/2-3}$ implies that $y < x - 3$. By observation (i), however, $a_y \in K^{x-3}_{(x-2)/2-3}$ implies that $y \leqq x - 3$. If $y = x - 3$, then, by observation (iii), since $x - 3 \geqq 11$, we must have that

$$\frac{x-2}{2} - 3 = \frac{(x-3)+1}{2},$$

which is false. □

4. $K_{3,x}$. In this section, we consider the case where $m = 3$. That $K_{3,1}$ is a 2-competition graph follows by a straightforward construction of a 2-ECC. It also follows from the result of [5] that every tree is a 2-competition graph. That $K_{3,2}$ is not a 2-competition graph follows from Theorem 9.

THEOREM 10. *$K_{3,3}$ is not a 2-competition graph.*

Proof. Let $K_{3,3}$ have one independent set $\{a, b, c\}$ and a second independent set $\{x, y, z\}$. If $K_{3,3}$ is a 2-competition graph, then, by the corollary to Theorem 1, there is a 2-ECC $S_1, \cdots, S_6$. We first show that each vertex of $K_{3,3}$ is contained in exactly three of the sets S_j. Now a and x are in two sets together. However, y can be in at most one of these sets, since x and y are nonadjacent. Thus a and y must be in a third set. Hence a is in at least three sets. Similarly, each vertex must be contained in at least three S_j's. Suppose that a vertex, say a, is contained in more than three S_j's, say S_1, S_2, S_3, S_4. Since b is in at least three sets, b is in one of S_1, S_2, S_3, S_4, and it cannot be in more than one of these sets, since a and b are in at most one S_j together. Thus b is in S_5 and S_6. Similarly, c is in S_5 and S_6. This, however, is impossible.

Let us suppose that a is contained in S_1, S_2, S_3 only. If b is in none of these sets, then b is in all three of S_4, S_5, S_6. However, c must either be in at least two of S_1, S_2, S_3 or in at least two of S_4, S_5, S_6. In either case, there is a contradiction, since either a and c or b and c are in two sets together. Thus we may assume that b is in one of these sets, say S_1. Then b cannot be in S_2 or S_3. Since b is in three sets, we may assume that it is also in S_4 and S_5. Similarly, c is in one of S_1, S_2, S_3 and two of S_4, S_5, S_6. Since b is in two of the latter, b and c will overlap in one of the latter, and hence c cannot be in S_1. Thus, without loss of generality, we have c in S_2, S_4, S_6. Then x must be in two sets with a, two with b, and two with c, and the only possibility is for x to be in S_1, S_2, S_4. The same argument, however, puts y in S_1, S_2, S_4. Then x and y are in two sets together, which is a contradiction. □

In the following lemma, we use the notation u defined in the remark after Lemma 3.

LEMMA 11. *If $S_1, \cdots, S_t$ is a 2-ECC for $K_{m,x}$ and $u \geqq m - 1$, then*

$$t \geqq mu - \frac{m(m-1)}{2}.$$

Proof. Note that element a_1 of A must be in at least u sets of the 2-ECC; say it belongs to $S_{11}, S_{12}, \cdots, S_{1u}$. Element a_2 must be in at least u sets of the 2-ECC, and, since a_1 and a_2 can be in at most one set together, a_2 is in at least $u - 1$ sets not included in the sets S_{1j}. Call these sets $S_{21}, S_{22}, \cdots, S_{2(u-1)}$. Similarly, a_3 is in at most one of the S_{1j} and at most one of the S_{2j}, so in at least $u - 2$ other sets. Call these $S_{31}, S_{32}, \cdots,$

$S_{3(u-2)}$. By continuing the argument, (since $u \geqq m-1$) we find that

$$\begin{aligned} t &\geqq u + (u-1) + (u-2) + \cdots + (u-m+1) \\ &= mu - [1 + 2 + \cdots + (m-1)] \\ &= mu - \frac{m(m-1)}{2}. \end{aligned}$$

□

THEOREM 12. *$K_{3,x}$ is not a 2-competition graph for $x = 4, 5, 7, 8, 11$.*

Proof. In addition to Lemma 11, the proof will use the following facts:

(i) $u \geqq (1 + \sqrt{1 + 8x})/2$,

(ii) $t \geqq u^2 m/(u + m - 1) = f_m(u)$,

(iii) For fixed m, $f_m(u)$ is increasing in u (since $u \geqq 1$),

(iv) $m + x < t$ implies that $K_{m,x}$ is not a 2-competition graph.

Fact (i) is noted in the remark after Lemma 3, fact (ii) in the first remark before Theorem 5. Fact (iii) is easy to check by taking the derivative. Fact (iv) follows from Theorem 1.

Let $x = 4$. By (i),

$$u \geqq \frac{1+\sqrt{33}}{2} > 3,$$

and therefore $u \geqq 4$. Then, by (ii) and (iii), however,

$$t \geqq f_3(u) \geqq f_3(4) = 48/6 = 8 > m + x.$$

By (iv), $K_{3,4}$ is not a 2-competition graph.

Next, let $x = 5$. By (i),

$$u \geqq \frac{1+\sqrt{41}}{2} > 3,$$

so $u \geqq 4$. By Lemma 11, $t \geqq 12 - 3 = 9 > m + x$. By (iv), $K_{3,5}$ is not a 2-competition graph.

Suppose that $x = 7$. By (i),

$$u \geqq \frac{1+\sqrt{57}}{2} > 4,$$

so $u \geqq 5$. Then

$$t \geqq f_3(u) \geqq f_3(5) = \tfrac{75}{7},$$

so $t \geqq 11 > m + x$.

If $x = 8$, then, by (i),

$$u \geqq \frac{1+\sqrt{65}}{2} > 4,$$

so $u \geqq 5$. By Lemma 11, $t \geqq 15 - 3 = 12 > m + x$.

If $x = 11$, then, by (i),

$$u \geqq \frac{1+\sqrt{89}}{2} > 5.$$

Thus $u \geqq 6$. By Lemma 11, $t \geqq 18 - 3 = 15 > m + x$. □

Theorem 12 gives some values of x for which $K_{3,x}$ is known not to be a 2-competition graph. The next result gives values of x for which it is known to be such a graph.

THEOREM 13 (see [3]). *$K_{3,x}$ is a 2-competition graph for $x \geqq 38$.*

We do not know if $K_{3,37}$ is a 2-competition graph or if there is any $x \in (1, 38)$ such that $K_{3,x}$ is a 2-competition graph. The smallest value of x for which we do not know whether $K_{3,x}$ is a 2-competition graph is $x = 6$.

5. $K_{x,x}$. We turn now to the case where $m = x$. Theorems 9 and 10 already show that $K_{x,x}$ is not a 2-competition graph if $x = 2$ or 3. The next theorem follows by a simple argument, and therefore we include it here although it also follows from a stronger (and more difficult) result of [3], which we state below. To state the next theorem, we first need a lemma. In this lemma, we use the notation $\lceil \alpha \rceil$ to denote the least integer greater than or equal to α.

LEMMA 14. *Every positive integer x can be expressed uniquely in the form*

$$x = \binom{w-1}{2} + q, \qquad 0 < q \leqq w-1, \tag{13}$$

where

$$w = \left\lceil \frac{1+\sqrt{1+8x}}{2} \right\rceil. \tag{14}$$

Proof. Let $h(s) = (s(s-1))/2$ and let w be the smallest positive integer so that $h(w) \geqq x$. Hence, if s is such that $h(s) = x$, it follows because h is increasing for $s \geqq 1$ that $w = \lceil s \rceil$. Since

$$\binom{w-1}{2} < x \leqq \binom{w}{2},$$

it follows that x can be expressed in the form (13). By the quadratic formula, it follows that $s(s-1)/2 = x$ and $s > 0$ imply that

$$s = \frac{1+\sqrt{1+8x}}{2}.$$

This gives us (14). □

THEOREM 15. *If w and q are defined as in Lemma 14, then $K_{x,x}$ is not a 2-competition graph if $q < w/2$.*

Proof. The proof uses (i)–(iv) of the proof of Theorem 12. By (i), $u \geqq w$. By (ii) and (iii), $t \geqq f_m(u) \geqq f_m(w) = f_x(w)$. Then

$$f_x(w) > m + x = 2x \leftrightarrow \frac{w^2 x}{w+x-1} > 2x$$
$$\leftrightarrow w^2 > 2w + 2x - 2.$$

Hence, by (13),

$$f_x(w) > m + x \leftrightarrow w^2 > 2w + [(w-1)(w-2) + 2q] - 2$$
$$\leftrightarrow w > 2q$$
$$\leftrightarrow q < w/2.$$

The theorem follows by (iv). □

Remark. By using this theorem, we can show, for example, that $K_{x,x}$ is not a 2-competition graph for $x = 2, 4, 7, 8, 11, 12, 16, 17, 18, 22, 23, 24, 29, 30, 31, 32, 37, 38, 39, 40$. From Theorem 10, we know that this conclusion also holds for $x = 3$.

THEOREM 16 (see [3]). *$K_{x,x}$ is not a 2-competition graph for $x \geqq 4$.*

6. Closing remarks. The results in this paper leave some natural questions unresolved. For instance, the proof of Theorem 9 shows that if $K_{2,x}$ is a 2-competition graph and $x > 1$, then $K_{2,x+1}$ is a 2-competition graph. We have not been able to settle whether, for $x \geqq m > 2$, $K_{m,x}$ being a 2-competition graph implies that $K_{m,x+1}$ is a 2-competition graph. While we have determined exactly for what values of x $K_{2,x}$ is a 2-competition graph, the problem for $K_{3,x}$ remains open. In particular, small values of x such as $x = 6$ remain unresolved, as does the question of whether $K_{3,x}$ can be a 2-competition graph for any $1 < x < 38$. For the case of $K_{4,x}$, which we have not discussed in this paper, [3] shows that $K_{4,x}$ is a 2-competition graph for $x \geqq 124$ and that $K_{4,x}$ is not a 2-competition graph for $x = 4, \cdots, 10$. However, nothing else is known here.

Acknowledgments. The authors thank Denise Sakai and Chi Wang for their helpful comments. The authors also thank Jeff Kahn for suggesting use of the Cauchy–Schwartz inequality in the proof of Lemma 4 and Rob Hochberg for suggesting use of the binomial theorem in the proof of Corollary 1 to Theorem 5.

REFERENCES

[1] J. A. BONDY AND U. S. R. MURTY, *Graph Theory with Applications*, American Elsevier, New York, 1976.

[2] R. D. DUTTON AND R. C. BRIGHAM, *A characterization of competition graphs*, Discrete Appl. Math., 6 (1983), pp. 315–317.

[3] M. JACOBSON, *On the p-edge clique covering number of complete bipartite graphs*, Dept. of Math., University of Louisville, Louisville, KY, 1989; SIAM J. Discrete Math., this issue, pp. 539–544.

[4] S. KIM, *Competition graphs and scientific laws for food webs and other systems*, Ph.D. thesis, Dept. of Math., Rutgers University, New Brunswick, NJ, 1988.

[5] S. KIM, T. A. MCKEE, F. R. MCMORRIS, AND F. S. ROBERTS, *p-competition graphs*, RUTCOR Research Report RRR 36-89, Rutgers Center for Operations Research, Rutgers University, New Brunswick, NJ, 1989.

[6] J. R. LUNDGREN, *Food webs, competition graphs, competition-common enemy graphs, and niche graphs*, in Applications of Combinatorics and Graph Theory in the Biological and Social Sciences, F. S. Roberts, ed., IMA Volumes in Mathematics and its Applications, Vol. 17, Springer-Verlag, New York, 1989, pp. 221–243.

[7] J. R. LUNDGREN AND J. S. MAYBEE, *A characterization of graphs of competition number m*, Discrete Appl. Math., 6 (1983), pp. 319–322.

[8] A. RAYCHAUDHURI AND F. S. ROBERTS, *Generalized competition graphs and their applications*, in Methods of Operations Research, 49, P. Brucker and P. Pauly, eds., Anton Hain, Konigstein, Germany, 1985, pp. 295–311.

[9] F. S. ROBERTS, *Applied Combinatorics*, Prentice-Hall, Englewood Cliffs, NJ, 1984.

[10] ———, *Applications of edge coverings by cliques*, Discrete Appl. Math., 10 (1985), pp. 93–109.

[11] F. S. ROBERTS AND J. E. STEIF, *A characterization of competition graphs of arbitrary digraphs*, Discrete Appl. Math., 6 (1983), pp. 323–326.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 539–544, November 1992
© 1992 Society for Industrial and Applied Mathematics
009

ON THE p-EDGE CLIQUE COVER NUMBER OF COMPLETE BIPARTITE GRAPHS*

MICHAEL S. JACOBSON†

Abstract. Given a digraph $D = (V, A)$ and a positive integer p, the *p-competition graph* of D, denoted $C_p(D)$, is defined to have vertex set V and for $x, y \in V$, $xy \in E(C_p(D))$ if and only if there are at least p distinct vertices $v_1, v_2, \cdots, v_p \in V(D)$ such that xv_i and $yv_i \in A(D)$ for $i = 1, 2, \cdots, p$. This paper furthers the study of complete bipartite graphs that are p-competition graphs. The primary technique used is the concept of the p-edge clique cover (p-ECC) number. General results are given for K_3-free graphs, as well as the result that $K_{n,n}$ is not a 2-competition graph for $n \geqq 4$.

Key words. graph, bipartite, clique cover number

AMS(MOS) subject classification. 05C

1. Introduction. Given a digraph $D = (V, A)$ and a positive integer p, the *p-competition graph* of D, denoted $C_p(D)$, is defined to have vertex set V and for $x, y \in V$, $xy \in E(C_p(D))$ if and only if there are at least p distinct vertices $v_1, v_2, \cdots, v_p \in V(D)$ such that xv_i and $yv_i \in A(D)$ for $i = 1, 2, \cdots, p$. The concept of p-competition graphs was introduced in [6], generalizing the idea of competition graphs [2], [5], [7]. The fundamental question is: For what graphs G does there exist a digraph D so that $G = C_p(D)$? That is, for a given integer p, characterize those graphs that are p-competition graphs. The study of this question for complete bipartite graphs was started by Isaak et al. in [4]. In this paper, we attempt to give additional insight into this problem and, essentially, with the results in [4], answer the question for $p = 2$.

It is easy to reduce the problem of determining if a graph is a p-competition graph to an alternate graph-theoretic problem. Let $\mathcal{F} = \{S_1, S_2, \cdots, S_r\}$ be a family, repetitions allowed, of subsets of the vertex set of G. We say that F is a *p-edge clique cover*, or a p-ECC, if, for every set of p distinct indices $i_1, i_2, \cdots, i_p$,

$$T = S_{i_1} \cap S_{i_2} \cap \cdots \cap S_{i_p}$$

is either empty or induces a complete subgraph of G, and, furthermore, the induced subgraphs of the sets of the form T compose an edge cover of G. Define the *p-edge clique cover number*, denoted $\theta_p(G)$, to be the smallest r for which there is a p-ECC containing r sets for G. We will use simply θ_p when no confusion will result. Kim et al. [6] proved that if G is a graph of order n, then G is a p-competition graph if and only if $\theta_p \leqq n$.

In the remainder of this paper, we study $\theta_p(G)$, the conclusion of which will enable us to deduce results pertaining to whether G is a p-competition graph. In particular, we will show that, for m and p fixed, $K_{m,n}$ is a p-competition graph for n sufficiently large. As a corollary to the proof, we get that $K_{2,n}$ is a 2-competition graph for $n \geqq 14$ (this is superceded by the results in [4], where it is shown that $K_{2,n}$ is not a 2-competition graph if and only if $2 \leqq n \leqq 9$.). Also, we are able to show that $K_{3,n}$ is a 2-competition graph for $n \geqq 38$. By using a counting argument, we show that, if $2 - \sqrt{3} < c \leqq 1$ and $m = cn$, then $K_{m,n}$ is not a 2-competition graph for sufficiently large n. In particular, we are able to answer the question completely for $K_{n,n}$ and to show that it is not a 2-competition graph for $n \geqq 4$. The case where $n = 2$ and 3 was previously considered in [6].

* Received by the editors February 7, 1990; accepted for publication (in revised form) June 11, 1991. This research was supported by Office of Naval Research grant N00014-85-0694.

† Department of Mathematics, University of Louisville, Louisville, Kentucky 40292.

2. General results. Although the principal results contained here are for complete bipartite graphs, some of the techniques also apply to a much broader class of graphs. To say that a graph is K_3-*free* means that K_3 is not a subgraph of the original graph. Of course, bipartite graphs and complete bipartite are K_3-free.

PROPOSITION 2.1. *Let* $\mathcal{F} = \{S_1, S_2, \cdots, S_\theta\}$ *be a* p-ECC *of* G. *If* G *is* K_3-*free, then for any vertex* $x \in V(G)$, x *must be contained in at least* r_x *elements of* $\mathcal{F}$, *where* r_x *is the smallest integer solution to* $\binom{r}{p} \geqq \deg x$.

Proof. Assume the hypotheses and let x be any vertex of G. Consider all the sets of $\mathcal{F}$ that contain the vertex x. Each element in the neighborhood of x must be in at least p elements of $\mathcal{F}$, which also contain x since $\mathcal{F}$ is a p-ECC. Furthermore, since G is K_3-free, each element in $N(x)$ that is in the intersection of p elements of $\mathcal{F}$, each also containing x, is the only such element in that intersection. Since the set of p intersections forms an edge cover, and since $\binom{r}{p}$ is the number of all p intersections of sets containing x, it follows that $\binom{r}{p} \geqq \deg x$.

COROLLARY 2.2. *If* G *is* K_3-*free, with* $\mathcal{F}$ *a* p-ECC, *then* x *is in at least* $(p! \deg x)^{1/p}$ *elements of* $\mathcal{F}$ *for each* $x \in V(G)$.

COROLLARY 2.3. *If* G *is* K_3-*free and* $\mathcal{F}$ *is a* 2-ECC, *then every vertex of* G *is in at least*

$$\left\lceil \frac{1+\sqrt{1+8 \deg x}}{2} \right\rceil$$

elements of $\mathcal{F}$.

Proof. From Proposition 2.1, we know that, for any vertex x, it must be in at least r sets, where r is the smallest integer that satisfies $r^2 - r \geqq 2 \deg x$. Using the quadratic formula, the result follows.

3. A lower bound for $\theta_2(K_{m,n})$. The remainder of the paper will focus on complete bipartite graphs. For convenience, when discussing $K_{m,n}$, we will assume bipartition X, Y with $|X| = m$ and $|Y| = n$. In addition, let $X = \{x_1, x_2, \cdots, x_m\}$ and $Y = \{y_1, y_2, \cdots, y_n\}$.

LEMMA 3.1. *If* $\mathcal{F} = \{S_1, S_2, \cdots, S_\theta\}$ *is a* 2-ECC *of* $K_{m,n}$, *then every vertex of* X *is in at least*

$$\left\lceil \frac{1+\sqrt{1+8n}}{2} \right\rceil$$

elements of $\mathcal{F}$, *and every element* Y *is in at least*

$$\left\lceil \frac{1+\sqrt{1+8m}}{2} \right\rceil$$

elements of $\mathcal{F}$.

Proof. This follows, since the degree of every vertex in X is n and since the degree of every vertex in Y is m.

THEOREM 3.2. *If* $m = cn$ *for some* $0 < c \leqq 1$, *then*

$$\theta_2(K_{m,n}) \geqq \left\lceil \frac{1+\sqrt{1+[(-c^2+4c-1)n^2+(1+c)n][\sqrt{1+8cn}-1]}}{2} \right\rceil.$$

Furthermore, for sufficiently large n *and* $c > 2 - \sqrt{3}$, $K_{m,n}$ *is not a* 2-*competition graph.*

Proof. Let $\mathcal{F} = \{S_1, S_2, \cdots, S_\theta\}$ be a 2-ECC of minimum order and let $X_i = S_i \cap X$ with $|X_i| = \alpha_i$ and $Y_i = S_i \cap Y$ with $|Y_i| = \beta_i$. Note that, since $\mathcal{F}$ has minimum order, no X_i nor Y_j can be empty.

Since no pair of vertices in X or Y can be in more than one set of $\mathcal{F}$, it follows that

$$\sum^{\theta}_{i=1}\binom{\alpha_i}{2}\leqq\binom{m}{2}, \qquad \sum^{\theta}_{i=1}\binom{\beta_i}{2}\leqq\binom{n}{2}.$$

(The summation indices will henceforth be omitted for convenience.) Also, since every edge must occur in at least two elements of $\mathcal{F}$, it follows that $\sum \alpha_i\beta_i \geqq 2mn$. This implies that

$$\sum \alpha_i^2-\alpha_i \leqq m^2-m=c^2n^2-cn,$$

$$\sum \beta_i^2-\beta_i \leqq n^2-n,$$

$$\sum \alpha_i\beta_i \geqq 2cn^2.$$

Consequently,

$$\sum(\alpha_i^2+\beta_i^2)-\sum(\alpha_i+\beta_i)\leqq(1+c^2)n^2-(1+c)n,$$

$$\sum(\alpha_i^2+\beta_i^2)-\sum(\alpha_i+\beta_i)\leqq(1+c^2)/(2c)\sum\alpha_i\beta_i-(1+c)n,$$

which implies that

$$\sum(\alpha_i^2-2\alpha_i\beta_i+\beta_i^2)+(2-(1+c^2)/(2c))\sum\alpha_i\beta_i\leqq\sum(\alpha_i+\beta_i)-(1+c)n,$$

$$\sum(\alpha_i-\beta_i)^2+(2-(1+c^2)/(2c))\sum\alpha_i\beta_i\leqq\sum(\alpha_i+\beta_i)-(1+c)n,$$

$$(2-(1+c^2)/(2c))\sum\alpha_i\beta_i\leqq\sum(\alpha_i+\beta_i)-(1+c)n.$$

This yields

$$\sum(\alpha_i+\beta_i)\geqq(-c^2+4c-1)n^2+(1+c)n. \tag{1}$$

Since $\sum(\alpha_i+\beta_i)$ is the sum of 2θ positive integers, it follows that for some i, either α_i or β_i is at least

$$\left\lceil\frac{(-c^2+4c-1)n^2+(1+c)n}{2\theta}\right\rceil. \tag{2}$$

(Note that the value in (2) is positive as long as $c>2-\sqrt{3}$.) By the implication of Lemma 3.1, we will assume that

$$\beta_i\geqq\left\lceil\frac{(-c^2+4c-1)n^2+(1+c)n)}{2\theta}\right\rceil$$

for some i. (The conclusion is stronger if we can find α_i that large.) Since Y_i has order β_i, and each element of Y_i must occur in at least

$$\left\lceil\frac{1+\sqrt{1+8cn}}{2}\right\rceil$$

elements of $\mathcal{F}$ (that is, each element is in

$$\left\lceil\frac{\sqrt{1+8cn}-1}{2}\right\rceil$$

additional sets, and no pair of vertices in Y_i can be duplicated), it follows that

$$\theta\geqq 1+\left\lceil\frac{(-c^2+4c-1)n^2+(1+c)n}{2\theta}\right\rceil\left\lceil\frac{\sqrt{1+8cn-1}}{2}\right\rceil. \tag{3}$$

The lower bound follows from (3).

It is an easy consequence to see from (3) that $\theta \geqq \varepsilon n^{5/4}$ for ε a function of c. Hence, for $c > 2 - \sqrt{3}$ and n sufficiently large,

$$\theta_2(K_{m,n}) \geqq \varepsilon n^{5/4} > m + n = (1+c)n.$$

Thus $K_{m,n}$ is not a 2-competition graph, for sufficiently large n.

As applications of this result, we have the following corollary.

COROLLARY 3.3. *If $n \geqq 6$ or $n = 4$, then $K_{n,n}$ is not a 2-competition graph.*

Proof. Suppose that $K_{n,n}$ is a 2-competition graph. This would imply that $\theta_2(K_{n,n}) \leqq 2n$. By (3) of Theorem 3.2, this implies that

$$2n \geqq \left\lceil \frac{2n^2+2n}{4n} \right\rceil \left\lceil \frac{\sqrt{1+8n}-1}{2} \right\rceil + 1,$$

which gives

$$2n \geqq \left\lceil \frac{n+1}{2} \right\rceil \left\lceil \frac{\sqrt{1+8n}-1}{2} \right\rceil + 1. \tag{4}$$

This inequality is false for $n = 4$ and $n \geqq 6$. Note, that for $n = 5$, (4) is an equality.

COROLLARY 3.4. *$K_{5,5}$ is not a 2-competition graph.*

Proof. For $K_{5,5}$ to be a 2-competition graph, it follows that there is a 2-ECC with at most ten subsets. Choose a 2-ECC with ten subsets and form X_i and Y_i as in Theorem 3.2. By Lemma 2.1, each element of X and Y is in at least four subsets of the 2-ECC. By (1) of Theorem 3.2, $\sum (\alpha_i + \beta_i) \geqq 60$. If X_i (or Y_j) contains four vertices, by (3) this would imply that $\theta_2 \geqq 1 + 4(3) = 13$. Hence X_i (and Y_j) must each contain exactly three elements. Since there are only ten different three-element subsets of X, however, some pair of them must contain at least two common vertices, and hence will not be a 2-ECC. Therefore $K_{5,5}$ is not a 2-competition graph.

Putting the results of [4] together with these corollaries, we get the following result.

THEOREM 3.5. *$K_{n,n}$ is not a 2-competition graph for $n \geqq 2$.*

4. A construction for p-ECCs. For convenience, we define for any real number t, $\alpha(t)$ to be the smallest integer greater than or equal to t, so that $\alpha(t)$ is a prime power. We will use the fact (cf. [1]) that, for every prime power q, there exists an affine geometry with q^2 points, where each line contains q points and each point is in $q + 1$ lines. In addition, the set of $q^2 + q$ lines can be partitioned into $q + 1$ classes of q parallel lines so that every point is in each parallel class exactly once. We also use the fact that every pair of points lies on precisely one line in the geometry.

It is known [3] that $\alpha(t) \leqq t + kt^{11/20+\varepsilon}$ for any $\varepsilon > 0$ and k a function of ε. In fact, between t and $t + kt^{11/20+\varepsilon}$, there is a prime under the above conditions.

THEOREM 4.1. *It holds that $\theta_p(K_{m,n}) \leqq mp(\alpha(\sqrt{n}))$, whenever $\alpha(\sqrt{n}) \geqq pm/(p-1)$.*

Proof. Let $s = \alpha(\sqrt{n})$. Consider the affine geometry with s^2 points. Associate with each vertex of Y a distinct point in the geometry. The points in the geometry with no vertex of Y associated with it can be ignored. Let $C_1, C_2, \cdots, C_{s+1}$ be a partition of the lines into parallel classes. Take $p - 1$ copies of each parallel class and order them in any manner, label them $C'_1, C'_2, \cdots, C'_{(p-1)(s+1)}$.

Construct a p-ECC $\mathcal{F}$ in the following manner. For each $x_i \in X$, take the union of x_i with the vertices in each of the lines in

$$C'_{p(i-1)+1}, C'_{p(i-1)+2}, \cdots, C'_{pi}$$

to be the sets of $\mathcal{F}$ containing x_i. This can be completed as long as $pm \leqq (p-1) \times (s+1)$. Each x_i is in sp sets and does not occur in any other set with any other element of X. This family contains mps sets.

It is easy to see that this forms a p-ECC. No pair of vertices from Y occur in more than $p-1$ sets, no pair of vertices in X occurs in any set together, and every pair of vertices $x_i y_j$ occurs in exactly p sets together. Thus $\mathcal{F}$ is a p-ECC, and it follows that

$$\theta_p(K_{m,n}) \leqq mp(\alpha(\sqrt{n}))$$

under the appropriate conditions.

COROLLARY 4.2. *For a fixed m and p, $\varepsilon > 0$, and n sufficiently large,*

$$\theta_p(K_{m,n}) \leqq mp(n^{1/2} + kn^{11/40+\varepsilon})$$

for some k dependent on ε. Furthermore, $K_{m,n}$ is a 2-competition graph.

Proof. The first part follows, since $\alpha(\sqrt{n}) \leqq n^{1/2} + kn^{11/40+\varepsilon}$; see [3]. This implies that $mp(\alpha(\sqrt{n}) \leqq m + n$ for sufficiently large n.

COROLLARY 4.3. *$K_{2,n}$ is a 2-competition graph for $n \geqq 14$.*

Proof. For $n = 14$, 15, and 16, by using the geometry containing 16 points and the construction of Theorem 4.1, we get a 2-ECC with 16 sets, which is at most $n+2$ in these cases. For $18 \leqq n \leqq 25$, using the geometry with 25 points, we get an appropriate 2-ECC with 20 sets. For $n = 17$, choose a particular line of the geometry and assign none of the vertices of Y to any of the points on that line. This can be accomplished since each line of the geometry contains only five points. Using the construction as described above, we arrive at a 2-ECC with 19 nonempty sets, the empty set can simply be discarded. For $26 \leqq n \leqq 49$ using the geometry with 49 points and the construction above, we get a 2-ECC with 28 sets, which is at most $n+2$. For $n \geqq 50$, Corollary 4.2 gives the result.

COROLLARY 4.4. *$K_{3,n}$ is a 2-competition graph for $n \geqq 38$.*

Proof. For $n \geqq 39$, the 2-ECC given by the construction on the geometries with 49, 81 and assured by Corollary 4.2 implies that $K_{3,n}$ is a 2-competition graph. For $K_{3,38}$, as in the previous result, by assigning no vertices of Y to the points of a particular line in the geometry on 49 points, a 2-ECC results that contains 41 sets. Thus $K_{3,38}$ is a 2-competition graph, and this completes the proof.

5. Conclusion. The material presented in this paper only lightly considers the general problem of determining when a graph is or is not a p-competition graph. Several obvious problems arise.

First, combining the results presented here with those given in [4], it is known that $K_{m,n}$ is not a 2-competition graph for $m \leqq n \leqq (m-2)^2/2$ and that it is a 2-competition graph when $n \geqq 16m^2$. What is the cutoff level? Does one exist? Is it possible that $K_{m,n}$ is a 2-competition graph, but that $K_{m,n+1}$ is not?

Second, what about the case where $p > 2$? Can the counting techniques of §§ 2 and 3 be extended to attain a general bound for θ_p? Finally, it is worth considering K_3-free graphs that are not complete bipartite graphs. It might also be worth studying complete multipartite graphs because of their nice structure.

REFERENCES

[1] P. DUMBOWSKI, *Finite Geometry*, Springer-Verlag, New York, 1968.

[2] R. D. DUTTON AND R. C. BRIGHAM, *A characterization of competition graphs*, Discrete Appl. Math., 6 (1983), pp. 315–317.

[3] D. R. HEATH-BROWN AND H. IWANIEC, *On the difference between consecutive primes*, Invent. Math., 55 (1979), pp. 49–69.

[4] G. ISAAK, S. KIM, T. A. MCKEE, F. R. MCMORRIS, AND F. S. ROBERTS, *2-competition graphs*, SIAM J. Discrete Math., this issue, pp. 524–538.

[5] S. KIM, *Competition graphs and scientific laws for food webs and other systems*, Ph.D. thesis, Dept. of Math., Rutgers University, New Brunswick, NJ, 1988.

[6] S. KIM, T. A. MCKEE, F. R. MCMORRIS, AND F. S. ROBERTS, *p-competition graphs*, Discrete Appl. Math., submitted.

[7] J. R. LUNDGREN, *Food webs, competition graphs, competition-common enemy graphs and niche graphs*, in Applications of Combinatorics and Graph Theory in Biological and Social Science, Vol. 17, F. S. Roberts, ed., IMA Volumes in Mathematics and Its Applications, Springer-Verlag, New York, 1989, pp. 221–243.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 545–557, November 1992
© 1992 Society for Industrial and Applied Mathematics
010

THE PROBABILISTIC COMMUNICATION COMPLEXITY OF SET INTERSECTION*

BALA KALYANASUNDARAM†‡ AND GEORG SCHNITGER†§

Abstract. It is shown that, for inputs of length n, the probabilistic (bounded error) communication complexity of *set intersection* is $\Theta(n)$. Since *set intersection* can be recognized nondeterministically by exchanging $O(\log n)$ bits, the result implies an exponential gap between nondeterminism and probabilism. This gap is the largest possible. The first general technique to analyze the probabilistic communication complexity of sparse languages is also described.

Key words. lower bound, nondeterminism, probabilistic communication complexity, set intersection

AMS(MOS) subject classifications. 05B20, 68Q30

1. Introduction. We first describe the model of communication complexity as introduced in [16]; a related model is given by [1].

Let $L \subset \{0, 1\}^{2n}$ be a language to be recognized by two agents P and Q. Agent P knows the first n bits, and Q knows the remaining bits of a given input string. P and Q have unlimited computational power and exchange messages to recognize L according to some protocol. The deterministic (nondeterministic) communication complexity of L is the minimum number of bits exchanged for the worst-case input, where the minimum is taken over all deterministic (nondeterministic) protocols recognizing L. This model is quite well understood for deterministic protocols [2], [6], [11], [12], [16] and also for nondeterministic protocols [8].

Bounded-error probabilistic protocols differ from deterministic protocols in allowing a probability distribution on the set of possible messages. An input is accepted if the probability of acceptance is at least $1 - \varepsilon$ for some fixed ε, $0 \leqq \varepsilon < \frac{1}{2}$, rejected if the probability of acceptance is at most ε, and all input pairs (x, y) must fall in one of these categories. The complexity of a protocol on input (x, y) is the average length of the messages exchanged between the two agents, where P knows x and Q knows y. The complexity of a protocol is defined as the maximum (over all inputs) of the average message length.

Halstenberg and Reischuk [7] show that, without altering the asymptotic complexity, we can restrict our attention to the worst-case message length of *regular* protocols. We say that a protocol is regular if it exchanges messages according to the *uniform* probability distribution. Let us now define the probabilistic communication complexity $C_\varepsilon(L)$ as the minimum (over all regular probabilistic protocols recognizing L with error probability ε) of the worst-case message length.

Probabilistic (bounded error) communication complexity is introduced by Yao [15]. Yao relates the complexity of probabilistic protocols to the *distributional* complexity, namely, the complexity of *almost* correct deterministic protocols. Using this technique,

* Received by the editors January 16, 1989; accepted for publication (in revised form) June 11, 1991. A preliminary version of this paper appeared in the Proceedings of the Second Annual Structure in Complexity Theory Conference, Ithaca, New York, 1987, pp. 41–49.

† Department of Computer Science, The Pennsylvania State University, University Park, Pennsylvania 16802.

‡ Current address, Department of Computer Science, University of Pittsburgh, Pittsburgh, Pennsylvania 15260.

§ This author's research was supported by National Science Foundation contract DCR-8407256 and by Office of Naval Research contract N0014-80-0517.

Yao develops methods to analyze the probabilistic communication complexity of dense languages. Examples of successfully analyzed languages include *multiplication modulo prime number* [15] and *inner product modulo* 2 [5].

A protocol is considered to be polynomial time if at most $(\log n)^c$ bits are exchanged for some constant c. Consequently, we can introduce the complexity classes P (deterministic polynomial time), NP (nondeterministic polynomial time), and BPP (probabilistic polynomial time) for communication complexity. Babai, Frankl, and Simon [3] introduce the notion of *rectangular reductions* to extend the notion of NP-completeness to communication complexity. One example of an "NP-complete" language is the language *set intersection*, which is defined as the set of all binary strings $(a_1, a_2, \cdots, a_n, b_1, b_2, \cdots, b_n)$ with $\sum_{i=1}^{n} a_i b_i \geqq 1$.

Chor and Goldreich's result [5] implies a tight $\Theta(n)$ lower bound if we are interested in computing the *exact size* of a set intersection, instead of only settling the *disjointness* problem. Babai, Frankl, and Simon derive an $\Omega(\sqrt{n})$ lower bound for the probabilistic communication complexity of *set intersection*, thus proving an exponential gap between nondeterminism and probabilism. On the other hand, they show that the distributional complexity of *set intersection*, when restricted to product distributions, is bounded by $O(\sqrt{n} \log n)$.

We can determine the probabilistic communication complexity of *set intersection* in the following theorem.

THEOREM. *For every* $\varepsilon < \frac{1}{2}$, $C_\varepsilon(\textit{set intersection}) = \Theta(n)$.

Thus we obtain the best possible gap between nondeterminism and probabilism. This result has been applied by Raz and Wigderson [13], who obtained exponential lower bounds for the size of (depth-constrained) monotone circuits.

Technically, the difficulty in analyzing *set intersection* is due to its sparsity: only 3^n of its 4^n inputs belong to the language. Previous methods [3], [5] proceed by testing the error performance of *all* possible protocols relative to a *single* product distribution defined on the set of inputs. Instead, our approach is to design a collection of subsets on which to perform "error accounting." Given the probabilistic protocol, we choose a subset from this collection at random, hoping that the given protocol is unprepared for this particular test set. Furthermore, our subsets are not product sets.

Recently, Razbarov [14] was able to simplify our analysis with an argument tailor-made for *set intersection*. However, our approach is more general and seems to be applicable to large classes of sparse languages.

The organization of this paper is as follows. First, in § 2 we give a proof outline and define our collection of test sets. We also (in Lemma 2.1) describe our general technique for sparse languages. The expected behavior of our collection of test sets is investigated in §§ 3 and 4. The proof then concludes in § 5. The results of § 5 are independent of the properties of *set intersection*.

A preliminary version of this paper appeared in [10].

2. Proof outline. Let $L \subset \{0, 1\}^{2n}$ be a language to be recognized by the agents P and Q. Following Yao [16], we associated the "truth matrix" $M(L)$ with L. The rows of $M(L)$ are labeled with the inputs for agent P, and its columns are labeled with inputs for agent Q. The entry (x, y) of $M(L)$ is defined to be 1, whenever (x, y) belongs to L. Otherwise, we assign the value 0.

Let P be a probabilistic bounded error protocol that recognizes L. Following [7], we can demand that P generates its messages according to the uniform distribution, and we can restrict our attention to the worst-case message length, since expected message length and worst-case message length are of the same order of growth. Therefore, if m is

the worst-case message length of P, we can interpret P as a collection $D_1, \cdots, D_{2^m}$ of equally likely deterministic protocols with worst-case message length m. Let ε $(0 < \varepsilon < \frac{1}{2})$ be the error probability of P. Then, for each input (x, y), all but $\varepsilon 2^m$ protocols D_i compute correctly.

We now focus on a particular deterministic protocol D_i. First, we demand that the last bit of each exchange of messages indicates acceptance or rejection. Therefore each fixed exchange of messages induces a submatrix of $M(L)$ and the deterministic protocol partitions $M(L)$ into "accepting," respectively, "rejecting," submatrices [16]. Let us define the error ratio ε_i of D_i as the proportion of all entries of $M(L)$ that are incorrectly covered by accepting or rejecting submatrices. We obtain that $\sum_{i=1}^{2^m} \varepsilon_i \leqq \varepsilon 2^m$.

This observation led Yao [15] to introduce the distributional complexity $D_\varepsilon(L)$. It is defined to be the minimum (over all deterministic protocols recognizing L with error ratio at most ε) of the worst-case message length. (Yao's original definition is relative to the average message length.) We obtain, analogous to [15], that $C_\varepsilon(L) \geqq D_{2\varepsilon}(L)$.

We extend Yao's technique by restricting our attention to certain special entries. We define those special entries using a *restriction matrix* A of the same dimension as $M(L)$. If an entry in A is 1, then the corresponding entry in $M(L)$ is one of the special entries in which we are interested.

For a restriction matrix A, let $0_A(L)$ (respectively, $1_A(L)$) be the number of 0s (respectively 1s) appearing as special entries of $M(L)$. Given a submatrix M of $M(L)$, we define $0_A(M)$ and $1_A(M)$ analogously.

We would like to evaluate the error of a probabilistic protocol P relative to a restriction matrix A. For a rejecting submatrix M (induced by P), we define $e_A(M) := (1_A(M)/1_A(L)) \cdot (0_A(L)/0_A(M))$. This quantity measures the ratio of incorrectly covered 1-entries divided by the ratio of correctly covered 0-entries. Obviously, $e_A(M)$ should be small for most rejecting matrices M. For a given constant δ, we evaluate P by

$$size_\delta(A, P) = \max\left\{\frac{0_A(M)}{0_A(L)} : e_A(M) < \delta \text{ and } M \text{ is a rejecting submatrix induced by } P\right\}.$$

Intuitively, $\text{size}_\delta(A, P)$ is proportional to the "size" of the largest submatrix M that does not introduce (relative to A) many errors.

LEMMA 2.1. *Let $\varepsilon < \frac{1}{2}$ and $\delta > \varepsilon/(1 - \varepsilon)$ be given constants. Assume that a probabilistic protocol P recognizes L with error ratio at most ε and worst-case message length m. Then, for any restriction matrix A, $m = \Omega(-\log(size_\delta(A, P)))$.*

Proof. Let P be some probabilistic protocol with error probability ε and worst-case message length m. Also, let $D_1, D_2, \cdots, D_{2^m}$ be its induced deterministic protocols. Let A be some assigned restriction matrix.

We first compute the ratio E of incorrectly covered 1-entries (R_i denotes the set of rejecting submatrices of D_i) as follows:

$$E = \frac{1}{2^m} \sum_{i=1}^{2^m} \sum_{M \in R_i} \frac{1_A(M)}{1_A(L)}$$

$$\geqq \frac{1}{2^m} \sum_{i=1}^{2^m} \sum_{M \in R_i, e_A(M) \geqq \delta} \delta \frac{0_A(M)}{0_A(L)} \quad \text{(by definition of } e_A(M)\text{)}.$$

On the other hand, the ratio of incorrectly covered 0-entries is bounded by ε. Therefore

$$\frac{1}{2^m} \sum_{i=1}^{2^m} \sum_{M \in R_i} \frac{0_A(M)}{0_A(L)} \geqq 1 - \varepsilon.$$

Since $E \leqq \varepsilon$, we can conclude that

$$\frac{1}{2^m} \sum_{i=1}^{2^m} \sum_{M \in R_i, e_A(M) < \delta} \frac{0_A(M)}{0_A(L)} \geqq 1 - \varepsilon - \varepsilon/\delta > 0.$$

Consequently,

$$\frac{1}{2^m} \sum_{i=1}^{2^m} \sum_{M \in R_i} size_\delta(L) \geqq 1 - \varepsilon - \varepsilon/\delta.$$

Since $\#R_i \leqq 2^m$, we get that $2^m\, size_\delta(L) = \Omega(1)$. □

We now consider *set intersection* as defined on a subset INPUT × INPUT of $\{0, 1\}^{2n} \times \{0, 1\}^{2n}$. INPUT consists of all words of length n over the alphabet $\{00, 01, 10\}$ with the letter 00 occurring exactly $n/2$ times and the remaining two letters occurring exactly $n/4$ times. We form the truth matrix *set*. The rows and columns of *set* are labeled with words from the set INPUT. An entry of *set* in row x and column y has value 1 if and only if there exists an i such that $x_i = y_i = 01$ or $x_i = y_i = 10$.

To find a lower bound for $C_\varepsilon(\textit{set intersection})$, we must find a suitable restriction matrix A. For our language *set intersection*, we introduce the notion of a *reason assignment* as a first step toward finding a proper restriction matrix.

DEFINITION 2.1. (a) We call an element $r \in \{0, 1\}^n$ a "reason";

(b) A function R with R :INPUT $\to \{0, 1\}^n$ is called a "reason assignment" for rows if and only if, for each $x \in$ INPUT,

(1) $R(x)$ is a reason,

(2) if $x_i = 01$, then $R(x)_i = 1$,

(3) if $x_i = 10$, then $R(x)_i = 0$.

(4) R is unrestricted for the letter 00;

(c) Analogously, we define a reason assignment for columns, but now 01 must be mapped into 0, and 10 must be mapped into 1.

Let $reason_r$ (respectively, $reason_c$) be a reason assignment for the rows (respectively, columns). Observe that $reason_r(x) = reason_c(y) = s$ implies that x and y have an empty set intersection. We interpret s as a "reason" for this empty set intersection.

DEFINITION 2.2. Let $reason_r$ and $reason_c$ be reason assignments for the rows and for the columns, respectively.

(a) An entry (x, y) in the matrix set is called *correct* if and only if $reason_r(x) = reason_c(y)$;

(b) We say that the entry (x, y) is *incorrect* if and only if there exists a position j such that

(1) $reason_r(x)$ and $reason_c(y)$ only differ in position j, and

(2) x and y intersect.

Observe that any correct entry corresponds to an empty set intersection, whereas an incorrect entry corresponds to an intersection in exactly one position. Finally, we need the notion of Kolmogorov complexity [9].

DEFINITION 2.3. (a) Let U be an universal Turing machine;

(b) Let x and y be words over $\{0, 1\}$. Then $K(x|y)$ is defined to be the length of the shortest program p, so that U on input (p, y) outputs x;

(c) Let S be a finite set of words. An element x of S is called Kolmogorov-random, given y if and only if $K(x|y) \geqq \log(\#S)$.

We can now define the family of restriction matrices. Given a probabilistic protocol P, we assign a pair of *Kolmogorov-random* reason assignments $reason_r$ (for the rows of *set*) and $reason_c$ (for the columns of *set*). The correct and incorrect entries specify a restriction matrix $A := A_P$, which we consider for the protocol P.

The properties of random restrictions guarantee that the ratio of 1-entries to 0-entries is bounded by $O(n)$ (see Lemma 3.4). In other words, *set intersection* is now far more balanced.

First, we describe the global structure of our proof. Let P be a probabilistic protocol recognizing *set intersection* with error ratio at most ε and worst-case message length at most μn (ε and μ are sufficiently small constants). Choose δ such that $\delta > \varepsilon/(1-\varepsilon)$. Also, let A be the assigned random restriction. Assume that M is an induced rejecting submatrix with

$$\frac{0_A(M)}{0_A(\textit{set intersection})} = \textit{size}_\delta(A, P) \quad \text{and} \quad e_A(M) < \delta.$$

We prove in Lemma 3.4 that Kolmogorov-random reason assignments must generate $\Omega(n^{-5/2}2^{2n})$ correct entries. According to Lemma 2.1, $0_A(M)$ must be large (otherwise, the message length would be large). Under this condition, we then show in § 5 that, for a Kolmogorov-random reason assignment R and its restriction matrix A, $e_A(M) > \delta$. Therefore we have derived the desired contradiction.

Now we give a more detailed sketch of our argument. In the next section, we prove some crucial technical properties of random reason assignments. In particular, we show that any Kolmogorov-random reason assignment (random given the probabilistic protocol P) forces almost all reasons to behave as expected in *every* submatrix induced by P, and therefore the reasons behave as expected for the given submatrix M.

We then restrict our attention to those reasons r, which appear for rows and columns of M, since only these reasons generate correct entries. We call a reason r *crucial* if a large number of rows and columns of M can be potentially assigned to r and if r behaves as expected. We show in Lemma 4.1 that it suffices to restrict our attention to these crucial reasons.

We obtain, in Lemma 4.2, the following property of a crucial reason r. For most bit positions of r, a constant proportion of all rows and all columns that are assigned to r possess the letter 00 in the corresponding position, and a constant proportion possesses the remaining letter. Therefore crucial reasons are "*unpredictable*" in many bit positions. We use this property in Lemma 4.3 to show that each crucial reason has $\Omega(n)$ crucial "adjacent" reasons (adjacent when interpreted on the n-dimensional hypercube).

Therefore, for each crucial row reason r, we obtain $\Omega(n)$ "adjacent" column reasons r_i. The given random restriction produces a too-large error if r (as a row reason) and r_i (as a column reason) are both unpredictable in the differing bit position, and the column reasons r and r_i are of roughly same size. The protocol errs on too many incorrect entries, since the number of incorrect entries covered by all pairs (r, r_i) is by a factor of $\Omega(n)$ larger than the number of correct entries covered by the pair (r, r). This last step is achieved in Lemma 5.1.

3. Expected behavior of reasons. We need the following versions of Chebyshev's inequality and Chernoff's bound.

FACT 3.1. *Consider N independent Bernoulli trials $X_1, \cdots, X_N$ with probability $p(p > 0)$ of success. Let $X = X_1 + \cdots + X_N$.*

(a) *For $c > 1$,* $\text{prob}[|X - E(X)| \geqq E(X)/c] < c^2/E(X)$;

(b) *If $m > Npe^2$, then* $\text{prob}[X \geqq m] \leqq e^{-m+Np}$.

Proof. (a) According to Chebyshev's inequality, $\text{prob}[|X - E(X)| \geqq t] < t^{-2}\,\text{Var}(X)$. Let $q = 1 - p$. Then $E(X) = Np$ and $\text{Var}(X) = Npq$. Since $\text{Var}(X) < E(X)$, we have that

$$\text{prob}[|X - E(X)| \geqq E(X)/c] < c^2\,\text{Var}(X)/(E(X))^2 < c^2/E(X).$$

(b) This inequality is due to Chernoff [4].

We now fix constants $\beta = 2^{-14}$ and $\mu = 2^{-18}$, where μn is the worst-case message length of the protocol P. Let (X, Y) denote the submatrix with X (respectively, Y) as its set of rows (respectively, columns). We formulate definitions and claims for rows only. The corresponding statements for columns are analogous.

DEFINITION 3.1. Let r be a reason.

(a) We say that r is *valid* for a row x of *set* if and only if there exists a reason assignment that maps x into r;

(b) $potential(r, X)$ is the set of those rows in X for which r is valid;

(c) $actual_R(r, X)$ is the set of rows in X that are actually mapped into r by the assignment R;

(d) We call a reason r *fat* for X if and only if $\#potential(r, X) \geq 2^{(1-2\beta)n}$;

(e) Given X and a row reason assignment R, we say that the reason r is good for X (and R) if and only if

(i) r is fat for X,

(ii) $\frac{1}{2}\#potential(r, X)2^{-n/2} \leq \#actual_R(r, X) \leq \frac{3}{2}\#potential(r, X)2^{-n/2}$.

(iii) For all positions $1 \leq j \leq n$ and for $b \in \{00, 01, 10\}$, let p_j^b be the number of rows in $potential(r, X)$ with letter b in position j, and let a_j^b be the number of rows assigned to r by R with letter b in position j. If $p_j^b > 2^{(1-2\beta)n}$, then

$$\tfrac{1}{2}\#p_j^b 2^{-n/2} \leq \#a_j^b \leq \tfrac{3}{2}\#p_j^b 2^{-n/2}.$$

Fix a probabilistic protocol P. Next, we investigate the actual size of a reason when restricted to submatrices induced by P. More precisely, we show that, for the purpose of determining the error performance of P, we can restrict ourselves to good reasons.

The sample space used in the following is the set for all reason assignments. We assume that all sample points are equally likely. Take a random reason assignment. We have the following proposition.

PROPOSITION 3.2. *Let r be a reason.*

(a) *Exactly $2^{n/2}$ reasons can be assigned to a row x in* INPUT;

(b) $\#potential(r, \text{INPUT}) = O(n^{-1}2^n)$;

(c) *With probability at least $1 - e^{-O(2^{n/2})}$, for all reasons s, $\#actual_R(s, \text{INPUT}) \leq e^2 n^{-1} 2^{n/2}$;*

(d) *With probability at least $1 - 2^{-4n/3}$, for every submatrix M (induced by P), at most $2^{(3/2-\beta)n}$ rows are assigned to reasons whose potential is at most $2^{(1-\beta)n}$.*

Proof. (a) This is obvious, since we restrict our attention to words in the set INPUT. All words in INPUT contain the letter 00 exactly $n/2$ times.

(b) The size of the potential is maximized for reasons with $n/2$ 0s and $n/2$ 1s. However, the number of rows that can be assigned to such a reason is

$$\binom{n/2}{n/4}^2 \leq \frac{2^n}{n}.$$

(c) The number of inputs actually assigned to a reason r is maximal if the potential of r is of maximal size. The size of $potential(r, \text{INPUT})$, however, is at most $2^n/n$. Therefore we must consider $2^n/n$ many independent Bernoulli trials with success probability $2^{-n/2}$. Applying Chernoff's bound, we get that

$$\text{prob}\,[\#actual_R(r, \text{INPUT}) \geq 2^{n/2}] \leq e^{-2^{n/2}}.$$

Therefore

$$\text{prob}\,[\text{for every reason } r, \#actual_R(r, \text{INPUT}) < 2^{n/2}] \geq 1 - 2^n e^{-2^{n/2}}.$$

(d) We say that a reason is *skinny* if its potential is of size at most $2^{(1-\beta)n}$. Now let $M = (X, Y)$ be a submatrix induced by the probabilistic protocol. The expected number of rows that can be assigned to skinny reasons is at most

$$E := \frac{2^n * 2^{(1-\beta)n}}{2^{n/2}} = 2^{(3/2-\beta)n}.$$

Now, by Fact 3.1, prob [# rows in X with skinny reasons $> 3E/2] \leqq 4/E$.

Therefore, with probability at least $1 - 4/E = 1 - 4*2^{-(3/2-\beta)n}$, at most $O(2^{(3/2-\beta)n})$ rows are assigned to skinny reasons. Since there are at most $2^{2\mu n}$ submatrices, the probability that $O(2^{(3/2-\beta)n})$ rows are assigned to skinny reasons in every submatrix induced by the protocol is at least $1 - 4*2^{-(3/2-\beta-2\mu)n} \geqq 1 - 2^{-4n/3}$ (by the choice of β and μ). □

The following lemma in conjunction with Proposition 3.2(d) shows that we can concentrate on good reasons only.

LEMMA 3.3. *With probability at least* $1 - 2^{-(1/40-4\beta)n}$, *for every submatrix* (X, Y) *induced by the probabilistic protocol* P, *all but* $2^{21n/40}$ *fat reasons are good for* X.

Proof. Consider a submatrix (X, Y) induced by P. The number of rows assigned to the reason r is determined by the number of successes in a sequence of $\#potential(r, X)$ independent Bernoulli trials (with success probability $2^{-n/2}$). Assume that the reason r is a fat reason. Then

$$\begin{aligned}\text{prob}\left[\frac{1}{2}\frac{\#potential(r,X)}{2^{n/2}} \leqq \#actual(r,X) \leqq \frac{3}{2}\frac{\#potential(r,X)}{2^{n/2}}\right] &\geqq 1 - \frac{4\,2^{n/2}}{\#potential(r,X)} \quad \text{(by Fact 3.1(a))} \\ &\geqq 1 - 4\,2^{-(1/2-2\beta)n} \quad (\text{since } \#potential(r,X) \geqq 2^{(1-2\beta)n}).\end{aligned}$$

For any reason and any position, the rows assigned to the reason contain only one of the two possible letters. Since there are only n positions, we must consider $2n + 1$ "potentials" for any reasons to determine whether it is good for X. Therefore the probability that one of the $2n + 1$ "potentials" violates the given condition is at most $4(2n + 1)2^{-(1/2-2\beta)n} \leqq 2^{-(1/2-3\beta)n}$. Let RA be the set of reason assignments. Since the probability that a fat reason is good for X is at least $1 - 2^{-(1/2-3\beta)n}$, there are at least $\#RA(1 - 2^{-(1/2-3\beta)n})$ reason assignments for which the fat reason is good for (X, Y). Let F be the set of fat reasons that occur as row reason for the submatrix (X, Y). Let us define a 0/1-matrix C, whose rows are the elements of F and whose columns are the elements in RA. An entry (r, R) in C is 1 if and only if the reason r is good for (X, Y) under the reason assignment R. Let RA^* be the set of those reason assignments R with $\sum_r C(r, R) \geqq \#F - 2^{21n/40}$. We are interested in estimating the number of reason assignments in RA^*. Observe that

$$\sum_{r\in F}\sum_{R\in RA} C(r,R) = \sum_{R\in RA}\sum_{r\in F} C(r,R) \geqq \#F\,\#RA(1 - 2^{-(1/2-3\beta)n}).$$

However,

$$\begin{aligned}\sum_{R\in RA}\sum_{r\in F} C(r,R) &= \sum_{R\in RA^*}\sum_{r\in F} C(r,R) + \sum_{R\in RA-RA^*}\sum_{r\in F} C(r,R) \\ &\leqq \#RA^*\,\#F + (\#RA - \#RA^*)(\#F - 2^{21n/40}).\end{aligned}$$

Therefore

$$\#RA^* \#F + (\#RA - \#RA^*)(\#F - 2^{21n/40}) \geqq \#F \#RA(1 - 2^{-(1/2-3\beta)n},$$

$$(\#RA - \#RA^*)2^{21n/40} \leqq \#F \#RA\, 2^{-(1/2-3\beta)n},$$

$$\#RA^*\, 2^{21n/40} \geqq \#RA(2^{21n/40} - 2^{(1/2+3\beta)n}) \quad (\text{since } \#F \leqq 2^n).$$

Therefore at most $\#RA\, 2^{-1/40-3\beta)n}$ reason assignments must be discarded because they induce more than $2^{21n/40}$ fat reasons that are not good for (X, Y). Since there are at most $2^{2\mu n}$ submatrices, the number of reason assignments to be discarded is at most $\#RA\, 2^{-(1/40-3\beta-2\mu)n}$. Therefore, with probability at most $1 - 2^{-(1/40-4\beta)n}$, for all submatrices induced by the probabilistic protocol, all but $2^{21n/40}$ fat reasons are good. □

We now fix a reason assignment R chosen at random given P. By the choice of β and μ, the probability of satisfying all the conditions of Proposition 3.2 and Lemma 3.3 is at least $(1 - 2^{-n/80})$. Therefore R also satisfies those conditions, and we can assume that Proposition 3.2 and Lemma 3.3 are valid for *every* submatrix induced by the protocol.

LEMMA 3.4. *For the matrix set* = (INPUT, INPUT),

(a) *The number of correct entries is* $\Omega(n^{-5/2}2^{2n})$, *and*

(b) *The ratio of the number of incorrect entries over the number of correct entries is at most* $4n$.

Proof. (a) There are $\Omega(2^n/\sqrt{n})$ reasons with exactly $n/2$ 1s. Among those reasons, at most $2*2^{21n/40}$ are not good for the rows or columns of *set*. We restrict our attention to those $\Omega(2^n/\sqrt{n}) - 2*2^{21n/40}$ good reasons. $\Omega(n^{-1}2^{n/2})$ rows are assigned to each such reason, and hence each such reason covers $\Omega(n^{-2}2^n)$ correct entries. Therefore there are $\Omega(2^n/\sqrt{n})\Omega(n^{-2}2^n) = \Omega(n^{-5/2}2^{2n})$ correct entries.

(b) Observe that there are at most n (neighbor) reasons that induce incorrect entries for a given reason. By Proposition 3.2(c), we have at most $e^2 2^{n/2}$ incorrect entries per row. Also, by Proposition 3.2(d), at most $O(2^{(3/2-\beta)n})$ rows are covered by reasons whose potential is at most $2^{(1-\beta)n}$. Therefore those reasons cover only $O(2^{(2-\beta)n})$ incorrect entries. Observe that this quantity is exponentially smaller than the number of correct entries.

At most, $2*2^{21n/40}$ fat reasons are not good for the rows or columns of *set*. By Proposition 3.2(c), the actual size of a reason is at most $e^2 n^{-1}2^{n/2}$. Therefore the number of incorrect entries covered by a reason is at most $e^4 n^{-1}2^n$. Consequently, the number of incorrect entries covered by fat but not good reasons is at most $2^{5n/3}$. Again, this quantity is exponentially smaller than the number of correct entries. Therefore it suffices to determine the ratio of the number of incorrect entries and correct entries covered by reasons that are good for *set*.

Observe that, for a good reason r, the actual sizes with respect to rows and with respect to columns vary only by a constant factor (because the size of its potential is the same for rows and columns). A good reason r (for rows and columns) has at most n neighbor reasons, and the actual size of each neighbor is larger by at most a factor of 3. Therefore the number of incorrect entries covered by r is by a factor of at most $3n$ larger than the number of correct entries it covers. Therefore, also considering the previously discarded incorrect entries, the total number of incorrect entries is at most $4n$ times larger than the total number of correct entries in the matrix *set*. □.

4. Crucial reasons. In this section, we work with the reason assignment R fixed in the previous section. We also pick one submatrix $M = (X, Y)$, which is induced by the protocol P. Now we focus on a subclass of good reasons that produces, "most" correct entries.

DEFINITION 4.1. We call a reason r *crucial* for M if and only if
(a) $\#potential(r, X) > 2^{(1-\beta)n}$ and $\#potential(r, Y) > 2^{(1-\beta)n}$, and
(b) $K(r|(X, Y)$, the reason assignment $R) \geqq 11n/20$.

The following claim shows that it suffices to consider only crucial reasons.

LEMMA 4.1. (a) *If r is a crucial reason for M, then r is good for the rows and columns of M;*
(b) *All but $o(2^{(1-\mu)2n})$ correct entries of M are covered by crucial reasons.*

Proof. (a) Let r be a crucial reason for M. Then r is fat. Now assume that r is not good for the rows of M. Then we can describe r by specifying r as one of the only $2^{21n/40}$ fat reasons that are not good (Lemma 3.3). This gives us the desired contradiction, since the resulting description is of length less than $11n/20$. With an analogous argument, we can establish that r is good for the columns of M.

(b) According to Proposition 3.2(d) at most $E = O(2^{(3/2-\beta)n})$ rows are covered by skinny reasons, namely, those reasons whose potential is bounded by $2^{(1-\beta)n}$. According to Proposition 3.2(c), every reason r has an actual size of at most $e^2n^{-1}2^{n/2}$. Therefore at most $O(2^{n/2}E) = O(2^{(2-\beta)n})$ correct entries are produced by skinny row reasons. Similarly, not more than this number of correct entries are produced by skinny column reasons.

Observe that only $2^{11n/20}$ reasons whose potential is greater than $2^{(1-\beta)n}$ can violate the Kolmogorov randomness condition in Definition 4.1. Since a reason can generate at most $O(2^n)$ correct entries, the number of correct entries covered by reasons violating the randomness condition is at most $O(2^{31n/40})$. By the choice of β and μ, all but $o(2^{(1-\mu)2n})$ correct entries are covered by crucial reasons. □

Our final goal in this section is to introduce and investigate the unpredictability of reasons. Intuitively, we would like to show that we can (almost) trust a set intersection predicted by adjacent crucial reasons.

DEFINITION 4.2. (a) We say that position i ($1 \leqq i \leqq n$) is *u-unpredictable* for X if and only if $u\#X \leqq \#\{s \in X : s_i = 00\} \leqq (1-u)\#X$;

(b) We call a reason r *u-unpredictable in bit position i* (relative to R) if and only if $actual_R(r, X)$ is u-unpredictable in position i.

LEMMA 4.2. *Let r be a good reason with $\#potential(r, X) \geqq 2^{(1-\beta)n}$. Then there is a set U of $(1-2^{-10})n$ positions such that*
(a) *$potential(r, X)$ is $(\frac{1}{4})$-unpredictable for all positions in U, and*
(b) *$actual_R(r, X)$ is $(\frac{1}{16})$-unpredictable for all positions in U.*

Proof. (a) Let F be the random variable that maps the rows of $potential(r, X)$ to themselves. Then, for the entropy H, we obtain that $H(F) = \log_2 \#potential(r, X) \geqq (1-\beta)n$. Assume that $potential(r, X)$ is not $(\frac{1}{4})$-unpredictable in $(1-2^{-10})n$ positions. We obtain that

$$H(F) \leqq \sum_{i=1}^{n} H(F_i) \leqq (1-2^{-10})n + H(\tfrac{1}{4})2^{-10}n, \quad \text{where } F_i \text{ is the } i\text{th projection of } F.$$

Since $H(\frac{1}{4}) < 0.9$, we get that $H(F) \leqq (1-2^{-10}/10)n$. Therefore $(1-\beta)n \leqq (1-2^{-10}/10)n$. This, however, is impossible, since $\beta = 2^{-14}$.

(b) Let j be a position that is $(\frac{1}{4})$-unpredictable for $potential(r, X)$. It suffices to show that j is $(\frac{1}{16})$-unpredictable for $actual_R(r, X)$. Without loss of generality, let us assume that only 00 and 01 can occur in position j. For $b \in \{00, 01\}$, we define

$$p_b := \#\{x \in potential(r, X) : x_j = b\} \quad \text{and} \quad \alpha_b := \#\{x \in actual_R(r, X) : x_j = b\}.$$

Since r is a good reason, we get that

$$\tfrac{1}{2}\, p_b 2^{-n/2} \leqq \alpha_b \leqq \tfrac{3}{2}\, p_b 2^{-n/2}.$$

Since potentials are $(\frac{1}{4})$-unpredictable, we have that $p_b \geqq \frac{1}{4}(p_{00} + p_{01})$. We also know that $\alpha_b \geqq \frac{1}{2} p_b 2^{-n/2} > \frac{1}{8}(p_{00} + p_{01})2^{-n/2}$. Therefore

$$\alpha_b \geqq \tfrac{1}{8}(p_{00} + p_{01})2^{-n/2} \geqq \tfrac{1}{16}(\alpha_{00} + \alpha_{01}). \qquad \square$$

DEFINITION 4.3. Let r and s be reasons. We say that s is a *neighbor* of r if and only if

(a) r and s differ in exactly one bit position, and

(b) The differing bit position is $(\frac{1}{4})$-unpredictable for *potential*(r, X) and *potential*(r, Y).

Observe that our definition is not symmetric concerning the unpredictability of the differing bit position. Now assume that s is a neighbor of a crucial reason r. Then

$$4\#\textit{potential}(s, X) \geqq \#\textit{potential}(r, X) \quad \text{and} \quad 4\#\textit{potential}(s, Y) \geqq \#\textit{potential}(r, Y).$$

Since the description complexity of r (given (X, Y) and the reason assignment R) is at least $11n/20$, the description complexity of s must be larger than $11n/20 - O(\log n)$. Therefore we can repeat the argument of Lemma 4.1(a) and obtain that s is good for X and Y. We obtain that $12\#\textit{actual}(s, X) \geqq \#\textit{actual}(r, X)$ and $12\#\textit{actual}(s, Y) \geqq \#\textit{actual}(r, Y)$.

We define $\psi(M)$ to be the number of correct entries covered by the submatrix M. We set $d := 2^{-10}$.

LEMMA 4.3. *Assume that* $\psi(M) = \Omega(2^{(1-\mu)2n})$. *Then all but* $o(\psi(M))$ *correct entries are covered by crucial reasons, where each such reason has at least* $(1 - 2d)n$ *crucial reasons as neighbors.*

Proof. Let r be a crucial reason. A position is *strongly unpredictable* if both *potential*(r, X) and *potential*(r, Y) are $(\frac{1}{4})$-unpredictable in this position. If a reason s differs from r only in position k, then we call s a k-neighbor of r.

We say that a crucial reason r is *unreliable* if there is one strongly unpredictable position j such that the j-neighbor s of r is not crucial. Observe that (by Lemma 4.2) there are at least $(1 - 2d)n$ strongly unpredictable positions. Since all but $o(\psi(M))$ correct entries are covered by crucial reasons (by Lemma 4.1), the claim follows if we show that only $o(\psi(M))$ correct entries are covered by unreliable reasons.

Now let us assume that the crucial reason r is strongly unpredictable in position j, but its j-neighbor s is not crucial. Since j is strongly unpredictable, we obtain that

$$4\#\textit{potential}(s, X) \geqq \#\textit{potential}(r, X) \quad \text{and} \quad 4\#\textit{potential}(s, Y) \geqq \#\textit{potential}(r, Y).$$

As a consequence of the argument preceding this lemma, we have that

$$12\#\textit{actual}(s, X) \geqq \#\textit{actual}(r, X) \quad \text{and} \quad 12\#\textit{actual}(s, Y) \geqq \#\textit{actual}(r, Y).$$

Therefore the expected number E_1 of correct entries covered by noncrucial neighbors is at least proportional to the expected number E_2 of correct entries covered by unreliable reasons. We know from Lemma 4.1, however, that all but $o(2^{2n}/2^{2\mu n}) = o(\psi(M))$ correct entries are covered by crucial reasons. Since $E_2 = O(E_1) = o(\Psi(M))$, only $o(\psi(M))$ correct entries are covered by unreliable reasons. $\square$

5. Error accounting. Assume that an adversary presents a probabilistic protocol P with error ratio $\epsilon < d^4$ and worst-case message length $m < \mu n$. Choose $\delta = 2\varepsilon/(1 - \varepsilon)$. We pick a Kolmogorov-random restriction A such that Proposition 3.2 and Lemma 3.3 are valid. Our adversary counters by selecting a rejecting submatrix M that is induced by P. Next, we determine the "size" of M relative to A. If $0_A(M) = O(2^{(1-\mu)2n})$, then we have won.

Why? By Lemma 3.4(b), we know that the truth matrix *set* possesses $\Omega(n^{-5/2}2^{2n})$ correct entries (relative to A). So, if $0_A(M) = O(2^{(1-\mu)2n})$, then the message length m must be at least μn (according to Lemma 2.1). Otherwise, applying Lemma 2.1 again, we can defeat our adversary by showing that $e_A(M) > \delta$, since this shows that his last move was *illegal*.

Our objective is now to estimate the number of incorrect entries inside a submatrix M that contains at least $2^{(1-\mu)2n}$ correct entries. More precisely, we must show that M covers $\Omega(n)$ times as many incorrect entries as there are correct entries. By Lemma 4.3, we know that almost all correct entries are covered by crucial reasons that have at least $(1-2d)n$ crucial reasons as neighbors. Unfortunately, a row assigned to a row reason r and a column assigned to a neighbor reason s need not produce an incorrect entry. The danger is that s may contain only the letter 00 in the differing bit position. We now strengthen Lemma 4.3 by showing a constant proportion of the correct entries is covered by crucial reasons that have $\Omega(n)$ neighbors, where the differing bit position is unpredictable for all involved reasons. First we must introduce some notation.

DEFINITION 5.1. Two crucial reasons r and s are adjacent if and only if

(a) The two reasons differ in exactly one bit position;

(b) The differing bit position is $(\frac{1}{4})$-unpredictable for *potential*(r, X) and *potential*(r, Y); and

(c) The differing bit position is $(\frac{1}{4})$-unpredictable for *potential*(s, X) and *potential*(s, Y).

We now summarize the properties of crucial reasons.

Property (a). Each crucial reason is predictable in at most dn positions.

Property (b). All but $o(\Psi(M))$ correct entries are covered by crucial reasons, where each such crucial reason has at least $(1-2d)n$ crucial reasons as neighbors.

Property (c). If s is a neighbor of a crucial reason r, then

$$12\ \#actual(s, X) \geqq \#actual(r, X) \quad \text{and} \quad 12actual(s, Y) \geqq \#actual(r, Y).$$

Property (d). If r and s are adjacent, then they cover at least

$$(\tfrac{1}{16})^2(\#actual(r, X)\#actual(s, Y) + \#actual(s, X)\#actual(r, Y))$$

incorrect entries. (We may lose a factor of $(\frac{1}{16})^2$ because, according to Lemma 4.2(b), the *actual* sets *are* $(\frac{1}{16})$-unpredictable in the differing bit position.)

Our goal is to find a set of crucial reasons covering a constant proportion of the correct entries of M such that each crucial reason in the set is adjacent to $\Omega(n)$ crucial reasons. Our construction is graph theoretical.

We construct a directed graph $G(M) = (V, E)$ with crucial reasons as its set of vertices. We insert an arc (r, s) if and only if s is a neighbor of r. We define *weight*(r) to be the number of correct entries of M covered by reason r. We also define the weight of an edge e to be the weight of its tail. The directed graph $G(M)$ has the following weight restrictions.

Restriction (a). The set of vertices of outdegree less than $(1-2d)n$ has a cumulative weight of at most $o(\Psi(M))$ (by Lemma 4.3).

Restriction (b). As a consequence of Property (c), if s is a neighbor of r, then 144 *weight*$(s) \geqq$ *weight*(r).

Set $\rho := (d/3)$. As a first step, we prove in the following lemma that the cumulative weight of all vertices of indegree greater than $(2d+\rho)n$ is $\Omega(\psi(M))$. By Restriction (a), however, the cumulative weight of vertices of outdegree less than $(1-2d)n$ is $o(\Psi(M))$. Observe that if a vertex has indegree larger than $(2d+\rho)n$ and outdegree larger than

$(1 - 2d)n$, then its corresponding reason is adjacent (in the sense of Definition 5.1) to at least ρn reasons. Therefore we have reached our goal.

We need the following definition. Let $S = \{v \in V : \text{indegree}(v) < (2dn + \rho n)\}$ and $B = V - S$.

LEMMA 5.1. *It holds that* $\sum_{a \in B} weight(a) \geqq 2^{-9} \sum_{a \in V} weight(a)$.

Proof. Set $W := \sum_{a \in V} weight(a)$. By Restriction (a),

$$\sum_{e \in E} weight(e) \geqq (\tfrac{4}{5})(1 - 2d)nW \geqq (0.8 - 2d)nW.$$

However, the total weight of those edges that are directed into vertices in S is at most $144(2dn + \rho n)W$ because of Restriction (b) and the indegree bound on the vertices in S. Therefore the total weight of those edges that are directed into vertices in B is at least $(0.8 - 2d - 288d - 144\rho)nW$. Since the indegree of any vertex is at most n, we have that

$$\begin{aligned}\sum_{a \in B} weight(a) &\geqq (\tfrac{1}{144})(0.8 - 290d - 144\rho)W \\ &\geqq (\tfrac{1}{144})(0.8 - 350d)W \\ &\geqq 2^{-9}W, \quad \text{since } 350d < 0.35 < 0.8. \end{aligned}$$ □

Consider a crucial reason r that is adjacent to ρn crucial reasons $r_1, \cdots, r_{\rho n}$. Observe that the bit position in which r and r_i differ is β-unpredictable for both r and r_i. Now we must observe that, for each r_i, $12\#actual(r_i, X) \geqq \#actual(r, X)$.

To avoid double counting of incorrect entries, we count incorrect entries induced by the column reason r_i and the row reason r for each crucial reason r that is adjacent to ρn crucial reasons. The number of incorrect entries induced by the pair (r, r_i) of reasons is by a factor of $f = (\frac{1}{16})^2(\frac{1}{12})(f > 2^{-12})$ larger than the number of correct entries covered by r. (Since the *actuals* are $\frac{1}{16}$-unpredictable for both the reasons, we may lose a factor of at most $(\frac{1}{16})^2$. We may also lose a factor of up to $(\frac{1}{12})$ due to the difference in the size of the two *actuals*.) Therefore, for each such crucial reason r, we obtain at least $(2^{-12}\rho n * weight(r))$ incorrect entries as compared to $weight(r)$ correct entries. Therefore, as a consequence of Lemma 5.1, there are at least $2^{-(12+9+1)}\rho n\psi(M) \geqq 8d^4 n\psi(M)$ incorrect entries in the submatrix $M = (X, Y)$.

Therefore we have that $1_A(M) \geqq 8d^4 n 0_A(M)$ and $1_A(L) \leqq 4n0_A(L)$ (by Lemma 3.4). Hence $e_A(M) \geqq 2d^4 > 2\varepsilon > \delta$.

REFERENCES

[1] H. ABELSON, *Lower bounds on information transfer in distributed computations*, in Proc. 19th Annual Symposium on Foundations of Computer Science, 1978, Ann Arbor, MI, pp. 151–158.

[2] A. V. AHO, J. D. ULLMAN, AND M. YANNAKAKIS, *On notions of information transfer in* VLSI *circuits*, in Proc. 14th Annual ACM Symposium on Theory of Computing, 1983, Boston, MA, pp. 133–139.

[3] L. BABAI, P. FRANKL, AND J. SIMON, BPP *and the Polynomial Time Hierarchy in Communication Complexity Theory*, in Proc. 27th Annual Symposium on Foundations of Computer Science, 1986, Toronto, Ontario, Canada, pp. 337–347.

[4] H. CHERNOFF, *A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations*, Ann. Math. Statist., 23 (1952), pp. 493–507.

[5] B. CHOR AND O. GOLDREICH, *Unbiased bits from weak sources of randomness*, in Proc. 26th Annual Symposium on Foundations of Computer Science, 1985, Portland, OR, pp. 429–442.

[6] P. DURIS, Z. GALIL, AND G. SCHNITGER, *Lower bounds on communication complexity*, in Proc. 16th Annual ACM Symposium on Theory of Computing, 1984, Washington, DC, pp. 81–91.

[7] B. HALSTENBERG AND R. REISCHUK, *On different modes of communication*, in Proc. 20th Annual ACM Symposium on Theory of Computing, 1988, Chicago, IL, pp. 162–172.

[8] J. JÁ JÁ, P. KUMAR, AND J. SIMON, *Information transfer under different sets of protocols*, SIAM J. Comput., 13 (1984), pp. 840–849.

[9] A. N. KOLMOGOROV, *Three approaches to the quantitative definition of information*, Problems Inform. Transmission, 1 (1965), pp. 1–7.

[10] B. KALYANASUNDARAM AND G. SCHNITGER, *The probabilistic communication complexity of set intersection*, in Proc. 2nd Annual Structure in Complexity Theory Conference, 1987, Ithaca, NY, pp. 41–49.

[11] K. MEHLHORN AND E. M. SCHMIDT, *Las Vegas is better than determinism in* VLSI *and distributed computing*, in Proc. 14th Annual ACM Symposium on Theory of Computing, 1982, San Francisco, CA, pp. 330–337.

[12] C. H. PAPADIMITRIOU AND M. SIPSER, *Communication complexity*, in Proc. 14th Annual ACM Symposium on Theory of Computing, 1982, San Francisco, CA, pp. 196–200.

[13] R. RAZ AND A. WIGDERSON, *Monotone circuits for matching require linear depth*, in Proc. 22nd Annual ACM Symposium on Theory of Computing, 1990, Baltimore, MD, pp. 287–292.

[14] A. A. RAZBOROV, *On the distributional complexity of disjointness*, in Proc. 17th International Colloquium on Automata, Languages and Programming, Springer-Verlag, Lecture Notes in Computer Science, no. 443, 1990, Warwick University, England.

[15] A. C. YAO, *Lower bounds by probabilistic arguments*, in Proc. 24th Annual Symposium on Foundations of Computer Science, 1983, Tucson, AZ, pp. 420–428.

[16] ———, *Some Complexity Questions Related to Distributed Computing*, in Proc. 11th Annual ACM Symposium on Theory of Computing, 1979, Atlanta, GA, pp. 209–213.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 558–563, November 1992
© 1992 Society for Industrial and Applied Mathematics
011

A TIGHT LOWER BOUND ON THE SIZE OF PLANAR PERMUTATION NETWORKS*

MARIA KLAWE† AND TOM LEIGHTON‡

Abstract. A tight lower bound is proved on the minimum number of vertices in a planar graph in which any permutation between t distinguished vertices can be realized by vertex disjoint paths.

Key words. planar graphs, permutation layouts, switching, circuits

AMS(MOS) subject classifications. 05C10, 05C35, 68E10, 94C15

1. Introduction. We define a *t-permutation network* to be a graph G with t distinguished vertices called terminals, with the property that, for any one-to-one pairing $\{(x_i, y_i)\}$ among the terminals, there is a set $\{P_i\}$ of vertex-disjoint paths in G with P_i joining x_i to y_i for each i. Permutation networks obviously have many applications in communication networks, but they have also received substantial attention in the context of permutation layouts, a basic tool in the layout of printed circuits and large scale integrated chips (see [CS], [KKF], [S], [TK], [AKLLW1], [AKLLW2], [AKS]).

A *permutation layout* is a permutation network where the graph G is a rectangular (two-dimensional) grid graph. The definition of permutation layout sometimes includes additional assumptions, such as the assumption that the terminals are partitioned into inputs and outputs with only one-to-one pairings between inputs and outputs considered. Since it is possible to modify our definition and result in a straightforward manner to correspond to these variants, we restrict our attention to the case described here.

One of the key questions concerning permutation layouts is how large a rectangle is needed to construct a t-permutation layout, since this influences how densely circuits can be laid out on chips. Examples of rectangular grids with $O(t^3)$ area that contain t-permutation layouts (and simple algorithms for finding the routings of the connecting paths) were given by Cutler and Shiloach in [CS], who also proved that, if all the terminals lie on at most two horizontal lines of the grid, then the rectangle must have area at least $\Omega(t^{2.5})$. Techniques very similar to those given by Cutler and Shiloach are commonly used in circuit layout. In [AKLLW1], Aggarwal et al. proved an $\Omega(t^3)$ lower bound on the area of t-permutation layouts, showing that the Cutler–Shiloach techniques are asymptotically optimal. This result raises two obvious questions. Can the area needed be reduced by using multiple layers of grids, or by using some other planar graph instead of rectangular grids? Since area is not an appropriate measure for planar graphs, in the second question, area is replaced by number of vertices as these two measures essentially agree on grids.

The first question is addressed in [AKLLW2], where the $\Omega(t^3)$ lower bound on area is extended to multilayer grid permutation networks with the restriction that some (arbitrarily small) fixed fraction of the connecting paths do not change layers. The restriction

* Received by the editors July 22, 1990; accepted for publication (in revised form) February 2, 1992.

† Department of Computer Science, University of British Columbia, Vancouver, British Columbia, V6T 1Z2, Canada. This research was partially supported by a Natural Sciences and Engineering Research Council of Canada operating grant.

‡ Department of Mathematics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139. This research was partially supported by Defense Advanced Research Projects Agency contracts N00014-87-K-825 and N00014-89-J-1988, U.S. Air Force contract AFOSR-89-0271, and U.S. Army contract DAAL-03-86-K-0171.

that a fixed fraction of the paths do not change layers is essential, since the standard crosspoint switch is a two layer t-permutation layout with $O(t^2)$ area, in which every routing path changes layers once. Despite the reduction of area obtainable with the use of layer changes, the practical advantages of avoiding layer changes continue to make planar permutation networks a useful tool in circuit layout. Thus the second question remains a significant issue. The purpose of this paper is to answer the second question by proving an $\Omega(t^3)$ lower bound on the number of vertices in a planar t-permutation network, showing that the current grid-based techniques are asymptotically optimal.

Like the lower bound for t-permutation grid graphs in [AKLLW1], [AKLLW2], our proof uses the permutation property of the graph to simulate a planar embedding of an expanding graph on $\Omega(t)$ vertices and then applies the quadratic lower bound on the crossing number of expanding graphs to get the desired $\Omega(t^3)$ lower bound. However, we also use an additional tool, namely, the existence of weight-balanced separators for planar graphs. Combining these two techniques results in a proof that is more general and simpler than the ones for grid graphs given in [AKLLW1] and [AKLLW2].

2. The lower bound. Let G be a t-permutation network with n vertices. We first note that we may assume that G has maximum degree 3, since replacing the edges adjacent to each vertex of higher degree with a binary tree connecting the vertex to its neighbours only increases the number of vertices by at most a constant factor, and does not affect the permutation property. In addition, we may assume that G is connected since all the terminals must lie in the same connected component of a permutation network, and the connected component will itself be a permutation network. Finally, we may assume that each terminal has degree 1, since, if necessary, we can hang a new terminal vertex from each original terminal.

We start the section by describing the weight-balanced separator theorem (Theorem 2.1) and one of its corollaries, culminating with the formulation we will actually apply, the balanced terminal separator lemma (Lemma 2.2). We then give the version of the lower bound on crossing number (Lemma 2.3), which we need, and close with the proof of the lower bound on the number of vertices in a planar permutation network (Theorem 2.4).

The weight-balanced separator is a generalization of the weighted version of the planar separator theorem given in [LT]. Specifically, the original theorem in [LT] proves that, if every vertex in an n-vertex planar graph has a weight, then there is a set of $O(\sqrt{n})$ edges and vertices whose removal splits the graph into two subsets so that each subset contains at most half the total weight. In the generalization, vertices have several different weights, and we seek a separator that simultaneously splits all the weights in half. The precise statement is as follows.

THEOREM 2.1 (weight-balanced separator). *Given an n-vertex planar graph where each vertex has a k-vector of weights, the graph can be split into two subsets by removing $O(k\sqrt{n})$ edges and vertices, such that, for each component of the weight vector, the total component weight of each subset is at most half the total component weight of the graph.*

A weaker form of this theorem was first proved by Leighton in [L2], using a combinatorial result on splitting necklaces of coloured beads. A stronger and very elegant form of the necklace-splitting result was proved by Goldberg and West [GW], though with a rather lengthy and involved proof. Alon and West [AW] later gave a very simple proof based on the Borsuk–Ulam "ham sandwich" theorem from topology. The proof of the weight-balanced separator theorem in its full generality can be found in the last two lectures of [LLS], though, in fact, the special cases found in [L2], [BL], and [GW, Thm. 4, p. 104] would suffice for our purposes.

It is well known and easy to prove by iteratively applying the original weighted planar separator theorem that, for any p and any weighted planar graph G of bounded degree, there exist $O(\sqrt{pn})$ edges whose removal splits G into p pieces each having at most $1/p$ of the total weight. Similarly, by using the weight-balanced separator theorem and assigning each vertex a pair of weights (one being the vertex's original weight, and the second, the number of removed edges that are adjacent to the vertex), it is easy to prove the following stronger result. For any p and any weighted planar graph G of bounded degree, there exist $O(\sqrt{pn})$ edges whose removal splits G into p pieces each having at most $1/p$ of the total weight, and such that each piece is adjacent to $O(1/p)$ of the removed edges. More precisely, we begin by finding $O(\sqrt{n})$ edges whose removal splits G into two pieces, each having half of the vertices. At the next step, an additional $O(\sqrt{n/2})$ edges are removed from each piece to produce a total of four pieces, with each piece containing quarter of the vertices, and with each piece adjacent to at most half of the edges removed at the first stage. At the next stage, $O(\sqrt{n/4})$ edges are removed from each piece to produce a total of eight pieces, with each piece containing one-eighth of the vertices. Moreover, each piece is adjacent to at most half of the edges removed during the first two stages that were adjacent to its "parent" piece at the second stage. Thus the piece is adjacent to at most $O(\sqrt{n}/4 + \sqrt{n/2}/2)$ edges. After iterating this procedure k times, we obtain 2^k pieces with each piece containing $1/2^k$ of the vertices, and with each piece adjacent to $O(\sum_{i=0}^{k-1} \sqrt{n/2^i}/2^{k-i})$ of the edges that have been removed. Such decompositions are called fully balanced decompositions and are discussed in detail in the last two lectures of [LL3] and in [BL]. Applying the fully balanced decomposition result in the context of planar permutation networks yields the following lemma.

LEMMA 2.2 (balanced terminal separator). *Given a bounded degree n-vertex planar graph with t terminals each having degree* 1, *there exist* $O(\sqrt{nt})$ *edges whose removal results in a graph such that each connected component is incident to* $O(\sqrt{n/t})$ *removed edges and each terminal is its own component.*

Proof. First, remove the t edges adjacent to terminals. Assign each vertex a weight equal to the number of removed edges adjacent to it. Now taking $p = t$, a fully balanced decomposition of this weighted graph has the desired properties. □

LEMMA 2.3 (crossing pairs). *There exists a constant* $c > 0$ *such that, for each s, there is an s-vertex graph H of degree at most* 3, *such that, for each planar embedding of H, there are at least* cs^2 *distinct pairs of edges that cross each other.*

Proof. We first note that in any planar embedding of a graph H with the minimum number of edge-crossings, each pair of edges crosses at most once. To see this, suppose that we have an embedding and that e and e' are edges that cross each other more than once. Let x and y be consecutive crossings between e and e'. The crossings at x and y can be eliminated by rerouting e' and e so that each follows the other's path between x and y (see Fig. 1), and hence the embedding could not have had the minimum number of crossings. Given this observation the lemma follows immediately from the well-known fact that there are expanding graphs of degree 3 and Leighton's quadratic lower bound on the crossing number of expanding graphs [L3]. □

We are now ready to prove the desired lower bound.

THEOREM 2.4. *If G is a connected n-vertex planar t-permutation network of degree at most* 3, *then* $n = \Omega(t^3)$.

Proof. By the terminal separator lemma, there is a set R of $O(\sqrt{nt})$ edges of G whose removal results in a graph such that each connected component is incident to $O(\sqrt{n/t})$ removed edges and each terminal is its own component. Let $G \backslash R$ be the graph obtained by removing the edges in R from G, and let G^c be the graph obtained from G by contracting every edge of G that is not in R, and then removing multiple edges. An

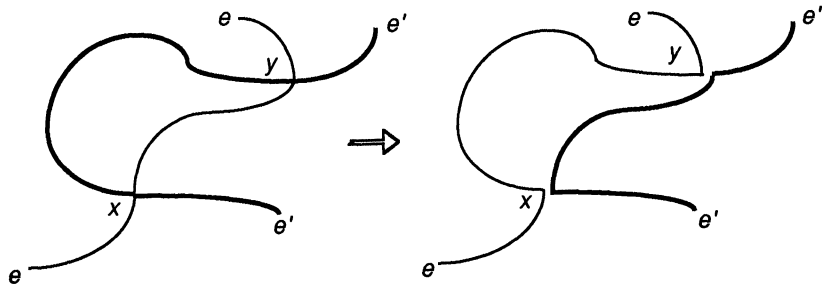


FIG. 1

example is shown in Fig. 2. It is easy to see that each vertex of G^c corresponds to a connected component of $G \setminus R$ and that G^c is a connected planar graph with $O(\sqrt{n/t})$ maximal degree. We will refer to a vertex of G^c as a *terminal node* if the corresponding connected component of $G \setminus R$ is a terminal. We will assume that we have a fixed embedding of G in the plane.

Let T^c be a subtree of G^c whose leaves are the terminal nodes. Such a tree can be obtained, for example, by taking a spanning tree of G^c and chopping off all branches that contain no terminal nodes. Since each terminal node has degree 1 in G and hence in G^c, it must be a leaf of any spanning tree of G^c, and hence the leaves of this tree will be exactly the terminal nodes. Let T be a subtree of G that maps onto T^c; i.e., T is obtained by replacing each edge of T^c with a representative edge in R and replacing each vertex of T^c with a subtree of the component corresponding to that vertex in G^c. An example is shown in Fig. 3. Note that the leaves of T are the terminals.

Let Q be a simple curve in the plane connecting the terminals, with Q running alongside the induced embedding of the edges of T in the plane, picking up the terminals as illustrated in Fig. 4. Q is assumed to be routed sufficiently closely to T so that it only intersects edges of G when it runs past a vertex of T, where it may have to cross an edge in $G \setminus T$ that is adjacent to the vertex. We label the terminals $z_1, \cdots, z_t$ in the order in which they are first visited by Q.

We will call each portion of Q joining a pair of consecutive terminals a *link* and say that a link *runs through a component* of $G \setminus R$ if it runs past some vertex in the component. It will be important to keep in mind that links are not part of any of the graphs but merely simple curves lying in the plane in which the graphs are embedded. Since each link starts and ends at a terminal, and at most two links can run alongside any edge in T, it is easy to see that, if y is a vertex of degree d in T^c, then at most $2d$ links can run through the component C_y represented by y. Let $C_1, \cdots, C_m$ be the

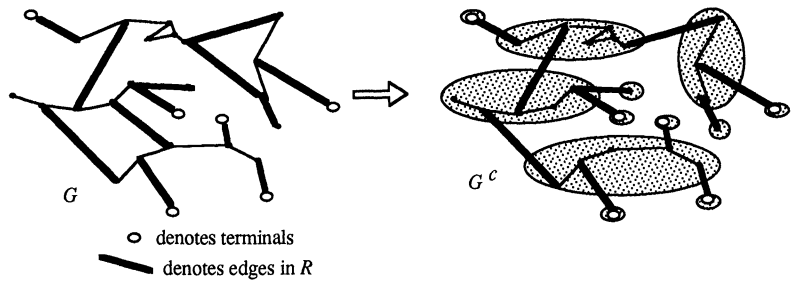


FIG. 2

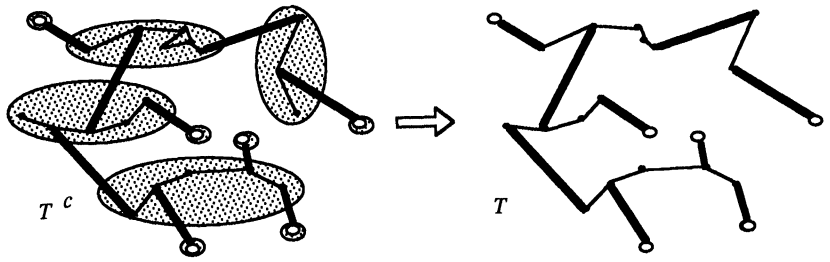


FIG. 3

components of $G\backslash R$, and, for each i, let n_i be the number of links that run through C_i. We now show that $\sum_{n_i>4} n_i \leqq 6t$. Let d_i be the degree of the vertex representing C_i in T^c. We already noted that $n_i \leqq 2d_i$, and hence it suffices to prove that $\sum_{d_i>2} d_i \leqq 3t$. First, note that Δ, the number of vertices of degree at least 3 in T^c, is at most the number of leaves of T^c and that T^c has exactly t leaves since its leaves are the terminal nodes. Next, since the average degree in any tree is less than 2, we have $\sum (d_i - 2) < 0$. Now $\sum (d_i - 2) = -t + \sum_{d_i>2} (d_i - 2) = -t + \sum_{d_i>2} d_i - 2\Delta$, and hence $\sum_{d_i>2} d_i < t + 2\Delta \leqq 3t$, as desired.

Now, taking $s = t/3$ in Lemma 2.3, suppose that H is a degree-3 graph on $t/3$ vertices $v_1, \cdots, v_{t/3}$ such that, for each planar embedding of H, there are at least $c(t/3)^2$ distinct pairs of edges that cross each other. We want to use the planar embeddings of G and Q to produce an embedding of H in the plane. We use z_{3h-1} to represent v_h for each h. Let $Z_h = \{z_{3h-2}, z_{3h-1}, z_{3h}\}$. Now let $\{(x_i, y_i)\}$ be a one-to-one pairing of the terminals of G such that, for each edge (v_j, v_k) in H, there is some i such that $x_i \in Z_j$ and $y_i \in Z_k$. This is easy to do, since H is of degree at most 3 and $|Z_h| = 3$ for each h. Now each edge (v_j, v_k) of H is embedded as the corresponding permutation path P_i, plus possibly a link at one or both ends to complete the connection to its endpoints. Note that each link is used by at most one edge of H. Since the P_i are mutually disjoint and the links are also mutually disjoint except at possibly the vertices of H, two embedded edges of H can only cross if one of the edges' permutation paths crosses a link used by the other edge. Thus there are $\Omega(t^2)$ distinct pairs of permutation paths and links that cross each other. By the choice of Q, each such crossing can only occur when the link runs past a vertex that is an endpoint of an edge in the permutation path. We will say a permutation path and link cross inside a component of $G\backslash R$ if the vertex is in that component.

For each i, let r_i be the number of edges in R incident to the connected component C_i. Since there is at most one permutation path using each edge in R, the number of

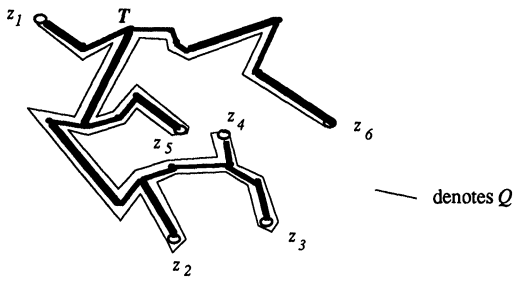


FIG. 4

permutation paths that pass through C_i is at most r_i, and hence the number of distinct pairs of permutation paths and links that cross inside C_i is at most $n_i r_i$. Let α be the total number of distinct pairs that cross. We have

$$\alpha \leqq \sum_{i=1}^{m} n_i r_i \leqq 4 \sum_{n_i \leqq 4} r_i + \sum_{n_i > 4} n_i r_i.$$

Clearly, we have $\sum r_i \leqq 2|R|$, and we proved earlier that $\sum_{n_i>4} n_i \leqq 6t$. In addition, we have $|R| = O(\sqrt{nt})$ and $\max\{r_i\} = O(\sqrt{n/t})$. Thus we have $4 \sum_{n_i \leqq 4} r_i = O(|R|) = O(\sqrt{nt})$ and $\sum_{n_i>4} n_i r_i \leqq 6t \max\{r_i\} = O(\sqrt{nt})$ also. Hence $\alpha = O(\sqrt{nt})$. Finally, combining this with the lower bound $\alpha = \Omega(t^2)$ implies that $n = \Omega(t^3)$, as desired. □

REFERENCES

[A] N. ALON, *Splitting necklaces*, Adv. Math., 63 (1987), pp. 247–253.

[AKS] A. AGGARWAL, M. KLAWE, AND P. SHOR, *Multi-layer grid embeddings for* VLSI, Algorithmica, 6 (1991), pp. 129–151.

[AKLLW1] A. AGGARWAL, M. KLAWE, D. LICHTENSTEIN, N. LINIAL, AND A. WIGDERSON, *Multilayer grid embeddings*, in Proc. IEEE 26th Annual Symposium on the Foundation of Computer Science, Portland, OR, 1985, pp. 186–196.

[AKLLW2] ———, *A lower bound on the area of permutation layouts*, Algorithmica, 6 (1991), pp. 241–255.

[AM] N. ALON AND V. MILMAN, λ_1, *isoperimetric inequalities for graphs, and superconcentrators*, J. Combin. Theory Ser. B, 38 (1985), pp. 73–88.

[AW] N. ALON AND D. WEST, *The Borsuk–Ulam theorem and bisection of necklaces*, Proc. Amer. Math. Soc., 98 (1986), pp. 623–628.

[BL] S. BHATT AND F. T. LEIGHTON, *A framework for solving* VLSI *graph layout problems*, J. Comput. System Sci., 28 (1984), pp. 300–343.

[BP] N. K. BOSE AND K. A. PRABHU, *Thickness of graphs with degree constrained vertices*, IEEE Trans. Circuits and Systems, CAS-24 (1977), pp. 184–190.

[CS] M. CUTLER AND Y. SHILOACH, *Permutation layout*, Networks, 8 (1978), pp. 253–278.

[GW] C. H. GOLDBERG AND D. B. WEST, *Bisection of circle colorings*, SIAM J. Algebraic Discrete Meth., 6 (1985), pp. 93–106.

[K] Y. KAJITANI, *On via hole minimization of routings on a two-layer board*, Tech. Report, Dept. of Electrical Engineering, Tokyo Institute of Technology, Tokyo, Japan, 1980.

[KKF] E. S. KUH, T. KASHIWABARA, AND T. FUJISAWA, *On optimum single row-routing*, IEEE Trans. Circuits and Systems, CAS-26 (1979), pp. 361–368.

[L1] F. T. LEIGHTON, *Layouts for the shuffle-exchange graph and lower bound techniques for* VLSI, Ph.D. thesis, Dept. of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, 1981.

[L2] ———, *A layout strategy for* VLSI *which is provably good*, in Proc. ACM Annual Symposium on the Theory of Computing, San Francisco, 1982, pp. 85–98.

[L3] ———, *New lower bound techniques for* VLSI, Math Systems Theory, 17 (1984), pp. 47–70.

[LL3] F. T. LEIGHTON, C. E. LEISERSON, AND E. SCHWABE, *Theory of parallel and* VLSI *computation*, MIT/LCS/RRS 6, Research Seminar Series, Lecture Notes for 18.435/6.848, March 1989.

[LT] R. J. LIPTON AND R. E. TARJAN, *A separator theorem for planar graphs*, SIAM J. Appl. Math., 36 (1979), pp. 177–189.

[R] R. RIVEST, *The placement and interconnect system*, in Proc. 19th Design Automation Conference, 1982, pp. 475–481.

[S] I. SHIRAKAWA, *Letter to the editor: Some comments on permutation layout*, Networks, 10 (1980), pp. 179–182.

[SO] H. C. SO, *Some theoretical results on the routing of multilayer printed wiring boards*, in Proc. IEEE International Symposium on Circuits and Systems, 1974, pp. 296–303.

[TK] S. TSUKIYAMA AND E. S. KUH, *Double-row planar routing and permutation layout*, Networks, 12 (1982), pp. 287–316.

[TKS] B. S. TING, E. S. KUH, AND I. SHIRAKAWA, *The multilayer routing problem: Algorithms and necessary and sufficient conditions for the single-row single layer case*, IEEE Trans. Circuits and Systems, CAS-23 (1979), pp. 768–778.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 564–570, November 1992
© 1992 Society for Industrial and Applied Mathematics
012

HAMILTONIAN PROPERTIES OF GRID GRAPHS*

CHRISTINA ZAMFIRESCU† AND TUDOR ZAMFIRESCU‡

Abstract. This paper presents sufficient conditions for a grid graph to be Hamiltonian. It is proved that all finite grid graphs of positive width have Hamiltonian line graphs.

Key words. Hamiltonian graph, infinite Hamiltonian path, grid graph, line graph

AMS(MOS) subject classification. 05C45

1. Introduction. The interest in subgraphs of the graph determined by the square lattice in $\mathbb{R}^2$ is mainly explained by the importance of the lattice itself, due, in turn, to its frequent occurrence in various mathematical fields. Hamiltonian properties of such graphs have been previously examined by several authors (see, for example, [2]–[6]). Their relevance for applications (such as automatic moves in a warehouse) is obvious.

Here we use the following (more restrictive) notion of a grid graph. Let $\mathcal{G}$ be the infinite graph embedded in $\mathbb{R}^2$, with $\mathbb{Z}^2$ as vertex set and with an edge between any two vertices at Euclidean distance 1. A *grid graph* is a connected subgraph of $\mathcal{G}$, whose intersection with any infinite path of vertex set $\{\cdot\} \times \mathbb{Z}$ or $\mathbb{Z} \times \{\cdot\}$ is connected or empty. For two graphs A, B, the graph $A - B$ denotes as usual, the graph obtained from A by deleting every vertex of B that is in A and any edge of A incident to such a vertex. The union of the boundaries of all unbounded domains of $\mathbb{R}^2$ determined by a grid graph G is called *boundary* of G. Any vertex of degree 2 in a grid graph is called a *corner*. For a connected graph G we also define its *width* $w(G)$ as the smallest number n for which there exists a path P of length n in G such that $G - P$ is disconnected and the diameter of each component of $G - P$ is at least n ($n = 0$ allowed). Note that not all connected graphs and not even all grid graphs possess a width. However, all finite grid graphs except P_1, P_2, and C_4—which for our purposes present little interest and will henceforth be excluded—do have a width. A *tour* in a finite graph G is a finite sequence of not necessarily distinct vertices

$$x_1, x_2, x_3, \cdots, x_n, x_{n+1} = x_1$$

such that all $\{x_i, x_{i+1}\}$'s are distinct edges of G ($1 \leqq i \leqq n$) and every edge of G has at least one endpoint in the sequence.

In this paper, we present sufficient conditions for a grid graph to be Hamiltonian. Also, we prove that all finite grid graphs of positive width have Hamiltonian line graphs.

2. Hamiltonian finite grid graphs. Let G be a finite grid graph of positive width. This restriction on the width is natural because every graph with vanishing width has connectivity 1 and is therefore non-Hamiltonian.

Let

$$W = \min\{x:(x,y)\in G\}, \qquad E = \max\{x:(x,y)\in G\},$$
$$S = \min\{y:(x,y)\in G\}, \qquad N = \max\{y:(x,y)\in G\}.$$

* Received by the editors May 3, 1989; accepted for publication (in revised form) July 2, 1991.

† Department of Computer Science, Hunter College of the City University of New York, 695 Park Avenue, New York, New York 10021. This author's research was supported in part by the City University of New York, PSC/CUNY Research Award Program.

‡ Fachbereich Mathematik, Universität Dortmund, Postfach 500500, 4600 Dortmund 50, Germany.

The corners of G, considered in their natural order on the boundary of G, are

$$(x_1, y_1), \cdots, (x_n, y_n),$$
$$(x_{n+1}, y_{n+1}), \cdots, (x_e, y_e),$$
$$(x_{e+1}, y_{e+1}), \cdots, (x_s, y_s),$$
$$(x_{s+1}, y_{s+1}), \cdots, (x_w, y_w),$$

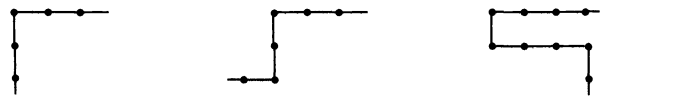
where $x_1 = x_w = W$, $y_n = y_{n+1} = N$, $x_e = x_{e+1} = E$, $y_s = y_{s+1} = S$. For all indices i, except $n + 1$ and $s + 1$, the above vertices (x_i, y_i) will be called *special* and receive *auxiliary coordinates* (ξ_i, η_i) as follows:

$$\xi_i = \begin{cases} x_{i+1} - x_i & \text{for } i \leqq n-1 \text{ or } e+1 \leqq i \leqq s-1, \\ x_i - x_{i-1} & \text{for } n+2 \leqq i \leqq e \text{ or } i \geqq s+2, \\ x_{i+1} - x_i + 1 & \text{for } i = n \text{ or } s; \end{cases}$$

$$\eta_i = \begin{cases} y_i + 1 & \text{for } i \leqq e, \\ y_i & \text{for } i \geqq e+1. \end{cases}$$

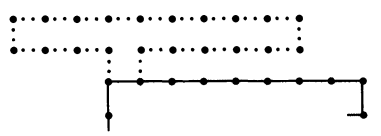
THEOREM 1. *Let G be a finite grid graph with $w(G) > 2$. If at every special vertex the product of its auxiliary coordinates is even, then G is Hamiltonian.*

Proof. First, we allow G to have smaller width, but still assume that $w(G) > 0$. Also, we assume that η is even at all special vertices, of auxiliary coordinates (ξ, η). Consider the height $h = N - S$. Then, clearly, h is odd. We prove that G has a Hamiltonian circuit containing the path on $y = N$ by induction on h. Indeed, the assertion is true if $h = 1$. Now suppose it is true for height h; we prove it for height $h + 2$. There are three possible situations at the left end of the path in G with $y = N$, shown below:



($y_{n-1} < N - 2$ or $n = 1$ and $y_1 - y_w > 1$) ($y_{n-1} = N - 2$) ($n = 1$ and $y_1 - y_w = 1$)

There are three analogous situations at the right end of the path. Consider now the subgraph H of G spanned by the vertices (x, y) with $y \leqq N - 2$. H satisfies the requirements of our induction hypotheses, as we can easily verify. So there is a Hamiltonian circuit in H, which can be transformed into one of G—here we also use $w(G) > 0$—as the following illustration shows:



Now, to prove the theorem in its general form, let $I = \{i : \eta_i \text{ is odd}\}$ and denote by Q the set of all points (x, y_i) with $i \in I$ and

$$\begin{aligned} x_i \leqq x < x_{i+1} &\quad \text{if } i < n, \\ x_n \leqq x \leqq x_{n+1} &\quad \text{if } i = n, \\ x_{i-1} < x \leqq x_i &\quad \text{if } n+2 \leqq i \leqq e, \\ x_{i+1} < x \leqq x_i &\quad \text{if } e < i < s, \\ x_{s+1} \leqq x \leqq x_s &\quad \text{if } i = s, \\ x_i \leqq x < x_{i-1} &\quad \text{if } i \geqq s+2. \end{aligned}$$

It is readily seen that $G - Q$ is a graph G' with $w(G') > 0$ and η even at all special vertices. Thus it is Hamiltonian, and (from the proof) it is clear that the constructed Hamiltonian circuit contains all of the paths with vertex sets, respectively,

$$\begin{aligned} &\{(x, y_i - 1) : x_i \leqq x < x_{i+1}\} && \text{if } i \in I \text{ and } i < n, \\ &\{(x, y_n - 1) : x_n \leqq x \leqq x_{n+1}\} && \text{if } n \in I, \\ &\{(x, y_i - 1\} : x_{i-1} < x \leqq x_i\} && \text{if } i \in I \text{ and } n+2 \leqq i \leqq e, \\ &\{(x, y_i + 1) : x_{i+1} < x \leqq x_i\} && \text{if } i \in I \text{ and } e < i < s, \\ &\{(x, y_s + 1) : x_{s+1} \leqq x \leqq x_s\} && \text{if } s \in I, \\ &\{(x, y_i + 1) : x_i \leqq x < x_{i-1}\} && \text{if } i \in I \text{ and } i \geqq s+2. \end{aligned}$$

Then it suffices to change each of them appropriately to obtain a Hamiltonian circuit in G. For instance, the path with consecutive vertices

$$(x_i, y_i - 1), (x_i + 1, y_i - 1), \cdots, (x_{i+1} - 1, y_i - 1),$$

where $i \in I$ and $i < n$, will be replaced by the path with consecutive vertices

$$(x_i, y_i - 1), (x_i, y_i), (x_i + 1, y_i), (x_i + 1, y_i - 1), (x_i + 2, y_i - 1), (x_i + 2, y_i), \cdots, (x_{i+1} - 1, y_i), (x_{i+1} - 1, y_i - 1).$$

This concludes the proof.

Theorem 1 possibly provides a useful sufficient condition for G to be Hamiltonian, but is unfortunately far from being a characterization. For an example of a graph that is Hamiltonian but does not satisfy the hypotheses of Theorem 1, see Fig. 1 in the following section.

3. Hamiltonian infinite grid graphs. We call an infinite graph Hamiltonian if it has a Hamiltonian two-way infinite path. There are several different types of infinite grid graphs. Some of them certainly do not contain any Hamiltonian graphs. This is the case, for instance, if the boundary has more than two components. We systematically consider all possible types in a subsequent paper; here we only mention two of them.

First, we investigate those infinite grid graphs with connected boundary for which the intersection with any infinite path of vertex set $\mathbb{Z} \times \{\cdot\}$ or $\{\cdot\} \times \mathbb{Z}$ is a one-way infinite path or empty. We may suppose without loss of generality that their vertex sets include $\mathbb{Z}_+ \times \mathbb{Z}_-$; we call these graphs *SE-grid graphs.*

Let G be an SE-grid graph. Let P be the boundary two-way infinite path of G. A (finite or infinite) subpath of P is called a *step* if its endpoints, and only its endpoints, have degree 4 in G. Such a step S must then have a corner (x_c, y_c) among its vertices.

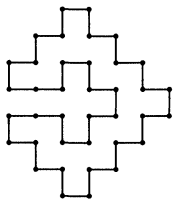


FIG. 1

Let

$$k_x = \text{card}\,\{x : (x, y_c) \text{ has degree } 3\},$$

$$k_y = \text{card}\,\{y : (x_c, y) \text{ has degree } 3\}.$$

Then we say that the step S at (x_c, y_c) is *of type* $(k_x + 1, k_y + 1)$. Clearly, a step of type (s, t) has length $s + t$. A step of type $(1, 1)$ is called a *unit step*.

Obviously, if G has only unit steps, it is not Hamiltonian. This is also the case if P only contains a one-way infinite path including just unit steps. We will prove the following theorem.

THEOREM 2. *Every SE-grid graph with just finitely many unit steps is Hamiltonian.*

Proof. We consider the SE-grid graph G and use all notation above. We may suppose, without loss of generality, that the natural ordering on P induces an increase of both Cartesian coordinates. If it has just one corner, then G is easily shown to be Hamiltonian. Let (v, w) be a corner of G and consider the following conditions:

(a) there is a step of type $(s, \aleph_0)$ at some corner $(x^*, y^*) \leqq (v, w)$;

(b) for infinitely many corners $(x, y) \leqq (v, w)$, the step at (x, y) is of type (s, t) with $t \geqq 2$;

(a′) there is a step of type $(\aleph_0, t)$ at some corner $(x^{**}, y^{**}) \geqq (v, w)$;

(b′) for infinitely many corners $(x, y) \geqq (v, w)$, the step at (x, y) is of type (s, t) with $s \geqq 2$.

The Hamiltonian path will be composed of a one-way infinite path A and infinitely many finite paths $B_1, C_1, B_2, C_2, \cdots$. To describe the Hamiltonian path, let us say that a path in G is *close* to another if they are disjoint but no point of the first is at Euclidean distance more than $\sqrt{2}$ from the second.

Construction of A. We simply take A to be a one-way infinite subpath of P ending (supposing that we come from infinity) at a point (x_0, y_0) chosen as follows:

If (a) is satisfied, let $(x_0, y_0) = (x^*, y^*)$. If not, let (x_0, y_0) be a corner of type (s_0, t_0) with $t_0 \geqq 2$ lying below all unit steps of G if (b) is satisfied, and a corner of type $(s_0, 1)$ lying below all unit steps and below all corners of type (s, t) with $t \geqq 2$ if (b) is not satisfied.

Construction of B_n. B_n begins at (x_{n-1}, y_{n-1}), goes through $(x_{n-1}, y_{n-1} - 1)$, then goes close to $A \cup \bigcup_{i<n} (B_i \cup C_i)$ up to a point (x'_n, y'_n) such that if (a′) is satisfied, $y'_n + 1 = y^{**}$; if not, $(x'_n - 1, y'_n + 1)$ is a corner above all unit steps of G, of type (s'_n, t'_n) with $s'_n \geqq 2$ if (b′) is satisfied, and $s'_n = 1$ if (b′) is not satisfied.

Construction of C_n. C_n begins at (x'_n, y'_n), goes through $(x'_n + 1, y'_n)$, then turns down and goes close to B_n until a point (x_{n-1}, y) is reached. Then if (a) is true, C_n ends at $(x_n, y_n) = (x_{n-1}, y) = (x_{n-1}, y_{n-1} - 2)$. Otherwise, C_n again follows P downward until the next corner (x_n, y_n) of type (s_n, t_n) with $t_n \geqq 2$ if (b) is fulfilled, and $t_n = 1$ if (b) is not fulfilled.

The proof that the mechanism works presents an interest only if (a) and (a′) are not satisfied, and essentially differs between having (b) (or (b′)) satisfied or unsatisfied.

While, in the case where (b) and (b′) are satisfied, all B_n and C_n are coordinate-monotone, this is not the case when (b) or (b′) is not satisfied. It works close to the previous path, essentially because G contains, together with a vertex (x, y), all vertices $(x + z, y - z)$, $z \in \mathbb{N}$. This is enough when (b) and (b′) are fulfilled. If not, the algorithm still works, essentially because, in that situation, for some corner (x_-, y_-) and any corner $(x, y) \leqq (x_-, y_-)$, $(x - z, y - z) \in G$ $(z \in \mathbb{N})$ if (b) is not satisfied, and for some corner (x_+, y_+) and any corner $(x, y) \geqq (x_+, y_+)$, $(x + z, y + z) \in G$ $(z \in \mathbb{N})$ if (b′) is not satisfied. Hence the theorem is proved.

Although Theorem 2 provides an excitingly weak sufficient condition for an *SE*-grid graph to be Hamiltonian, it fails, however, to be a characterization.

Now we mention a sufficient condition for a grid graph G to be Hamiltonian, in the case where its intersection with any infinite path of vertex set $\mathbb{Z} \times \{\cdot\}$ is a finite path or empty and its intersection with any path of vertex set $\{\cdot\} \times \mathbb{Z}$ is a one-way infinite path or empty. In such a case, G is called an *N-grid graph*. The easy proof is left to the reader.

THEOREM 3. *If, for every corner (x, y) of an N-grid graph G, y is even, then G is Hamiltonian.*

In §§ 1 and 2 we made little progress toward a satisfactory answer to the following problem.

Problem. Provide a characterization of Hamiltonian grid graphs.

Among the cases treated here, we consider that of an *SE*-grid graph as the most hopeful, since our Theorem 2 already comes rather close to being a characterization.

4. Hamiltonian line graphs of grid graphs. Clearly, a finite grid graph G is Eulerian if and only if it has no vertices of degrees 1 or 3; see Fig. 2. The line-graph $L(G)$ of G is Eulerian if and only if all vertex degrees in G are of the same parity, which also means that vertices of degree 1 or 3 must fail, with the remarkable exception illustrated in Fig. 3. In short, G is Eulerian precisely when it is as shown in Fig. 2, and $L(G)$ is Eulerian if and only if G is as shown in Fig. 2 or 3. Much more often, $L(G)$ is Hamiltonian, as already demonstrated by the following corollary.

COROLLARY TO THEOREM 1. *If G is a finite grid graph with $w(G) > 2$, then $L(G)$ is Hamiltonian.*

Proof. We use the terminology of § 1. Let k be the number of all special vertices of G having at least one of its auxiliary coordinates 1. We prove by induction on k the following assertion: G has a tour visiting all special vertices (ξ_i, η_i) with $\xi_i \eta_i$ even.

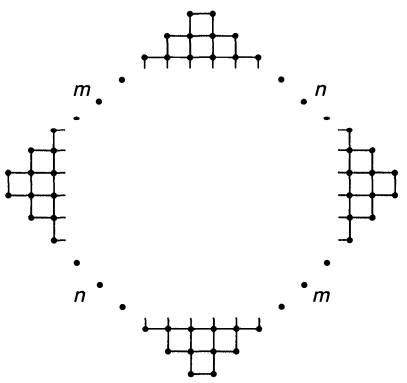


FIG. 2

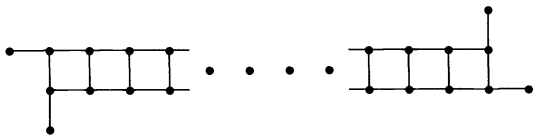


FIG. 3

For $k = 0$, delete all special vertices (ξ_i, η_i) with $\xi_i \eta_i$ odd. The resulting graph has, by Theorem 1, a Hamiltonian path, which is a tour in G visiting all special vertices of G that were not deleted.

Suppose now that the assertion is true for at most k special vertices with at least one auxiliary coordinate 1; we prove it for $k + 1$ such vertices.

Case 1. There exists a special vertex (x_i, y_i) with auxiliary coordinates $(1, 1)$. Suppose that $i \leqq n$. Delete (x_i, y_i). If the previous corner (x_{i-1}, y_{i-1}) had ξ_{i-1} even and η_{i-1} odd, it also must be deleted. If, moreover, $\eta_{i-1} = 1$, we must look at the corner (x_{i-2}, y_{i-2}) and delete it, also, in the case where ξ_{i-2} is even and η_{i-2} odd, and so forth. If the corner (x_{i+1}, y_{i+1}) had ξ_{i+1} odd and η_{i+1} even, delete it. If, moreover, $\xi_{i+1} = 1$, we look at (x_{i+2}, y_{i+2}), and delete it, provided that ξ_{i+2} is odd and η_{i+2} even, and so forth. If $n + 2 \leqq i \leqq e$ or $e + 1 \leqq i \leqq s$ or $i \geqq s + 2$, proceed similarly.

Case 2. There is a special vertex (x_i, y_i) with auxiliary coordinates $(\xi_i, 1)$ $(\xi_i \neq 1)$. If $i \leqq n$, delete that vertex. If the previous corner (x_{i-1}, y_{i-1}) had ξ_{i-1} even and η_{i-1} odd, delete it, also. If, moreover, $\eta_{i-1} = 1$, look at (x_{i-2}, y_{i-2}) and delete it if ξ_{i-2} is even and η_{i-2} odd, and so forth. Proceed similarly if $n + 2 \leqq i \leqq e$ or $e + 1 \leqq i \leqq s$ or $i \geqq s + 2$, instead of $i \leqq n$.

Case 3. There is a special vertex (x_i, y_i) with auxiliary coordinates $(1, \eta_i)$ $(\eta_i \neq 1)$ (analogous to Case 2).

After the deletions, there will be at most k special vertices with at least one auxiliary coordinate 1. The induction hypothesis guarantees the existence in the graph obtained after deletions of a tour T containing all special vertices whose (new) auxiliary coordinates have even products. This ensures that T is a tour of the same kind in the original graph, also.

Now the following known result finishes the proof.

LEMMA (see [1]). *For any finite graph G, $L(G)$ is Hamiltonian if and only if G has a tour.*

The restriction $w(G) > 2$ is not natural, merely imposed by the proof method. This is shown by the following strengthening, which has, however, a less elegant proof.

THEOREM 4. *For every finite grid graph G of positive width, $L(G)$ is Hamiltonian.*

Proof. Let $P_k(G)$ be the path of all vertices (x, k) in G. We prove that G has a tour T such that $P_N(G) - T$ consists (if not empty) of isolated vertices. We proceed again by induction on the height $h = N - S$. Indeed, the assertion is true for $h = 1$. We suppose it to be true for height h and prove it for height $h + 1$.

Consider the subgraph H of the grid graph G of height $h + 1$, spanned by the vertices (x, y) with $y \leqq N - 1$. If H has vertices of degree 1, they must lie on the line $y = N - 1$, and will be deleted. If the resulting graph has vertices of degree 1, delete them also. Repeat the procedure until the remaining graph H' has no vertices of degree 1. Then H' is a grid graph of height h and therefore has a tour T' such that $P_{N-1}(H) - T'$ consists of isolated vertices, at most.

Let us consider a connected component P' of $P_{N-1}(H') \cap T'$. Observe that P' must exist because $w(G) > 0$, and that P' must be a path. Let

$$P'' = \{x : (x, N-1) \in P' \text{ and } (x, N) \in G\}.$$

If $P'' \neq \varnothing$ and $u = \min P''$ is different from $v = \max P''$, then we replace the subpath of P' spanned by P'' by the path ex (P') spanned by

$$(u,N-1),(u,N),(u+1,N),(u+1,N-1),(u+2,N-1),(u+2,N),$$
$$(u+3,N),(u+3,N-1),\cdots,(v-1,N-1),(v-1,N),(v,N),(v,N-1)$$

for $v-u$ odd, or by

$$(u,N-1),(u,N),(u+1,N),(u+1,N-1),(u+2,N-1),(u+3,N-1),(u+3,N),$$
$$(u+4,N),(u+4,N-1),\cdots,(v-1,N-1),(v-1,N),(v,N),(v,N-1)$$

for $v-u$ even. Let

$$Q=\bigcup_{P'} \text{ex}\,(P').$$

Now, if $(x_n, y_n) \notin Q$ (remember that $y_n = N$), we extend Q by replacing its edge $\{(x, N-1), (x, N)\}$ with minimal x by the path spanned by

$$(x,N-1),(x-1,N-1),\cdots,(x_n,N-1),(x_n,N),(x_n+1,N),\cdots,(x,N).$$

Similarly, if $(x_{n+1}, y_{n+1}) \notin Q$ (remember also that $y_{n+1} = N$), we extend Q by replacing its edge $\{(x, N-1), (x, N)\}$ with maximal x by the path spanned by

$$(x,N-1),(x+1,N-1),\cdots,(x_{n+1},N-1),(x_{n+1},N),(x_{x+1}-1,N),\cdots,(x,N).$$

Suppose now that $(x_0, N-1) \in P_{N-1}(H') - T'$. Then, since

$$(x_0,N-1),(x_0,N)\notin Q,$$

we replace for every such x_0 the edge $\{(x_0+1, N-1), (x_0+1, N)\}$ by the path spanned by

$$(x_0+1,N-1),(x_0,N-1),(x_0,N),(x_0+1,N).$$

Thus we transformed T' into a tour of G. By the lemma, $L(G)$ is Hamiltonian.

Remark. Concerning the width, Theorem 4 is the best possible, because finite grid graphs of vanishing width may have non-Hamiltonian line graphs; P_3 is a suitable example.

Acknowledgment. Thanks are due to the referee for his comments.

REFERENCES

[1] F. HARARY AND C. ST. J. A. NASH-WILLIAMS, *On Eulerian and Hamiltonian graphs and line graphs*, Canad. Math. Bull., 8 (1965), pp. 701–709.

[2] A. ITAI, C. H. PAPADIMITRIOU, AND J. L. SZWARCFITER, *Hamilton paths in grid graphs*, SIAM J. Comput., 11 (1982), pp. 676–686.

[3] S. M. HEDETNIEMI, S. T. HEDETNIEMI, AND P. J. SLATER, *Which grids are Hamiltonian?*, Congr. Numer., 29 (1980), pp. 511–524.

[4] C. ST. J. A. NASH-WILLIAMS, *Decomposition of the n-dimensional lattice graph into Hamiltonian lines*, Proc. Edinburgh Math. Soc., 12 (1960/1961), pp. 123–131.

[5] G. RINGEL, *Über drei kombinatorische Probleme am n-dimensionalen Würfel und Würfelgitter*, Abh. Math. Sem. Univ. Hamburg, 20 (1955), pp. 10–19.

[6] E. VAZSONYI, *Über Gitterpunkte des mehrdimensionalen Raumes*, Acta Lit. Sci. Szeged Sec. Sci. Math., 9 (1939), pp. 163–173.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 571–585, November 1992
© 1992 Society for Industrial and Applied Mathematics
013

ROOTS OF THE RELIABILITY POLYNOMIAL*

JASON I. BROWN† AND CHARLES J. COLBOURN‡

Abstract. The *reliability* of a graph G is the probability that G is connected, given that edges are independently operational with probability p. This is known to be a polynomial in p, and the location of the roots of these functions is discussed. In particular, it is conjectured that the roots of the reliability polynomial of any connected graph lie in the disc $|z-1| \leq 1$, and evidence for this conjecture is provided. It is shown that all real roots lie in $\{0\} \cup (1,2]$ and that every graph has a subdivision for which the roots of the reliability polynomial lie in the conjectured disc.

Key words. reliability polynomial, graphs, matroid, roots, H-vector, Enestrom–Kakeya theorem

AMS(MOS) subject classifications. 05C30, 30C15, 68M15

1. Introduction.

Let G be a connected undirected graph (with possibly multiple edges and loops) of order n and size m (i.e., $|V(G)| = n$ and $|E(G)| = m$). The *reliability* of G, $\mathrm{Rel}(G,p)$, is the probability that G is connected, where each edge of G is independently operational with probability p. It is easy to see that $\mathrm{Rel}(G,p)$ is always a polynomial in p (the *reliability polynomial* of G). As examples, the reliability of any tree of order n is p^{n-1}, and the reliability of a cycle of order n is $np^{n-1}-(n-1)p^n$. Much work has been done on calculating and approximating $\mathrm{Rel}(G,p)$(cf. [12]), and a number of sequences associated with the polynomial have been conjectured to be unimodal (see [8]–[10], [12]). In providing evidence for the latter [10], the problem of locating the roots of the reliability polynomial arose. It may be surprising that the roots of reliability polynomials do not appear to be spread out in the complex plane. (Figure 1 shows the location of the roots of the reliability polynomials of all 143 simple graphs of order at most 6.) In fact, we propose here that, for any connected graph G, the roots of $\mathrm{Rel}(G,p)$ lie in the unit disc centered at 1 in the complex plane.

CONJECTURE 1.1. *The roots of the reliability polynomial of a connected graph G lie in $|z-1| \leq 1$ in the complex plane.*

Such a result would be the best possible, as, for example, the graph of order 2 with μ parallel edges has reliability polynomial $1-(1-p)^\mu$ and hence has all its roots on $|z-1| = 1$. We show that all real roots of reliability polynomials lie in $\{0\} \cup (1,2]$ and that every graph has a subdivision for which the roots of the reliability polynomial lie in the prescribed disc. It suffices to consider the conjecture only for 2-connected graphs, as it is easy to see that the reliability of a graph is the product of the reliabilities of its blocks [8].

It has been conjectured that for any graph G, the S, F, H, N, and C sequences [12] are *unimodal* (i.e., first nondecreasing, then nonincreasing); this conjecture includes the restriction to cographic matroids of Welsh's [25] well-known conjecture for unimodality of the the independence numbers for matroids. In [10] strong evidence was provided for the conjectures, and the arguments centered on determining sufficient conditions for the reliability polynomial to have only real roots. In fact, as Gernert [15] has observed, if a polynomial $f(x)$ has positive coefficients and every root z

* Received by the editors March 25, 1991; accepted for publication (in revised form) March 12, 1992. This work was supported by the Natural Science and Engineering Council of Canada.

† Department of Mathematics and Statistics, York University, North York, Ontario, Canada M3J 1P3.

‡ Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1.

satisfies $\mathrm{Re}z \leq 0$ and $3(\mathrm{Re}z)^2 \geq (\mathrm{Im}z)^2$, then the coefficients of f form a unimodal sequence. This suggests that locating the roots of the reliability polynomial would be useful in studying the unimodal conjectures.

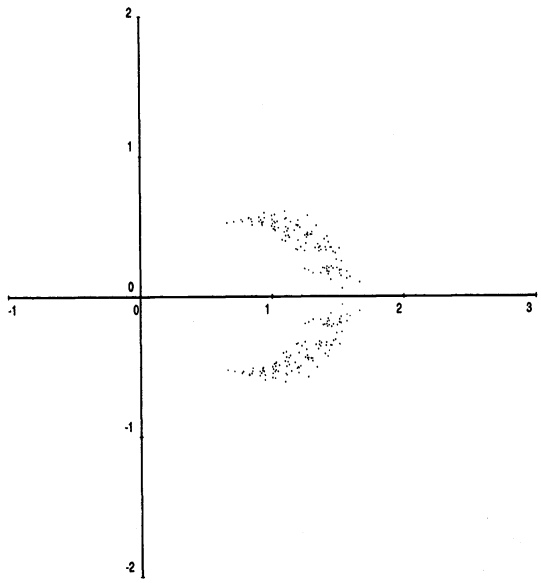


FIG. 1

An investigation of the location of the roots of another graph polynomial, the *chromatic polynomial*, has attracted some attention, but much of the work remains as conjecture. Berman and Tutte [4] noted that certain families of chromatic polynomials had their roots clustered in a cardioid-like curve. Farrell [14] plotted the roots of all chromatic polynomials of graphs of order at most 8 and observed that there are certain complex numbers that appear often as roots and that each is surrounded by a small neighbourhood that is free of such roots. Recently, Thier (see [15]) has found bounded regions in the complex plane containing the roots of all chromatic polynomials of graphs of order n and size m. Some other results can be found in [5], and [20], [21] survey roots of chromatic polynomials.

2. Reliability and H-vectors. There are a number of expansions for the reliability polynomial (cf. [12]), two of which are

$$\mathrm{Rel}(G,p) = \sum_{i=0}^{m-n+1} F_i p^{m-i}(1-p)^i \qquad \text{(F-form)}$$

$$= p^{n-1} \sum_{i=0}^{m-n+1} H_i(1-p)^i \qquad \text{(H-form)}.$$

Here $\langle F_i \rangle$ is the *face vector* (or F-*vector*) of $\mathcal{F}$, the *cographic matroid* of G (the cographic matroid of a connected graph G has underlying set $E(G)$ with $S \subseteq E(G)$ independent if and only if $G - S$ is connected). $\langle H_i \rangle$ is the H-*vector* of $\mathcal{F}$ [6], [23] and

is defined by

$$H_i = \sum_{j=0}^{i} (-1)^{i-j} \binom{m-n+1-j}{m-n+1-i} F_j, \qquad (0 \le i \le m-n+1).$$

The F- and H-vectors of a graph refer throughout to the corresponding vectors of the associated cographic matroid.

We find the H-form of the reliability polynomial the most useful here, for Conjecture 1.1 is equivalent to stating that, for any connected graph G with reliability $p^{n-1} \sum H_i (1-p)^i$, the roots of $h(z) \equiv \sum H_i z^i$ lie in the unit disc $|z| \le 1$. The H-vector can be interpreted in the following sense. A *complex* on a finite set X is a nonempty set of subsets of X (called *simplices*) that is closed under containment. A complex is *pure* if all bases (i.e., maximal simplices) are of the same cardinality (the *dimension* of such a complex is the cardinality of any facet). For any sets L and U with $L \subseteq U$, the interval $[L, U]$ is $\{S : L \subseteq S \subseteq U\}$. A pure complex $\mathcal{C}$ is *partitionable* if its simplices can be partitioned into intervals

$$[L_1, U_1], \ldots, [L_l, U_l],$$

where each U_i is a facet of $\mathcal{C}$. In such a case, we can derive that $H_i = |\{j : |L_j| = i\}|$. $\langle H_i \rangle$ is known as the H-vector of the partitionable complex. Every matroid is partitionable [24] (see also [12]). It follows that $\langle H_i \rangle$ is a sequence of nonnegative integers. We can also view H-vectors another way. A *multicomplex* (or *generalized complex*) is a collection of multisets closed under multiset containment; its F-vector is defined analogously to the definition for complexes. We can also consider a multicomplex as a set of monomials that is closed under division. The important fact (see [12]) is that every H-vector is the F-vector of a multicomplex.

For later, we must prove the following result showing that H-vectors of matroids have no internal zeros (we follow [25] for most matroidal terminology). If x is an element of a matroid M on ground set S and $T = S - \{x\}$, then $M|T$ denotes the (*restriction*) matroid on ground set T whose independent sets are those of M contained in T, and $M.T$ denotes the (*contraction*) matroid on ground set T whose independent sets are those sets $I \subset T$ such that $I \cup \{x\}$ is independent in M.

LEMMA 2.1. *Let M be a matroid of rank d on a set S with* H*-vector $\langle H_0^M, \ldots, H_d^M \rangle$. Then $H_d^M = 0$ if and only if M has a coloop (i.e., an element in every base) and $\langle H_0^M, \ldots, H_d^M \rangle$ has no internal zeros.*

Proof. If e is a coloop of M, consider a partition $[L_1, U_1], \ldots, [L_k, U_k]$ of $M.T = M|T$, where $T = S - \{e\}$. As e belongs to every base of M, we see that $[L_1, U_1 \cup \{e\}], \ldots, [L_k, U_k \cup \{e\}]$ is an interval partition of M. Then, for all i,

$$H_i^{M.T} = |\{j : |L_j| = i\}| = H_i^M.$$

In particular, as $H_d^{M.T} = 0$ (since $M.T$ has rank $d-1$), we see that $H_d^M = 0$. It also follows by induction on d that the H-vector of $M.T$, and hence the H-vector of M, has no internal zeros.

On the other hand, assume that M has no coloop. If $d = 0$ or $\nu = |S| = 0$, then $H_0 = 1$; so the results are true. Thus assume that $d > 0$ and $\nu > 0$. Pick any e that is not a loop of M and set $T = S - \{e\}$. As e is not a coloop, $M.T$ and $M|T$ have ranks $d-1$ and d, respectively. As in [9], we define for any pure complex N of dimension r on a ground set of cardinality μ a two variable polynomial as follows:

$$\alpha(N, p, q) = \sum_{Y \in N} q^{|Y|} p^{\mu - |Y|}.$$

As noted there, if N is partitionable, then

$$\alpha(N,p,q) = \sum_{j=0}^{r} H_j q^j p^{\mu-r}(p+q)^{r-j},$$

where $\langle H_j \rangle$ is the H-vector of N. We use Theorem 2.2 of [9] (with $q = 1-p$) to calculate the H-vector as follows:

$$\begin{aligned}\alpha(M,p,1-p) &= (1-p)\cdot\alpha(M.T,p,1-p) + p\cdot\alpha(M|T,p,1-p)\\ &= p^{\nu-1-(d-1)}\sum_{i=0}^{d-1} H'_i(1-p)^{i+1} + p\cdot p^{\nu-1-d}\sum_{i=0}^{d} H''_i(1-p)^i\\ &= p^{\nu-d}(1+\sum_{i=1}^{d}(H'_{i-1}+H''_i)(1-p)^i).\end{aligned}$$

It follows that

$$H_i = \begin{cases} 1 & \text{if } i = 0,\\ H'_{i-1} + H''_i & \text{if } i = 1,\dots,d.\end{cases} \tag{1}$$

By induction, we have that the H-vectors of both $M.T$ and $M|T$ have no internal zeros, so from (1) (and the fact that the sequences consist of nonnegative integers) we see that M also has no internal zeros. If $M.T$ or $M|T$ has no coloops, then we see by induction on d and ν that either $H'_{d-1} > 0$ or $H''_d > 0$; so, as each term is nonegative, it follows from above that $H_d > 0$. The only case remaining is that $M.T$ has a coloop f and $M|T$ has a coloop g. Let $b_1,\dots,b_k$ be the bases of $M.T$ and let $c_1,\dots,c_r$ be the bases of $M|T$. Consider any $c_i = \{g\}\cup A$, where $g \notin A$ and pick a b_j such that $g \notin b_j$ (if $g \in b_j$ for every j, then g would be a coloop of M, a contradiction). Now, by the exchange property of matroids, for each $i = 1,\dots,k$ there is an $x \in b_j \cup \{e\}$ such that $A\cup\{x\}$ is a base of M. As $x \neq g$, $A\cup\{x\} = b_l \cup \{e\}$ for some l; as $e \notin A$, we must have $x = e$, and so $f \in A$. Thus f is a coloop of $M|T$ as well, so it is a coloop of M, a contradiction. Thus we conclude that this case cannot occur, so indeed $H_d > 0$. □

Note that, if x is a coloop of M, then $y \neq x$ is a coloop of M if and only if y is a coloop of $M.(S-\{x\})$. From this we can conclude the following.

COROLLARY 2.2. *If M has exactly r coloops, then the last r terms in $\langle H_0,\dots,H_d\rangle$ are zero (and $H_i > 0$ for $i = 0,\dots,d-r$).*

The fact that the H-vector has no internal zeros also follows from the fact that they form the F-vector of a multicomplex.

3. The real roots of reliability polynomials. If G is connected, then it is easy to see (from the H-form) that $\mathrm{Rel}(G,p)$ has zero as a root of multiplicity $n-1$, has sign $(-1)^{n-1}$ on $(-\infty,0)$, and is positive in $(0,1]$. Thus all real nonzero roots of $\mathrm{Rel}(G,p)$ lie in $(1,\infty)$, and, by considering the reliability of cycles, we see that 1 is a limit point of the roots of reliability polynomials. In concurrence with Conjecture 1.1, we show that the real roots of the reliability polynomial always lie in the unit disc centered at 1 and, in fact, lie in $\{0\}\cup(1,2]$.

Let G be a connected, loopless graph with n vertices and m edges, and hence with dimension (or "cyclomatic number") $d = m-n+1$. From Corollary 2.2, it follows that $(H_0,\dots,H_d)$ are the nonzero terms of the H-vector of G. We take $H_i = 0$ for $i < 0$ or $i > d$.

THEOREM 3.1. *For any* H-*vector* $(H_0, \ldots, H_d)$ *of a connected, loopless multigraph, any real number* $b \geq 1$, *and any integer* j,

$$\sum_{i=0}^{j} (-b)^i H_i \begin{cases} \geq 0 & \text{if } \min(j,d) \text{ is even,} \\ \leq 0 & \text{if } \min(j,d) \text{ is odd.} \end{cases}$$

Equality holds only if $b = 1$ *or* $j < 0$.

Proof. The proof proceeds by induction on the dimension d of G and the number of vertices n. The statement is trivial if the dimension is zero, since the H-vector is then $(H_0 = 1)$. If $n = 1$, the dimension is necessarily zero. So suppose that the statement of the theorem holds for all integer values of j for all connected, loopless multigraphs of dimension less than d and for all those of dimension d having fewer than n vertices. Suppose that G is a connected, loopless multigraph having n vertices and dimension d, with H-vector $(H_0, \ldots, H_d)$.

Fix j to be some integer. Let e be an edge in a bundle of t parallel edges in G. Define G^+ to be the connected, loopless multigraph obtained from G by contracting e and removing the $t-1$ loops that result; let the H-vector of G^+ be $(H_0^+, \ldots, H_{d-t+1}^+)$ (the dimension of G^+ is easily computed to be $d-t+1$). For $0 \leq r \leq t$, let G^{-r} be the loopless multigraph obtained from G by deleting r of the t edges parallel to e (including e itself); let the H-vector of G^{-r} be $(H_0^{-r}, \ldots, H_{d-r}^{-r})$. The dimension of G^{-r} is $d-r$ if G^{-r} is connected.

We use the identity

$$H_i = H_{i-r}^{-r} + \sum_{x=0}^{r-1} H_{i-x}^{+} ,$$

which holds for $0 \leq r \leq t$ and all i (by repeatedly applying deletion and contraction to each edge, in turn, in the bundle of t edges).

We consider the resulting identity

$$\sum_{i=0}^{j} (-b)^i H_i = \sum_{i=0}^{j} (-b)^i H_{i-r}^{-r} + \sum_{i=0}^{j} (-b)^i \sum_{x=0}^{r-1} H_{i-x}^{+}$$

and show, for suitable selections of r, that the two sums on the right-hand side agree in sign with $(-1)^{\min(j,d)}$ or 0.

We treat two cases. First, suppose that either (i) t is odd and $j \leq d-t+1$ or $j \geq d$, or (ii) $d-t+1 < j < d$ and $j \equiv d-t+1 \pmod 2$. Then, taking $r = 1$ in the identity gives

$$(*) \qquad \sum_{i=0}^{j} (-b)^i H_i = (-b) \sum_{i=0}^{j-1} (-b)^i H_i^{-1} + \sum_{i=0}^{j} (-b)^i H_i^{+}.$$

The second sum on the right side has sign $(-1)^{\min(j,d-t+1)}$ or zero by induction, with sign zero only if $b = 1$ or $j < 0$; we also see that, in the former case, the sign is identical to that of $(-1)^{\min(j,d)}$ by considering the value of j and cases (i) and (ii). If G^{-1} is connected, the first sum has sign $(-1)^{\min(j-1,d-1)}$ or zero by induction, and hence, upon multiplication by $-b$, has sign $(-1)^{\min(j,d)}$ or zero as required. If G^{-1} is disconnected, the second sum is identically zero. As the sums on the right side of $(*)$ do not have opposite signs, we see that if the left side is zero, so are each sum on the

right. Thus we conclude that $(*)$ has sign $(-1)^{\min(j,d)}$ or zero, and the latter implies that $b = 1$ or $j < 0$.

On the other hand, suppose that either (iii) t is even and $j \leq d - t + 1$ or $j \geq d$, or (iv) $d - t + 1 < j < d$ (whence $t > 1$) and $j \not\equiv d - t + 1 \pmod 2$. Then taking $r = 2$ in the identity yields

$$(**) \quad \begin{aligned} \textstyle\sum_{i=0}^{j}(-b)^i H_i &= \textstyle\sum_{i=0}^{j}(-b)^i(H_i^+ + H_{i-1}^+) + (-b)^2 \sum_{i=0}^{j-2}(-b)^i H_i^{-2} \\ &= (-b)^j H_j^+ + (1-b)\textstyle\sum_{i=0}^{j-1}(-b)^i H_i^+ + (-b)^2 \sum_{i=0}^{j-2}(-b)^i H_i^{-2}. \end{aligned}$$

Now $(-b)^j H_j^+$ has sign zero if and only if $j < 0$ or $j > d - t + 1$; otherwise, it has sign $(-1)^j$. By induction, the first sum has sign $(-1)^{\min(j-1,d-t+1)}$ or zero, with zero occurring only if $b = 1$ or $j - 1 < 0$. Similarly, $1 - b$ has sign -1 unless $b = 1$, in which case the sign is zero. Thus, if $b > 1$ and $j > 0$, the sign of $(1-b)\sum_{i=0}^{j-1}(-b)^i H_i^+$ is $(-1)^{\min(j,d-t+2)}$, which (by considering the value of j once again) is the same as the sign of $(-1)^{\min(j,d)}$, as required. When $j = 0$ and $b > 1$, the sum simplifies to $(-b)^0 H_0^+$, which is strictly greater than zero. Now consider the second sum in the right side of $(**)$. If G^{-2} is disconnected, the sum is identically zero. If G^{-2} is connected, by induction the sum has sign $(-1)^{\min(j-2,d-2)}$ or zero; upon multiplication by $(-b)^2$, the sign is $(-1)^{\min(j,d)}$ or zero, as required. Again, we conclude that $(**)$ has sign $(-1)^{\min(j,d)}$ or zero, and the latter implies that $b = 1$ or $j < 0$. □

COROLLARY 3.2. *The reliability polynomial of a connected, loopless multigraph has real roots only in* $\{0\} \cup (1, 2]$.

Proof. Consider the H-form of reliability polynomial, below:

$$\mathrm{Rel}(G, p) = p^{n-1} \sum_{i=0}^{d} H_i (1-p)^i.$$

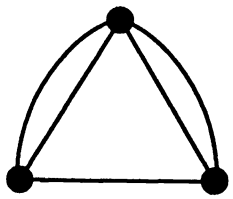
There is a root of multiplicity $n - 1$ at $q = 1$. All other roots are roots of $h(q) \equiv \sum_{i=0}^{d} H_i q^i$ (here $q = 1 - p$). If $q \geq 0$ is real, the sum is strictly positive (it is at least $H_0 = 1$). Thus all remaining real roots are negative. Now suppose that $q = -b$ is a real root. Then $\sum_{i=0}^{d}(-b)^i H_i = 0$, and hence, by Theorem 3.1, $b \leq 1$. Therefore such a q lies in $[-1, 0)$, and this is equivalent to $p \in (1, 2]$. □

One of the best techniques for estimating reliability is the Ball–Provan method [2] (see also [12]), which relies on Stanley's bounds (see [12]) for the terms of successive terms in the H-vectors of certain pure complexes (which include matroids). These bounds are the only ones known for terms of the H-vectors of shellable complexes. Theorem 3.1 establishes that the H-vectors arising in reliability problems are much more constrained than those of shellable complexes in general. Hence the above theorem underlies an improvement to the Ball–Proven bounds for reliability. In fact, the inequalities given in Theorem 3.1 can be used in conjunction with Stanley's constraints as well as with other bounds on individual coefficients [11] to tighten the bounds on reliability further.

4. Complex roots of the reliability polynomial. The *Eneström–Kakeya theorem* [13], [18], [17] (see also [19, eq. (22), p. 107] and [3, eq. (6b), p. 180]) states that, if a polynomial $g(z) = \sum_{i=0}^{d} b_i z^i$ has $0 < b_0 \leq b_1 \leq \cdots \leq b_d$, then all the roots of g lie in $|z| \leq 1$. By considering the H-form of the reliability polynomial, we derive the following sufficient condition for the conjecture to hold for a particular graph G.

PROPOSITION 4.1. *If the* H*-vector of a connected graph* G *is nondecreasing, then the roots of the reliability polynomial all lie in* $|z - 1| \leq 1$.

FIG. 2



(1,2,3,2)

FIG. 3

Since the reliability of a graph is the product of the reliabilities of its blocks [8], the proposition implies that, if every block of a graph G has its H-vector nondecreasing, the roots of $\mathrm{Rel}(G,p)$ lie in the desired disc (although the H-vector of G itself may not be nondecreasing; for example, the graph G in Fig. 2 has H-vector $(1,2,1)$, but has its roots in $|z-1| \leq 1$ as each of its two blocks has H-vector $(1,1)$). Thus it suffices to only consider 2-connected graphs G.

Examples of graphs for which the H-vectors are nondecreasing include all 2-connected graphs of order n and size at most $n+1$. Any such graph G is either a tree (of order 1 or 2), cycle, or theta graph, and the H-vectors for each are (1), $(1,n-1)$, and $(1,a+b+c-2,ab+ac+bc-a-b-c+1)$ (where the theta graph has two vertices joined by paths of lengths a, b, and $c=n+1-a-b$). For the first two cases, the H-vectors are clearly nondecreasing. For the third, we can check (via elementary calculus) that it is nondecreasing as well for all a, b, and c. It follows that Conjecture 1.1 holds for any connected graph of order n and size at most $n+1$.

On the other hand, there are many 2-connected graphs for which the H-vector fails to be nondecreasing. The smallest 2-connected graph whose H-vector is not nondecreasing is shown in Fig. 3, and the only simple graphs of order at most 5 whose H-vectors are not nondecreasing are shown in Fig. 4 (the corresponding H-vectors are listed); we have verified though, that, for each of these graphs, the reliability polynomial has all its roots in the unit disc centered at 1. Also, we now show that every complete graph of order at least 5 has its H-vector not nondecreasing (and we do not know whether Conjecture 1.1 holds for all complete graphs).

PROPOSITION 4.2. *The* H*-vector of* K_n *is nondecreasing if and only if* $n \leq 4$.

Proof. The H-vectors of K_1, K_2, K_3, K_4, and K_5 are (1), (1), $(1,2)$, $(1,3,6,6)$, and $(1,4,10,20,30,36,24)$; so we may assume that $n \geq 6$. From the H-form of the reliability polynomial of a connected loopless graph G of order n and size $m \geq n$,

$$\mathrm{Rel}(G,p) = p^{n-1} \sum_{i=0}^{m-n+1} H_i(1-p)^i,$$

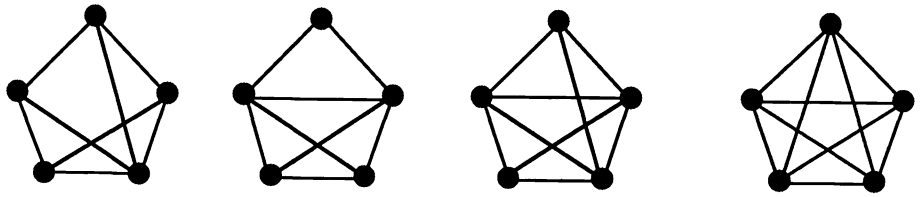


FIG. 4

the coefficients of p^m and p^{m-1} are, respectively, $(-1)^d H_d$ and $(-1)^{d-1}(H_{d-1} + dH_d)$ (where $d = m - n + 1$). The reliability of complete graphs can be calculated from a recursive formula of Gilbert [16] (see also [12, p. 34]) as follows:

$$1 = \sum_{j=1}^{n} \binom{n-1}{j-1} \mathrm{Rel}(K_j, p) q^{j(n-j)}.$$

Let $H_{\text{last}}(K_n)$ and $H_{\text{second last}}(K_n)$ denote the last and second last nonzero terms in the H-vector of K_n, respectively. By comparing the coefficients of $p^{\binom{n}{2}}$, we derive from Gilbert's formula that

$$H_{\text{last}}(K_n) = (n-1) H_{\text{last}}(K_{n-1})$$

for $n \geq 2$, and hence

$$H_{\text{last}}(K_n) = (n-1)!$$

for all $n \geq 1$ (this fact has also been proved in [7] in the context of reliability domination). Moreover, by comparing the coefficients of $p^{\binom{n}{2}-1}$ and using our newly found formula for $H_{\text{last}}(K_n)$, we can, with a bit of work, see that

$$H_{\text{second last}}(K_n) = \tfrac{1}{2}(n-1)! + (n-1) H_{\text{second last}}(K_{n-1})$$

for $n \geq 4$. From this last formula and the formula for $H_{\text{last}}(K_n)$, we find that

$$H_{\text{second last}}(K_n) = \frac{n-2}{2}(n-1)! = \frac{n-2}{2} H_{\text{last}}(K_n)$$

for $n \geq 4$. In particular, $H_{\text{second last}}(K_n) > H_{\text{last}}(K_n)$ for all $n \geq 5$, so we are done. □

Thus there are many 2-connected graphs for which Proposition 4.1 fails to show whether the roots lie in the disc $|z-1| \leq 1$. We can show in support of Conjecture 1.1 that every graph is homeomorphic to a graph for which we can apply the proposition.

THEOREM 4.3. *If G is any connected graph of order n with size m, then there is a subdivision $\tilde{G}$ of G such that* $\mathrm{Rel}(\tilde{G}, p)$ *has all its roots in* $|z-1| \leq 1$.

To prove this, we consider the general setting of matroids. Let M be a matroid of rank d on set S with independent sets in $\mathcal{I}$. Given an element $e \in S$, we define an

l-thickening of M at e to be the matroid $M(e,l)$ on set $S-\{e\}\cup\{e_1,\ldots,e_l\}$ (where $e_1,\ldots,e_l$ are not in S) with collection of independent sets $\mathcal{I}(e,l)=\{I\in\mathcal{I}: e\notin I\}\cup\{(I-e)\cup\{e_j\}: e\in I\in\mathcal{I},\ j=1,\ldots,l\}$. An l-thickening of M at e can be achieved as well by a sequence of l 2-thickenings of M. We can consider this operation as replacing e by l copies of itself. A *thickening* of M [10] is a matroid arising by a sequence of thickenings of M (i.e., by replacing each point in S by a number of copies of itself). It is easy to see that a thickening of a cographic matroid corresponds to a subdivision of the corresponding graph (while a thickening of a graphic matroid corresponds to replacing each edge by a set of parallel edges). Recall that we define a two variable polynomial $\alpha(M,p,q)$ on the matroid M of dimension d by

$$\alpha(M,p,q)=\sum_{I\in\mathcal{I}}q^{|I|}p^{|S|-|I|}=\sum_{i=0}^{d}F_ip^{|S|-i}q^i=p^{|S|-d}\sum_{i=0}^{d}H_iq^i.$$

(Here $\langle F_i\rangle$ and $\langle H_i\rangle$ are, respectively, the F- and H-vectors of M.) Note that, if M is the cographic matroid of a connected graph G, then $\alpha(M,p,1-p)=\mathrm{Rel}(G,p)$.

PROPOSITION 4.4. *If M is any matroid, there is an N such that, if we thicken each point of the matroid into at least N points, then, in the resulting matroid $\tilde{M}$, the* H*-vector is nondecreasing (and hence all the roots of $\alpha(\tilde{M},p,1-p)$ lie in $|z-1|\le 1$).*

Proof. Let M have dimension d and ground set S of cardinality ν. If $d=0$, then the H-vector of M is $\langle 1\rangle$ with any thickening of M having the same H-vector (as all elements of M are loops). If $\nu=0$, then $d=0$. Thus assume that $d,\ \nu>0$. Let M have underlying set S.

Assume first that M has a point e that is neither a loop nor a coloop. By induction on d and ν, we can subdivide each point of $S-\{e\}$ enough (say into at least N_0 points) so that, in the resulting matroid M_0, $M_0.T$, and $M_0|T$ have no coloops, and the H-vectors $\langle H'_0,\ldots,H'_{d-1}\rangle$ and $\langle H''_0,\ldots,H''_d\rangle$ of $M_0.T$ and $M_0|T$, respectively, are nondecreasing (as usual, $T=S_0-\{e\}$, where S_0 is the underlying set of M_0). Let M_l be the matroid formed by thickening e of M_0 into l points. It is easy to see that

$$\alpha(M_l,p,1-p)=l\cdot p^{l-1}(1-p)\cdot\alpha(M_0.T,p,1-p)+p^l\cdot\alpha(M_0|T,p,1-p),$$

and we derive

$$H_i=\begin{cases}1 & \text{if } i=0,\\ lH'_{i-1}+H''_i & \text{if } i=1,\ldots,d,\end{cases}$$

where $\langle H_0,\ldots,H_d\rangle$ is the H-vector of M_l. The fact that this sequence is nondecreasing follows, so take $l\ge N_0$ and $N=l$.

The only case remaining is that every point of M is either a loop or a coloop; that is, M is a d-simplex together with some loops. We may assume that there are no loops, as these do not affect the H-vector. Thus, without loss, $S=\{1,\ldots,d\}$, and M is the power set of S. For $i\in\{0,\ldots,d\}$, let $\binom{S}{i}$ denote the set of i-subsets of S. If we thicken each point i of M into k_i points to form $M_{\underline{k}}$ ($\underline{k}=(k_1,\ldots,k_d)$), then the number of faces of cardinality i is $\sum_{T\in\binom{S}{i}}\prod_{j\in T}k_j$. It follows that

$$\begin{aligned}\alpha(M_{\underline{k}},p,q)&=\prod_{j=1}^{d}(p+k_jq)\\&=\prod_{j=1}^{d}(1+k'_jq)\end{aligned}$$

(where $k'_j = k_j - 1$), and thus

$$H_i^{M_{\underline{k}}} = \sum_{T \in \binom{S}{i}} \prod_{j \in T} k'_j.$$

Let $i \geq 1$ and consider any term in this sum, say $k'_1 k'_2 \cdots k'_i$. If, without loss, $k'_i = \min\{k'_1, k'_2, \ldots, k'_i\}$ and $k'_i \geq i$, then

$$\begin{aligned} k'_1 k'_2 \cdots k'_i &\geq i k'_1 k'_2 \cdots k'_{i-1} \\ &\geq \sum_{T \in \binom{\{k'_1, k'_2, \ldots, k'_i\}}{i-1}} \prod_{j \in T} k'_j. \end{aligned}$$

It follows that, if each $k'_j = k_j - 1$ is at least d, then the H-vector of $M_{\underline{k}}$ is nondecreasing. Therefore we take $N = d + 1$, and we are done. □

As thickenings in the cographic matroid correspond to subdivisions of the associated graph and the alpha poynomial of the former equals the reliability poynomial of the latter, we derive Theorem 4.3. In fact, we show in the following theorem that all large subdivisions of any graph have their roots in the desired disc.

THEOREM 4.5. *If G is any graph, there is an N such that, if we subdivide each edge of the graph into a path of length at least N, then, in the resulting graph $\tilde{G}$, all the roots of* $\mathrm{Rel}(G, p)$ *lie in* $|z - 1| \leq 1$.

This theorem implies that, to prove Conjecture 1.1, we need only show that the property of having its roots in the disc $|z - 1| \leq 1$ is preserved under a degree-2 reduction in loopless graphs (a *degree-2 reduction* is the removal of a vertex of degree 2 and adding an edge between its two neighbours).

We conclude this section by showing that the roots of the reliability polynomial of any connected graph of order $n \geq 2$ are bounded away from $z = 1$.

THEOREM 4.6. *If G is a connected graph of order $n \geq 2$, then the roots z of* $\mathrm{Rel}(G, p)$ *satisfy* $|z - 1| \geq 1/(n - 1)$.

Proof. The result is true if $n = 2$, as then G is a collection of parallel edges between two vertices; we check that all the roots of $\mathrm{Rel}(G, p)$ satisfy $|z - 1| = 1$. Thus $n \geq 3$. If G has a cut edge, then, using the facts that G is not 2-connected and that the reliability of G is the product of the reliabilities of its blocks, we can conclude the result by induction on n. Thus we can assume that G has no cut edges. Let G have m edges, let

$$\mathrm{Rel}(G, p) = p^{n-1} \sum_{i=0}^{d} H_i (1 - p)^i,$$

and set $h(w) \equiv \sum_{i=0}^{d} H_i w^i$. It clearly suffices to show that all roots w of H satisfy $|w| \geq 1/(n - 1)$. Now, by a well-known theorem on polynomials (cf. [3, eq. (6c), p. 181]), if $p(x) = \sum_{i=0}^{k} a_i x^i$ is a polynomial of degree k with positive coefficients, then the roots z of p satisfy $|w| \geq u$, where

$$u \equiv \min\left\{ \frac{a_i}{a_{i+1}} : i = 0, \ldots, k - 1 \right\}.$$

Thus, returning to H, it suffices to show that $H_0/H_1 \leq H_i/H_{i+1}$ for all $i = 1, \ldots, d-1$.

It holds that $F_0 = 1$, and that, as G has no cut edges, $F_1 = m$. From the formula

$$H_i = \sum_{j=0}^{i} (-1)^{i-j} \binom{m-n+1-j}{m-n+1-i} F_j, \qquad (0 \le i \le m-n+1),$$

mentioned earlier, we calculate that $H_0 = 1$ and $H_1 = n-1$. Thus we must show that

$$H_{i+1} \le (n-1)H_i, \qquad (1 \le i \le d-1).$$

Now a result of Stanley [22] (see also [12]) states the following bound on H-vectors of a family of pure complexes that includes all matroids. Let $(H_0, \ldots, H_d)$ be the H-vector of such a complex. Fix $i \in \{0, \ldots, d-1\}$. The *i-canonical form* of a positive integer r is the integer-valued vector $(a_i, a_{i-1}, \ldots, a_j)$, where

$$r = \binom{a_i}{i} + \binom{a_{i-1}}{i-1} + \cdots + \binom{a_j}{j}$$

and

$$a_i > a_{i-1} > \ldots a_j \ge j \ge 1$$

(such a representation exists for all r, and i and is unique). Stanley has shown that, if H_i has i-canonical form $(a_i, a_{i-1}, \ldots, a_j)$, then $H_{i+1} \le H_i^{<i+1/i>}$, where

$$H_i^{<i+1/i>} \equiv \binom{a_i+1}{i+1} + \binom{a_{i-1}+1}{i} + \cdots + \binom{a_j+1}{j+1}.$$

Thus, to show $H_{i+1} \le (n-1)H_i$, it suffices to show that

$$H_i^{<i+1/i>} \le (n-1)H_i, \qquad (1 \le i \le d-1).$$

As above, let H_i have i-canonical form $(a_i, a_{i-1}, \ldots, a_j)$. Then

$$H_i^{<i+1/i>} = \binom{a_i+1}{i+1} + \binom{a_{i-1}+1}{i} + \cdots + \binom{a_j+1}{j+1},$$

so we must show that

$$\binom{a_i+1}{i+1} + \binom{a_{i-1}+1}{i} + \cdots + \binom{a_j+1}{j+1} \le (n-1)\left(\binom{a_i}{i} + \binom{a_{i-1}}{i-1} + \cdots + \binom{a_j}{j}\right);$$

i.e.,

$$(n-1)\binom{a_i}{i} - \binom{a_i+1}{i+1} + (n-1)\binom{a_{i-1}}{i-1} - \binom{a_{i-1}+1}{i} + \cdots + (n-1)\binom{a_j}{j} - \binom{a_j+1}{j+1} \ge 0.$$

To prove this, we show that

$$(n-1)\binom{a_k}{k} - \binom{a_k+1}{k+1} \ge 0$$

for $k = i, \ldots, j$. Now, by some elementary simplifications, this holds if and only if

$$(***) \qquad a_k \le (n-1)(k+1) - 1$$

for $k = i, \ldots, j$. To prove this, we show first that

$$(****) \qquad a_k \leq n - 2 + k.$$

Consider first $k = i$. H_i is the number of multisets of cardinality i in a multicomplex on a set of size $|H_1| = n - 1$. There are at most $\binom{n-2+i}{i}$ of such multisets, and hence

$$\binom{a_i}{i} \leq H_i \leq \binom{n-2+i}{i}.$$

This implies that $a_i \leq n-2+i$, and we have shown $(****)$ for $k = i$. For $k = j, \ldots, i-1$, we see that, from the definition of the i-canonical form, we have

$$a_k \leq a_i - (i - k),$$

so that $a_k \leq n-2+i-(i-k) = n-2+k$; we have proved $(****)$. Now $n-2+k \leq (n-1)(k+1)-1$ as $n \geq 2$, so $(***)$ follows as well, and we are done. □

The cycle of order n illustrates that equality can occur in the theorem above for every $n \geq 2$. Also, in [1] it is shown that a polynomial $\sum_{i=0}^{N} b_i z^i$ with positive coefficients has a root on $|z| = \alpha = \min\{b_i/b_{i+1} : 0 \leq i < N\}$ only if $\gcd(\{i = 1, 2, \ldots, N+1 : b_{i-1} - \alpha b_i > 0\}) > 1$, where $b_{N+1} \equiv 0$. From the proof of Theorem 4.6, if $n \geq 3$ (i.e., G is not a collection of parallel edges), then $n-2+k < (n-1)(k+1)-1$, so that the minimum of H_i/H_{i+1} occurs only at $i = 0$. Thus, in this case, if we consider the polynomial $\mathrm{Rel}(G, p)$, we see that, with the notation above, that $\alpha = \gcd(\{2, 3, \ldots, d+1\}) = 1$ if $d \geq 3$. It follows that, if G is a loopless graph of order at least 3 with cyclomatic number at least 2, then, in fact, the roots of $\mathrm{Rel}(G, p)$ satisfy $|z - 1| > 1/(n-1)$.

5. Conclusion. In an attempt to further investigate Conjecture 1.1, perhaps we might somehow be able to utilize Rouché's theorem. We could also attempt to prove the conjecture for some interesting families of graphs. We do not know if the conjecture holds even for all complete graphs, and the approach of the previous sections do not appear to apply, as the H-vectors of complete graphs need not be nondecreasing.

It would be interesting as well to discover more about the nature of the roots of reliability polynomials as well. Can we find a bounded region that is guaranteed to contain all the roots of the reliability polynomials? Are the roots of the reliability polynomials dense in $|z - 1| \leq 1$? We show that the answer to the latter is in the affirmative. For $x \in \mathbb{C}$ and real $r > 0$, let $D(x, r)$ be the open disc in the complex plane centered at x with radius r.

PROPOSITION 5.1. *Let $\mathcal{S}$ be the collection of all roots of reliability polynomials of graphs, and let $\mathcal{R}$ be the subcollection of all real roots of reliability polynomials. Then $\overline{\mathcal{S}} \supseteq \overline{D(1,1)}$ and $\overline{\mathcal{R}} = \{0\} \cup [1, 2]$.*

Proof. It suffices to show that, if w is a complex number in D and $\varepsilon > 0$ have the property that $D(w, \varepsilon) \subseteq D(1, 1)$, then $D(w, \varepsilon)$ contains a root of some reliability polynomial. Let $1 - w = re^{i\theta}$, where $r > 0$ and $\theta \geq 0$.

Consider the graph C_n^l formed from the cycle C_n of order n by replacing each edge by l parallel edges. It is easy to see that

$$\mathrm{Rel}(C_n^l, p) = \mathrm{Rel}(C_n, 1 - (1-p)^l).$$

As $\mathrm{Rel}(C_n, p) = np^{n-1} - (n-1)p^n$ has root $1 + 1/(n-1)$, we see that the nonzero roots of C_n^l are precisely those z such that $1 - (1-z)^l = 1 + 1/(n-1)$; i.e., $1 - z$ is an

lth root of $-1/(n-1)$. It follows that $1-z$ has modulus $(1/n-1)^{1/l}$ and argument $(2k+1)\pi/l$ for some $k=0,\ldots,l-1$. Now, for all sufficiently large integers l, there is an integer $n \geq 2$ such that

$$(*****) \qquad r-\frac{\varepsilon}{2} < \left(\frac{1}{n-1}\right)^{1/l} < r+\frac{\varepsilon}{2}.$$

Let $\delta > 0$ be such that $\{y \in D(1, r-\varepsilon/2) \cap D(1, r+\varepsilon/2) : \theta - \delta < \arg(y) < \theta + \delta\} \subseteq D(w, \varepsilon)$. Choose l so large that $(*****)$ holds for some $n \geq 2$, and $(2k+1)\pi/l$ lies between $\theta - \delta$ and $\theta + \delta$ for some $k = 0, \ldots, l-1$, as well. Then, for such choices of l and n, $\mathrm{Rel}(C_n^l, p)$ will have a root in $D(w, \varepsilon)$. This shows that $\overline{\mathcal{S}} \supseteq \overline{D(1,1)}$.

Finally, note that that one of the roots of C_n^l for l odd is the real number $1 + (1/n-1)^{1/l}$. By the argument above, $\{(1/n-1)^{1/l} : l \geq 1 \text{ odd}, n \geq 2\}$ is dense in $[0,1]$,and hence $\{1 + (1/n-1)^{1/l} : n \geq 2, l \geq 1 \text{ odd}\}$ is dense in $[1,2]$. It follows that $\overline{\mathcal{R}} \supseteq \{0\} \cup [1,2]$, and, as $\mathcal{R} \subseteq [1,2]$ by Corollary 3.2, $\mathcal{R}$ is dense in $\{0\} \cup [1,2]$. □

Another interesting question relates to what real numbers appear often as roots of reliability polynomials of 2-connected graphs. Along these lines, we have found the following curious result concerning complete bipartite graphs.

PROPOSITION 5.2. *$\frac{4}{3}$ is a root of* $\mathrm{Rel}(K_{2,m}, p)$ *if and only if m is even (and this is the only possible nonzero real root of* $\mathrm{Rel}(K_{2,m}, p)$*). Also, the* H*-vector of $K_{2,m}$ explicitly is*

$$\left(\binom{m}{0}, \binom{m}{0}+\binom{m}{1}, \ldots, \binom{m}{0}+\binom{m}{1}+\cdots+\binom{m}{m-1}\right),$$

so the H*-vector of $K_{2,m}$ is increasing, and hence* $\mathrm{Rel}(K_{2,m}, p)$ *has its roots in the conjectured disc.*

Proof. We begin first by finding a closed form for the reliability of such a graph. First, we show that

$$p\mathrm{Rel}(K_{2,m}, p) = p^{m+1}((1+q)^m - (2q)^m),$$

where $q = 1-p$ and $m \geq 0$ (and hence $\mathrm{Rel}(K_{2,m}, p) = p^m((1+q)^m - (2q)^m)$).

The proof of this proceeds by induction on m. For $m = 0$ or 1, the result is easy to verify, so we assume that $m \geq 2$ and that the result holds for all smaller values of m. From [12, p. 34], the reliability of $K_{2,m}$ can be calculated recursively from the formula

$$1 = \sum_{i=1}^{2}\sum_{j=0}^{m}\binom{1}{i-1}\binom{m}{j}\mathrm{Rel}(K_{i,j}, p)q^{i(m-j)+j(2-i)}.$$

Noting that $\mathrm{Rel}(K_{1,j}, p) = p^j$, we derive

$$\begin{aligned}\mathrm{Rel}(K_{2,m}, p) &= 1 - \sum_{j=0}^{m}\binom{m}{j}p^j q^m - \sum_{j=0}^{m-1}\binom{m}{j}\mathrm{Rel}(K_{2,j}, p)q^{2(m-j)} \\ &= 1 - (1-p^2)^m - \sum_{j=0}^{m-1}\binom{m}{j}\mathrm{Rel}(K_{2,j}, p)q^{2(m-j)}.\end{aligned}$$

It follows by induction that

$$\begin{aligned}
p\mathrm{Rel}(K_{2,m},p) &= p - p(1-p^2)^m - \sum_{j=0}^{m-1} \binom{m}{j} p^{j+1}((1+q)^j - (2q)^j) q^{2m-2j} \\
&= p - p(1-p^2)^m - \sum_{j=0}^{m-1} \binom{m}{j} \left(pq^{2m} \left(\frac{p(1+q)}{q^2} \right)^j - pq^{2m} \left(\frac{2pq}{q^2} \right)^j \right) \\
&= p - p(1-p^2)^m - pq^{2m} \sum_{j=0}^{m-1} \binom{m}{j} \left(\left(\frac{p(1+q)}{q^2} \right)^j - \left(\frac{2pq}{q^2} \right)^j \right) \\
&= p - p(1-p^2)^m - pq^{2m} \left(\left(1 + \frac{p(1+q)}{q^2} \right)^m - \left(\frac{p(1+q)}{q^2} \right)^m - \right. \\
&\qquad \left. \left(1 + \frac{2p}{q} \right)^m + \left(\frac{2p}{q} \right)^m \right) \\
&= p - p(1-p^2)^m - pq^{2m} \left(\frac{1}{q^{2m}} - \frac{p^m(1+q)^m}{q^{2m}} - \frac{(1+p)^m}{q^m} + \left(\frac{2p}{q} \right)^m \right) \\
&= p - p(1-p^2)^m - p + p^{m+1}(1+q)^m + pq^m(1+p)^m - p^{m+1}q^m 2^m \\
&= p^{m+1}((1+q)^m - (2q)^m).
\end{aligned}$$

We derive now that $\mathrm{Rel}(K_{2,m},p) = p^m((1+q)^m - (2q)^m)$. If $p \neq 0$ is a real root of $\mathrm{Rel}(K_{2,m},p)$, then

$$(1+q)^m = (2q)^m.$$

If m is odd, then this implies that $1+q = 2q$, i.e., $p = 0$, which we ignore. If m is even, then we derive that

$$1 + q = \pm(2q),$$

which is equivalent to $p = 0$ or $\frac{4}{3}$.

Finally, from above,

$$\begin{aligned}
p\mathrm{Rel}(K_{2,m},p) &= p^{m+1} \left((1+q)^m - q^m - \left(\binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{m-1} \right) q^m \right) \\
&= p^{m+1} \left(\binom{m}{0} q^0 + \binom{m}{1} q^1 + \cdots + \binom{m}{m-1} q^{m-1} \right. \\
&\quad \left. - \left(\binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{m-1} \right) q^m \right),
\end{aligned}$$

so it follows that

$$\begin{aligned}
\mathrm{Rel}(K_{2,m},p) &= p^{m+1} \frac{1}{1-q} \left(\binom{m}{0} q^0 + \binom{m}{1} q^1 + \cdots + \binom{m}{m-1} q^{m-1} \right. \\
&\qquad \left. - \left(\binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{m-1} \right) q^m \right) \\
&= p^{m+1} \left(\binom{m}{0} q^0 + \left(\binom{m}{0} + \binom{m}{1} \right) q^1 + \cdots \right. \\
&\qquad \left. + \left(\binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{m-1} \right) q^{m-1} \right).
\end{aligned}$$

The H-vector can immediately be read from the last equation. □

Finally, we note that, if we subdivide every edge of a connected graph G into a path with k edges to form a graph G', then $\mathrm{Rel}(G', p) = (p^k + kp^{k-1}(1-p))\,\mathrm{Rel}(G, p^k/(p^k + kp^{k-1}(1-p)))$. It then follows from this and Proposition 5.2 that, for every $k \geq 1$, $4k/4k-1$ is a root of infinitely many 2-connected graphs.

REFERENCES

[1] N. ANDERSON, E. B. SAFF, AND R. S. VARGA, *On the Eneström–Kakeya theorem and its sharpness*, Linear Algebra Appl., 28 (1979), pp. 5–16.

[2] M. O. BALL AND J. S. PROVAN, *Bounds on the reliability polynomial for shellable independence systems*, SIAM J. Alg. Discrete Meth., 3 (1982), pp. 166–181.

[3] E. J. BARBEAU, *Polynomials*, Springer-Verlag, New York,1989.

[4] G. BERMAN AND W. T. TUTTE, *The golden root of a chromatic polynomial*, J. Combin. Theory, 6 (1969), pp. 301–302.

[5] N. L. BIGGS, R. M. DAMERELL, AND D. A. SANDS, *Recursive families of graphs*, J. Combin. Theory B, 12 (1972), pp. 123–131.

[6] L. J. BILLERA AND C. W. LEE, *A proof of the sufficiency of McMullen's conditions for f-vectors of simplicial convex polytopes*, J. Combin. Theory A, 31 (1981), pp. 237–255.

[7] F. T. BOESCH, A. SATYANARAYANA, AND C. L. SUFFEL, *Least reliable networks and the reliability domination*, IEEE Trans. Comm., 38 (1990), pp. 2004–2009.

[8] J. I. BROWN AND C. J. COLBOURN, *A combinatorial study of the reliability polynomial*, in Proc. Sixteenth Manitoba Conf. Num. Math. and Computing (Congressus Numeratium, 56), Winnipeg, Manitoba, Canada, 1986, pp. 71–89.

[9] ———, *A set system polynomial with colouring and reliability applications*, SIAM J. Discrete Math., 1 (1988), pp. 151–157.

[10] ———, *On the log concavity of reliability and matroidal sequences*, Adv. Appl. Math., to appear.

[11] J. I. BROWN, C. J. COLBOURN, AND J. S. DEVITT, *Network transformations and bounding network reliability*, Networks, to appear.

[12] C. J. COLBOURN, *The Combinatorics of Network Reliability*, Oxford University Press, New York, 1987.

[13] G. ENESTRÖM, *Remarque sur un théorème relatif aux racines de l'equation* $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ *où tous les coefficientes a sont réels et positifs*, Tôhoku Math. J., 18 (1920), pp. 34–36.

[14] E. J. FARRELL, *Chromatic roots—Some observations and conjectures*, Discrete Math., 29 (1980), pp. 161–167.

[15] D. GERNERT, *A survey of partial proofs for Read's conjecture and some recent results*, Meth. Oper. Res., 49 (1985), pp. 233–238.

[16] E. N. GILBERT, *Random graphs*, Ann. Math. Statist., 30 (1959), pp. 1141–1144.

[17] A. HURWITZ, *Über einen Satz des Herrn Kakeya*, Tôhoku Math. J., 4 (1913), pp. 89–93.

[18] S. KAKEYA, *On the limit of the roots of an algebraic equation with positive coefficients*, Tôhoku Math. J., 2 (1912), pp. 140–142.

[19] G. PÓLYA AND G. SZEGÖ, *Problems and Theorems in Analysis*, Vol. 1, Springer-Verlag, New York, 1972.

[20] R. C. READ, *Some applications of computers in graph theory*, in Selected Topics in Graph Theory, L. W. Beineke and R. J. Wilson, eds., Academic Press, New York, 1978, pp. 417–444.

[21] R. C. READ AND W. T. TUTTE, *Chromatic polynomials*, in Selected Topics in Graph Theory 3, L. W. Beineke and R. J. Wilson, eds., Academic Press, New York, 1988, pp. 15–42.

[22] R. P. STANLEY, *Cohen–Macaulay complexes*, in Higher Combinatorics, M. Aigner, ed., Reidel, Dordrecht, the Netherlands, 1977, pp. 51–64.

[23] ———, *Combinatorics and Commutative Algebra*, Birkhäuser, Boston, 1983.

[24] W. T. TUTTE, *A ring in graph theory*, Proc. Cambridge Phil. Soc., 43 (1947), pp. 26–40.

[25] D. J. A. WELSH, *Matroid Theory*, Academic Press, London, 1976.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 586–595, November 1992
©1992 Society for Industrial and Applied Mathematics
014

LABELLING GRAPHS WITH A CONDITION AT DISTANCE 2*

JERROLD R. GRIGGS† AND ROGER K. YEH†‡

Abstract. Given a simple graph $G = (V, E)$ and a positive number d, an $L_d(2,1)$-*labelling* of G is a function $f : V(G) \to [0, \infty)$ such that whenever $x, y \in V$ are adjacent, $|f(x) - f(y)| \geq 2d$, and whenever the distance between x and y is two, $|f(x) - f(y)| \geq d$. The $L_d(2,1)$-*labelling number* $\lambda(G, d)$ is the smallest number m such that G has an $L_d(2,1)$-labelling f with $\max\{f(v) : v \in V\} = m$.

It is shown that to determine $\lambda(G, d)$, it suffices to study the case when $d = 1$ and the labelling is nonnegative integral-valued. Let $\lambda(G) = \lambda(G, 1)$. The labelling numbers of special classes of graphs, e.g., $\lambda(C) = 4$ for any cycle C, are described. It is shown that for graphs of maximum degree Δ, $\lambda(G) \leq \Delta^2 + 2\Delta$. If G is diameter 2, $\lambda(G) \leq \Delta^2$, a sharp bound for some Δ. Determining $\lambda(G)$ is shown to be NP-complete by relating it to the problem of finding Hamilton paths.

Key words. T-coloring, channel assignments, graph coloring, NP-completeness

AMS(MOS) subject classifications. 05C15, 05C35, 05C78, 05C85, 68R10

1. Introduction.

There has been a considerable effort (cf. [CR], [CW], [FGK], [G], [H], [R1], [R2], [Ro1], and [T]) to study properties of "T-colorings" of graphs, which is motivated by the task of assigning channel frequencies without interference. Roberts [Ro2] proposed the problem of efficiently assigning radio channels to transmitters at several locations, using nonnegative integers to represent channels, so that close locations receive different channels, and channels for very close locations are at least two apart. Therefore these channels would not interfere with each other.

We propose an analogous problem for simple graph $G = (V, E)$. Given a real number $d > 0$, an $L_d(2,1)$-*labelling* of G is a nonnegative real-valued function $f : V(G) \to [0, \infty)$ such that, whenever x and y are two adjacent vertices in V, then $|f(x) - f(y)| \geq 2d$, and, whenever the distance between x and y is 2, then $|f(x) - f(y)| \geq d$. The $L_d(2,1)$-*labelling number* of G is the smallest number m such that G has an $L_d(2,1)$-labelling with no label greater than m and is denoted by $\lambda(G, d)$. If f is an $L_d(2,1)$-labelling of G, then we say that $f \in L_d(2,1)(G)$.

Let G be a graph and $f \in L_d(2,1)(G)$. Define $\|f(G)\| = \max\{f(v) : v \in V(G)\}$. Then $\lambda(G, d) = \min \|f(G)\|$, where the minimum runs over all $f \in L_d(2,1)(G)$. In the language of Roberts [Ro3], we are trying to minimize the *span* of an $L_d(2,1)$-labelling. However, we allow 0 to be a label, unlike most other analogous parameters, because we can then nicely characterize $\lambda(G, d)$ in terms of $\lambda(G, 1)$. We describe this in §2, where we also show that for $\lambda(G, 1)$ it suffices to consider integral-valued labellings. Thereafter we confine our study to $\lambda(G, 1)$, which we denote simply by $\lambda = \lambda(G)$. Similarly, $L(2,1) = L(2,1)(G)$ denotes $L_1(2,1)(G)$. We let $[0, k]$ denote the set $\{0, 1, \ldots, k\}$.

In §§3–5 we consider the labelling numbers of some fundamental classes of graphs. In §6 we present general upper bounds on λ in terms of the maximum degree Δ. We

*Received by the editors July 13, 1990; accepted for publication (in revised form) February 6, 1992. Research supported in part by NSA/MSP (National Security Agency Mathematical Sciences Program) grant MDA90-H-4028.

† Department of Mathematics, University of South Carolina, Columbia, South Carolina 29208.

‡ Present address, Department of Applied Mathematics, Feng Chia University, Seatwen, Taichung, Taiwan, Republic of China.

find that λ is never much larger than Δ^2. Diameter 2 graphs are studied in the next section, and the sharp upper bound Δ^2 is obtained for λ in this case. Infinite families of graphs with λ close to Δ^2 are described in §8. After investigating the complexity of the $L_1(2,1)$-labelling problem in §9, we conclude by proposing some problems for further research.

2. Reduction to integral-valued labellings. First, we want to characterize $\lambda(G,d)$ in terms of $\lambda(G,1)$. Furthermore, we show that to determine $\lambda(G,1)$ it suffices to study the case when the labelling is integral-valued.

LEMMA 2.1. *It holds that* $\lambda(G,d) = d \cdot \lambda(G,1)$.

Proof. We prove the lemma with the following claims.

Claim 1. We have that $\lambda(G,d) \geq d \cdot \lambda(G,1)$.

Let $f \in L_d(2,1)(G)$. Define $f_1(x) = f(x)/d$, for all $x \in V(G)$. It follows easily that $f_1 \in L_1(2,1)(G)$. This implies that $\|f(G)\|/d = \|f_1(G)\| \geq \lambda(G,1)$. By compactness, some f attains $\lambda(G,d)$, and the claim follows.

Claim 2. We have that $\lambda(G,d) \leq d \cdot \lambda(G,1)$.

The proof is similar to Claim 1. Therefore the result follows. □

LEMMA 2.2. *Let* $x, y \geq 0$, $d > 0$ *and* $k \in Z^+$. *If* $|x-y| \geq kd$, *then* $|x'-y'| \geq kd$, *where* $x' = \lfloor x/d \rfloor d$ *and* $y' = \lfloor y/d \rfloor d$.

The two lemmas above imply the following theorem.

THEOREM 2.3. *Given a graph* G, *there is an* $f \in L_1(2,1)(G)$ *such that* f *is integral-valued and* $\|f(G)\| = \lambda(G,1)$.

For general d, we see that $\lambda(G,d)$ is attained by some $f \in L_d(2,1)(G)$ whose values are all multiples of d, i.e., $f = d \cdot f'$, where $f' \in L_1(2,1)(G)$ is integral-valued (by Lemma 2.1). Therefore it suffices to study the case where $d = 1$ and to consider in what follows only integral-valued $f \in L_1(2,1)(G)$.

3. Paths, cycles, and cubes. First, let us look at the $L(2,1)$-labelling of an elementary graph, the path. We have the following easy result (cf. [Y]).

PROPOSITION 3.1. *Let* P_n *be a path on* n *vertices. Then* (i) $\lambda(P_2) = 2$, (ii) $\lambda(P_3) = \lambda(P_4) = 3$, *and* (iii) $\lambda(P_n) = 4$, *for* $n \geq 5$.

If we join the first vertex and the last vertex of a path, then we have a cycle. So what is the labelling number of a cycle?

PROPOSITION 3.2. *Let* C_n *be a cycle of length* n. *Then* $\lambda(C_n) = 4$, *for any* n.

Proof. If $n \leq 4$, then it is easy to verify the result. Thus suppose that $n \geq 5$. For all $n \geq 5$, C_n must contain a P_5 as a subgraph. Hence $\lambda(C_n) \geq \lambda(P_5) = 4$, by Proposition 3.1.

Now we want to show that $\lambda(C_n) \leq 4$, $n \geq 5$. It suffices to show that there is an $L(2,1)$-labelling f such that $\|f(C_n)\| = 4$. Let $v_0, \ldots, v_{n-1}$ be vertices of C_n such that v_i is adjacent to v_{i+1}, $0 \leq i \leq n-2$ and v_0 is adjacent to v_{n-1}. Consider the following labelling:

(1) If $n \equiv 0 \pmod 3$, then define

$$f(v_i) = \begin{cases} 0, & \text{if } i \equiv 0 \pmod 3, \\ 2, & \text{if } i \equiv 1 \pmod 3, \\ 4, & \text{if } i \equiv 2 \pmod 3; \end{cases}$$

(2) If $n \equiv 1 \pmod 3$, then redefine the above f at $v_{n-4}, \ldots, v_{n-1}$ as

$$f(v_i) = \begin{cases} 0, & \text{if } i = n-4, \\ 3, & \text{if } i = n-3, \\ 1, & \text{if } i = n-2, \\ 4, & \text{if } i = n-1; \end{cases}$$

(3) If $n \equiv 2 \pmod 3$, then redefine the f in (1) at v_{n-2} and v_{n-1} as

$$f(v_i) = \begin{cases} 1, & \text{if } i = n-2, \\ 3, & \text{if } i = n-1. \end{cases}$$

It is easy to show that f, defined above, is in $L(2,1)(C_n)$ for every n, for each case. Hence $\lambda(C_n) \leq 4$. Therefore the theorem is proved. □

If we take a cycle C_n joined by a vertex, then we have a graph W_n called a *wheel of length* n, i.e., $W_n = C_n \vee K_1$. In [Y] it is shown that $\lambda(W_n) = n+1$.

Next, consider the *n-cube* Q_n, which has 2^n vertices $v = (v_1, \ldots, v_n)$, where each v_i is 0 or 1, and edges join vertices v, w when there exists a unique i such that $v_i \neq w_i$. This bipartite graph is regular of degree n.

THEOREM 3.3. *Let Q_n be the n-cube. Then, for all $n \geq 5$, $n+3 \leq \lambda(Q_n) \leq 2n+1$.*

Proof. The following modular labelling implies the stated upper bound for $n \geq 1$:

$$f(v) = \sum_{i:v_i=1} (i+1) \pmod{2n+2},$$

where all labels are chosen to belong to $[0, 2n+1]$. To verify this, consider adjacent vertices v and w. We may assume that $v_i - w_i = 1$ (0, respectively) when $i = a$ ($i \neq a$, respectively). Then $f(v) - f(w) \equiv a+1 \pmod{2n+2}$, so that $|f(v) - f(w)| \geq 2$. Similarly, if v and w are at distance 2, we may assume that $v_i - w_i$ is 1 when $i = a$, 1 or -1 when $i = b$, and 0 otherwise. Then $f(v) - f(w) \equiv a+b+2$ or $a-b \pmod{2n+2}$, so that $f(v) \neq f(w)$.

The lower bound of $n+3$ is due to Jonas [J]. Suppose for contradiction that $\lambda(Q_n) \leq n+2$ for some $n \geq 5$ and let f be an optimal labelling for such Q_n. Some vertex v is labelled 0 in an optimal labelling. The n vertices adjacent to v receive labels that are distinct and greater than 1; i.e., each of $2, 3, \ldots, n+2$ is used with just one exception i. Since $n \geq 5$, if the labelling f does not use the label 3, it must use the label $n-1$. In the later case, we may "reflect" f and instead consider another optimal labelling, $n+2-f$. By permuting vertices, we may assume that our optimal labelling f assigns 3 to vertex w. Let W_i denote the set of vertices at distance i from w. The vertices in W_1 must receive the distinct labels $0, 1, 5, 6, \ldots, n+2$. There are $\binom{n}{2}$ vertices in W_2, each adjacent to two vertices in W_1. If $x \in W_2$ is adjacent to the vertex in W_1 with label j, then $f(x) \neq j-1, j, j+1$. Two vertices in W_2 with the same label have no neighbors in common. It follows that label i is used on W_2 at most $\lfloor (n-2)/2 \rfloor$ times when $i = 0, 1, 5, n+2$; $\lfloor (n-1)/2 \rfloor$ times when $i = 2, 4$; $\lfloor (n-3)/2 \rfloor$ times when $i = 6, \ldots, n+1$. Label 3 cannot be used on W_2. Adding up the possible labels does not account for all $\binom{n}{2}$ vertices in W_2, a contradiction. □

With considerable effort, we have determined the first several values as follows: $\lambda(Q_0) = 0$, $\lambda(Q_1) = 2$, $\lambda(Q_2) = 4$, $\lambda(Q_3) = 6$, $\lambda(Q_4) = 7$, $\lambda(Q_5) = 8$. No pattern is yet evident in the labellings that attain these values. Jonas recently showed that $\lambda(Q_n) \geq n+4$ for $n = 8$ and 16. Using methods from coding theory, it was recently shown by Jonas and by Georges, Mauro, and Whittlesey that $\liminf_{n\to\infty} \lambda(Q_n)/n = 1$.

4. Trees. We next discuss the labelling numbers of connected graphs without cycles, that is, trees. The maximum degree nearly determines the labelling number.

THEOREM 4.1. *Let T be a tree with maximum degree $\Delta \geq 1$. Then $\lambda(T)$ is either $\Delta + 1$ or $\Delta + 2$.*

Proof. Since T contains the star $K_{1,\Delta}$, we have $\lambda(T) \geq \lambda(K_{1,\Delta}) = \Delta + 1$. We obtain the upper bound by a first-fit (greedy) labelling. First, order $V(T)$ so that $V(T) = \{v_1, \ldots, v_n\}$, where, for all $i > 1$, v_i is attached just once to $\{v_1, \ldots, v_{i-1}\}$. This can be done since T is a tree. Now we describe an $L(2,1)$-labelling of T: Label v_1 as 0; then successively label $v_2, v_3, \ldots, v_n$ by the lowest available element of $[0, \Delta + 2]$. Since each v_i, $2 \leq i \leq n$, is adjacent to only one v_j, $j < i$ and is distance 2 away from at most $\Delta - 1$ v_j's with $j < i$, there are at most $\Delta + 2$ labels that cannot be used for v_i. Hence at least one label in $[0, \Delta + 2]$ is available to v_i when its turn comes to be labeled. Thus the labelling number is at most $\Delta + 2$, and the theorem follows. □

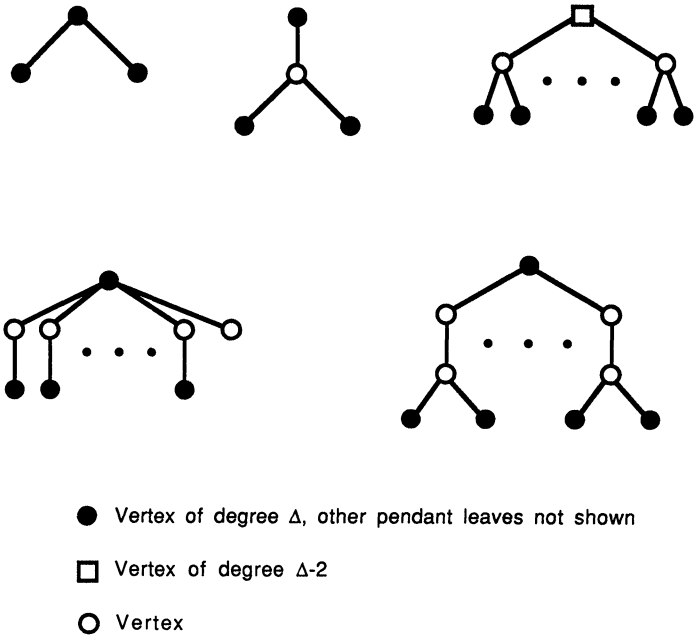


FIG. 1. *Critical trees with labelling number $\Delta + 2, \Delta \geq 3$.*

Both values can occur. The value $\Delta + 1$ holds for many trees, e.g., the star $K_{1,\Delta}$. We exhibit several trees in Fig. 1 with $\lambda = \Delta + 2$. All trees shown are, in fact, λ–critical; i.e., deleting any vertex (or edge) drops λ. It seems that characterizing all trees with $\lambda = \Delta + 2$ is very difficult (see §10).

5. k-colorable graphs. Before considering the labelling number of general graphs G, we want to look at graphs with specified chromatic number $\chi(G)$.

THEOREM 5.1. *Let G be a graph with $\chi(G) = k$ and $|V(G)| = \nu$. Then $\lambda(G) \leq \nu + k - 2$.*

Proof. Since $\chi(G) = k$, we can partition G into $G_1 \cup \cdots \cup G_k$, where $|V(G_i)| = \nu_i$ and each G_i is an independent set. Let $V_i = V(G_i) = \{v_{i,1}, v_{i,2}, \ldots, v_{i,\nu_i}\}$, $1 \leq i \leq k$. Now consider the labelling f defined by

$$f(v_{1,j}) = j - 1, \qquad 1 \leq j \leq \nu_1,$$

$$f(v_{i,j}) = \sum_{t=1}^{i-1} \nu_t + i + j - 2, \qquad 1 \le j \le \nu_i, \qquad \text{for } 2 \le i \le k.$$

It is easily verified that f is in $L(2,1)(G)$. Hence $\lambda(G) \le \|f(G)\| = \nu + k - 2$. □

COROLLARY 5.2. *Let G be a complete k-partite graph with $|V(G)| = \nu$. Then $\lambda(G) = \nu + k - 2$.*

Proof. Since G is a k-partite graph, $\chi(G) = k$. By Theorem 5.1, $\lambda(G) \le \nu + k - 2$. On the other hand, since the distance between any two vertices in G is at most 2, the labels must be distinct. Furthermore, consecutive labels cannot be used at vertices from different parts. Since we have k components, we find that $\lambda(G) \ge \nu + k - 2$, and the result follows. □

6. Upper bounds on λ in terms of the maximum degree. In this section, we determine the upper bound on $\lambda(G)$ in terms of the maximum degree of G. The upper bound we have is analogous to the Brooks theorem.

THEOREM 6.1 (Brooks [Br]). *Let G be a connected graph of maximum degree Δ. If G is not a complete graph or an odd cycle, then $\chi(G) \le \Delta$.*

Before showing the result, whose proof is analogous to that of Theorem 6.1, we give the following simple result, which uses a first-fit (greedy) labelling to provide a bound on λ in terms of the maximum degree Δ for any graph.

THEOREM 6.2. *Let G be a graph with maximum degree Δ. Then $\lambda(G) \le \Delta^2 + 2\Delta$.*

Proof. Arbitrarily order the vertices of G, and label them in succession by the lowest allowed integer. A vertex $v \in V$ is adjacent to at most Δ vertices, and there are at most $\Delta^2 - \Delta$ vertices, which are distance 2 away from v. So, when we want to label v, there are at most $3\Delta + \Delta^2 - \Delta = \Delta^2 + 2\Delta$ numbers to be avoided. Thus the labelling number $\lambda(G)$ is at most $\Delta^2 + 2\Delta$. (Since we can use 0 to label a vertex, there are $\Delta^2 + 2\Delta + 1$ numbers that can be used.) □

We can improve the bound above when G is 3-connected. The argument of the next theorem is analogous to the proof, which is due to Lovász [BM] of the Brooks theorem.

THEOREM 6.3. *If G is a 3-connected graph, then $\lambda(G) \le \Delta^2 + 2\Delta - 3$.*

Proof. If G is complete, then it is trivial, since it is easy to see that $\lambda(G) = 2\Delta$. Suppose that G is not complete. Then there exist three vertices u, v, w in V such that $\{u,v\}$ and $\{v,w\}$ are in E but $\{u,w\}$ is not in E. Set $v_1 = u$ and $v_2 = w$ and let $v_3, v_4, \ldots, v_\nu = v$ ($\nu = |V|$) be any ordering of the vertices of $V - \{u,w\}$ such that each $v_i, 3 \le i \le \nu - 1$ is adjacent to some v_j with $j > i$; e.g., order the vertices by nonincreasing distance from v in $G - \{u,w\}$. We can now describe an $L(2,1)$-labelling of G: Label v_1 as 0 and v_2 as 1; then successively label $v_3, v_4, \ldots, v_\nu$ with the lowest available label ≥ 0. Each vertex v_i, $1 \le i \le \nu - 1$ is adjacent to at most $\Delta - 1$ vertices v_j with $j < i$. Each such v_j eliminates at most three possible choices for the label at v_i. Furthermore, there are at most $\Delta(\Delta - 1)$ vertices v_k, $k < i$, at distance 2 from v_i, and each such v_k eliminates at most one choice for the label at v_i. It follows that, when its turn comes to be labeled, some label in $[0, \Delta^2 + 2\Delta - 3]$ will be available for v_i. Finally, since $v_\nu = v$ is adjacent to two vertices with labels 0 and 1, there is some label in $[0, \Delta^2 + 2\Delta - 3]$ available for v_ν. □

We can show that the first-fit labelling given in the above proof uses label $\Delta^2 + 2\Delta - 3$ at most once, so it is likely that a more careful argument can improve the bound.

Sakai [S] observed that the bound in Theorem 6.2 can be improved for chordal graphs, which are graphs that contain no induced cycles of length at least 4. The idea

is order the vertices carefully: A chordal graph has an ordering $\{v_1, v_2, \dots\}$ of V such that, for all i, the neighbors of v_i among $\{v_1, \dots, v_{i-1}\}$ form a clique. An analysis of the first-fit coloring for this sequence yields her result.

THEOREM 6.4 (Sakai [S]). *Let G be a chordal graph with maximum degree Δ. Then $\lambda(G) \leq (\Delta+3)^2/4$.*

7. Diameter 2 graphs. We have a better upper bound for a class of graphs that is important in our study, namely, the diameter 2 graphs. The upper bound for this case, Δ^2, is sharp for some Δ.

Now we present the following lemma, which will allow us to prove the result mentioned above and to determine the complexity of the $L(2,1)$-labelling problem in the next section.

LEMMA 7.1. *The following two statements are equivalent:*

(1) *There exists an injection $f : V(G) \to [0, |V|-1]$ such that $|f(x)-f(y)| \geq 2$ for all $\{x,y\} \in E(G)$;*

(2) *G^c contains a Hamilton path.*

Proof. (1) $\Rightarrow$ (2): Let f be an injection defined on V that satisfies the condition in (1). Since f is injective, f^{-1} exists. Order the vertices of V as follows: $v_i = f^{-1}(i)$, $0 \leq i \leq |V|-1$. Then v_i is adjacent to v_{i+1} in G^c for $0 \leq i \leq |V|-1$. Therefore the path $\{v_0, v_1, \dots, v_{|V|-1}\}$ is a Hamilton path of G^c.

(2) $\Rightarrow$ (1): Let $P = \{v_0, v_1, \dots, v_{|V|-1}\}$ be a Hamilton path of G^c. Define the function $f : V(G) \to [0, |V|-1]$ by $f(v_i) = i$, $0 \leq i \leq |V|-1$. Then it is easy to see that f is injective. Let $\{x,y\} \in E(G)$. Then $f(x) = f(v_i) = i$ and $f(y) = f(v_j) = j$ for some i, j with $|i-j| \geq 2$ since x is not adjacent to y in G^c. Hence f is the injection we need. □

To prove Theorem 7.3, we also need the following theorem due to Dirac [D] (see also [BM]).

THEOREM 7.2. *Let G be a simple graph with $|V| \geq 3$ and minimum degree $\delta \geq |V|/2$. Then G is Hamiltonian.*

Now we present our bound for diameter 2 graphs.

THEOREM 7.3. *If G is a graph with diameter 2, then $\lambda(G) \leq \Delta^2$.*

Proof. If $\Delta = 2$, then we can verify the result directly, since, in this case, G is either C_4, C_5 or a path of length 2. Thus assume that $\Delta \geq 3$.

Suppose that $\Delta \geq (|V|-1)/2$. By Theorem 5.1, we have that $\lambda \leq |V| + \chi - 2 \leq 2\Delta + 1 + \Delta - 2 = 3\Delta - 1 < \Delta^2$, since $\Delta \geq 3$.

Now suppose that $\Delta < (|V|-1)/2$. Then $\delta(G^c) \geq |V|/2$. Since diam(G)=2, obviously $|V| \geq 3$. Hence, by Theorem 7.2, G^c is Hamiltonian; i.e., G^c contains a Hamilton path. By Lemma 7.1, there is an injection $f : V \to [0, |V|-1]$ such that $|f(x) - f(y)| \geq 2$, for all $\{x,y\} \in E(G)$. From here, it is easy to see that $f \in L(2,1)(G)$ and $\|f(G)\| = |V|-1$. G is a diameter 2 graph, so $|V| \leq \Delta^2 + 1$. Therefore $\lambda(G) \leq |V| - 1 \leq \Delta^2$. □

Note that the upper bound Δ^2 is the best possible only when $\Delta = 2, 3, 7$, and possibly 57 because a diameter 2 graph with $|V| = \Delta^2 + 1$ can exist only if Δ is one of these numbers (cf. [HS]). When $\Delta = 2$, the graph is C_5; when $\Delta = 3$, it is the *Petersen graph*. For the graph when $\Delta = 7$, it is called the *Hoffman–Singleton graph* (see [HS] or [BM]). Since diam(G)=2, all labels in V must be distinct. Hence $\lambda(G) \geq |V| - 1 = \Delta^2$. On the other hand, by Theorem 7.3, $\lambda \leq \Delta^2$. Thus $\lambda(G) = \Delta^2$ only if $\Delta(G) = 2, 3, 7$, and possibly 57.

According to the proof of Theorem 7.3, if $\Delta \geq 3$, we also know that $\lambda < \Delta^2$ whenever $|V| < \Delta^2+1$. Hence, in general, except for those extremal graphs mentioned above and C_4, whose labelling number also is $\Delta^2(=4)$, $\Delta^2 - 1$ is an upper bound on λ for a diameter 2 graph.

8. Two special classes of graphs. In this section, we will present two classes of graphs with λ that is close to the bound we have in Theorem 6.2.

First, let us give some definitions. We say a graph G is an *incidence graph* of a projective plane $\Pi(n)$ of order n, if $G = (A, B, E)$ is a bipartite graph such that

(1) $|A| = |B| = n^2 + n + 1$,

(2) each $a \in A$ corresponds to a point p_a in $\Pi(n)$ and each $b \in B$ corresponds to a line ℓ_b in $\Pi(n)$, and

(3) $E = \{\{a, b\} : a \in A, b \in B$ such that $p_a \in \ell_b$ in $\Pi(n)\}$.

By the definition of $\Pi(n)$, we know that such G is $(n + 1)$-regular, for every $x, y \in A$, $d_G(x, y) = 2$, and for every $u, v \in B$, $d_G(u, v) = 2$. Also, if $a \in A$, $b \in B$ such that a is not adjacent to b, then $d_G(a, b) = 3$. In [Y] we have the following theorem.

THEOREM 8.1. *If G is the incidence graph of a projective plane of order n, then $\lambda(G) = n^2 + n = \Delta^2 - \Delta$, where $\Delta = n + 1$, the maximum degree of G.*

Before the next theorem, let us recall the definition of the *Galois plane*. Let K be the *Galois field of order* n and let $P = \{(x_1, x_2, x_3) : x_i \in K\}\backslash\{(0, 0, 0)\}$. Define an equivalence relation $\equiv$ on P in the following manner: $(x_1, x_2, x_3) \equiv (y_1, y_2, y_3)$ if and only if there exists $c \in K$, $c \neq 0$ for which $y_1 = cx_1$, $y_2 = cx_2$, $y_3 = cx_3$. Let these equivalence classes be called *points*. The set of all points defined by an equation $a_1x_1 + a_2x_2 + a_3x_3 = 0$, where a_1, a_2, $a_3 \in K$ and not all are zero, will be called a *line*, which is denoted by $[a_1, a_2, a_3]$.

The projective plane determined above will be called a *Galois plane* (*over the coordinate field* $GF(n)$) and will be denoted by $PG_2(n)$ (cf. [K]).

Next, we construct another class of graphs from the Galois plane $PG_2(n)$ (cf. [B]). Let $V(H)$ be the set of points of $PG_2(n)$ and join a point (x, y, z) to a point (x', y', z') if $xx' + yy' + zz' = 0$, i.e., if (x', y', z') lies on the line $[x, y, z]$. We call such a graph H the *polarity graph* of $PG_2(n)$. Then by the properties of $PG_2(n)$, we know that $|V(H)| = n^2 + n + 1$, the maximum degree $\Delta(H) = n + 1$, the minimum degree $\delta(H) = n$ and the diameter is 2 (cf. [B]). Now we present the following theorem from [Y].

THEOREM 8.2. *If H is the polarity graph of the Galois plane, $PG_2(n)$ then $\lambda(H) = n^2 + n = \Delta^2 - \Delta$, where Δ is the maximum degree of H.*

9. The complexity of the $L(2,1)$-labelling problem. It is well known that the coloring problem is an NP-complete problem. Since our $L(2, 1)$-labelling problem is similar to the coloring problem, we may guess it is also NP-complete. In this section, we verify this claim.

We need to consider the following special form of the $L(2, 1)$-labelling problem, where (DL) denotes distance 2 labelling:

(DL)
Instance: Graph $G = (V, E)$ with diameter 2.
Question: Is $\lambda(G) \leq |V|$?

THEOREM 9.1. (DL) *is* NP*-complete.*

Proof. To show that (DL) is NP-complete, we study the following decision problem, where (IDL) denotes injective distance 2 labelling:

(IDL) *Instance*: Graph $G = (V, E)$.
Question: Is there an injection $f : V \to [0, |V| - 1]$ such that $|f(x) - f(y)| \geq 2$ whenever $\{x, y\} \in E$?

In view of Lemma 7.1, the NP-completeness of (IDL) follows as an immediate consequence of the well-known NP-completeness of the Hamilton path problem (HP) (see [GJ]) below:

(HP) *Instance*: Graph $G = (V, E)$,
Question: Is there a Hamilton path in G?

Next, we observe that (DL) is in NP. A graph $G = (V, E)$ can be input in time $O(|V| + |E|)$, and clearly we can verify in polynomial time that G has diameter 2, that a labelling f is in $L(2, 1)(G)$, and that $\|f(G)\| \leq |V|$.

We now show that (DL) is NP-complete by transformation from (IDL) to (DL). Let $G = (V, E)$ be any graph in the instance of (IDL). Construct a graph G' as follows: Add a vertex x to V and let x be adjacent to every vertex of V, i.e., $G' = (V', E')$, where $V' = V \cup \{x\}$ and $E' = E \cup \{\{x, v\} : \text{ for all } v \in V\}$. Then $|V'| = |V| + 1$ and diam(G')=2.

The NP-completeness of (DL) then follows from the NP-completeness of (IDL) and from the following claim.

Claim. There is an injection $f : V(G) \to [0, |V| - 1]$ such that $|f(u) - f(v)| \geq 2$ for every $\{u, v\} \in E(G)$ if and only if $\lambda(G') \leq |V'|$.

Proof of claim. Suppose that there exists an injective function f defined on V that satisfies the condition above. Define $g(v) = f(v)$ for all $v \in V$ and $g(x) = |V| + 1 = |V'|$. Then easily $g \in L(2, 1)(G')$ and $\|g(G')\| = |V| + 1 = |V'|$. Hence $\lambda(G') \leq |V'|$.

Conversely, suppose that $\lambda(G') \leq |V'|$, i.e., there is a g in $L(2, 1)(G')$ such that $\|g(G')\| \leq |V| + 1$. Suppose that $g(x) \neq 0$ or $|V| + 1$. By the property of $L(2, 1)(G')$, there is no v in V such that $g(v) = g(x) + 1$ or $g(x) - 1$. This implies that we must use $|V| + 3$ numbers to label V', which is a contradiction, since $\|g(G')\| \leq |V| + 1$ and all labels are distinct.

Hence $g(x)$ is either 0 or $|V| + 1$. If $g(x) = |V| + 1$, then restricting g to V gives the desired injection f. Similarly, if $g(x) = 0$, then restricting $g - 2$ to V gives f. □

10. Further research. Inspired by the more general proximity-interference problems, a more general context would be to study labellings f, where N is a positive integer and $m_1 \geq m_2 \geq \cdots \geq m_N > 0$ are given numbers. We require that $|f(x) - f(y)| \geq m_i$ if $d_G(x, y) = i$, $1 \leq i \leq N$. If $N = 1$ and $m_1 = 1$, then we have ordinary graph coloring. If $N = 2$, $m_1 = 2$, and $m_2 = 1$, then it is the $L(2, 1)$-labelling. If $N = 2$, $m_1 = 1 = m_2$, then we have the $L(1, 1)$-labelling, which has been studied in [Y].

Recall from the proof of Theorem 7.3 that, if $\Delta \geq 3$ and $\Delta \geq (|V| - 1)/2$, then we have $\lambda \leq \Delta^2$, regardless of whether G has diameter 2. Therefore it is reasonable to propose the following conjecture.

CONJECTURE 10.1. *For any graph G with maximum degree $\Delta \geq 2$, $\lambda(G) \leq \Delta^2$.*

This conjecture holds for $\Delta = 2$ in view of Propositions 3.1 and 3.2.

In §9 we proved that the problem (DL) is NP-complete, but λ was not a fixed value there. Consider the following decision problem for fixed λ, where (DLk) denotes distance 2 labelling with upper bound k:

(DLk)
Instance: Graph $G = (V, E)$.
Question: Is $\lambda(G) \leq k$?

CONJECTURE 10.2. *For $k \geq 4$,* (DLk) *is* NP*-complete.*

For nontrivial trees T, we saw in §4 that $\lambda(T)$ is either $\Delta + 1$ or $\Delta + 2$. It seems to be quite difficult to determine which of the two values holds, somewhat analogous to the situation for edge-colorings of graphs. Consider this decision problem for trees:

(TREE)
Instance: Tree $T = (V, E)$.
Question: Is $\lambda(T) = \Delta(T) + 1$?

CONJECTURE 10.3. *The problem* (TREE) *is* NP*-complete.*

Acknowledgments. The authors thank Zoltán Füredi and Denise Sakai for editorial suggestions and for interest in this project.

REFERENCES

[B] B. BOLLOBÁS, *Extremal Graph Theory*, Academic Press, London, 1978.
[BM] J. A. BONDY AND U. S. R. MURTY, *Graph Theory with Applications*, North-Holland, New York, 1976.
[Br] R. L. BROOKS, *On coloring the nodes of a network*, Proc. Cambridge Philos. Soc., 37 (1941), pp. 194–197.
[CR] M. B. COZZENS AND F. S. ROBERTS, *T-colorings of graphs and the channel assignment problem*, Congr. Numer., 35 (1982), pp. 191–208.
[CW] M. B. COZZENS AND D. I. WANG, *The general channel assignment problem*, Congr. Numer., 41 (1984), pp. 115–129.
[D] G. A. DIRAC, *Some theorems on abstract graphs*, Proc. London Math. Soc., 2 (1952), pp. 69–81.
[FGK] Z. FÜREDI, J. R. GRIGGS, AND D. J. KLEITMAN, *Pair labellings with given distance*, SIAM J. Discrete Math., 2 (1989), pp. 491–499.
[G] M. C. GOLUMBIC, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.
[GJ] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of* NP*-Completeness*, W. H. Freeman, New York, 1979.
[H] W. K. HALE, *Frequency assignment: Theory and applications*, Proc. IEEE, 68 (1980), pp. 1497–1514.
[HS] A. J. HOFFMAN AND R. R. SINGLETON, *On Moore graphs with diameters 2 and 3*, IBM J. Res. Develop., 4 (1960), pp. 497–504.
[J] K. JONAS, private communication, 1991.
[K] F. KÁRTESZI, *Introduction to Finite Geometries*, North-Holland, New York, 1976.
[R1] A. RAYCHAUDHURI, *Intersection assignment, T-coloring and powers of graphs*, Ph.D. thesis, Department of Mathematics, Rutgers University, New Brunswick, NJ, 1985.
[R2] ———, *Further results on T-coloring and frequency assignment problems*, SIAM J. Discrete Math., submitted.
[Ro1] F. S. ROBERTS, *T-colorings of graphs: Recent results and open problems*, Research Report RRR 7-86, RUTCOR, Rutgers University, New Brunswick, NJ, 1986.
[Ro2] ———, private communication, 1988.

[Ro3] F. S. ROBERTS, *From garbage to rainbows: Generalizations of graph colorings and their applications*, in Proc. of the Sixth International Conference on the Theory and Applications of Graphs, Y. Alavi, G. Chartrand, O. R. Oellermann, and A. J. Schwenk, eds., John Wiley, New York, 1989.

[S] D. SAKAI, private communication, 1991.

[T] B. TESMAN, *T-colorings, list T-colorings, and set T-colorings of graphs*, Ph.D. thesis, Department of Mathematics, Rutgers University, New Brunswick, NJ, 1989.

[Y] R. K. YEH, *Labeling graphs with a condition at distance two*, Ph.D. thesis, Department of Mathematics, University of South Carolina, Columbia, SC, 1990.

SIAM J. DISC. MATH.
Vol. 5, No. 4, pp. 596–603, November 1992
© 1992 Society for Industrial and Applied Mathematics
015

IMPLICIT REPRESENTATION OF GRAPHS*

SAMPATH KANNAN†, MONI NAOR‡, AND STEVEN RUDICH§

Abstract. How to represent a graph in memory is a fundamental data structuring question. In the usual representations of an n-vertex graph, the names of the vertices (i.e., integers from 1 to n) betray nothing about the graph itself. Indeed, the names (or labels) on the n vertices are just $\log n$ bit place holders to allow data on the edges to encode the structure of the graph. In this scenario, there is no such waste. By assigning $O(\log n)$ bit labels to the vertices, the structure of the graph is completely encoded, so that, given the labels of two vertices, one can test if they are adjacent in time linear in the size of the labels. Furthermore, given an arbitrary original labeling of the vertices, structure coding labels are found (as above) that are no more than a small constant factor larger than the original labels. These notions are intimately related to vertex-induced universal graphs of polynomial size. For example, planar graphs can be labeled with structure coding labels of size $< 4 \log n$, which implies the existence of a graph with n^4 vertices that contains all n-vertex planar graphs as vertex-induced subgraphs. The theorems on finite graphs extend to a theorem about the constrained labeling of infinite graphs.

Key words. graph representation, universal graphs, arboricity, intersection graphs, data structures

AMS(MOS) subject classifications. 05C75 68P05 68R10

1. Introduction.

We will consider graphs $G = (V, E)$ with vertex set V and edge set $E \subseteq (V \times V)$. The graphs we consider will not have any self-loops, i.e., edges of the form (v, v). We will also disallow parallel edges between two vertices, as the definition of E, above, indicates.

DEFINITION 1. A *vertex-induced subgraph* or simply an *induced subgraph* G' of G is a vertex set $V' \subseteq V$ together with the edge set $E' = E \cap (V' \times V')$.

DEFINITION 2. An *edge-induced subgraph* or simply a *subgraph* is a subset $V' \subseteq V$ together with an edge set $E' \subseteq E \cap (V' \times V')$.

Consider the following problems.

Problem 1. Label the vertices of an n-vertex tree with labels that are $O(\log n)$ bits long such that, given the name of two vertices, we can determine adjacency quickly.

Solution 1. Root the tree. Prelabel each vertex with arbitrary, distinct, positive integers. Let the label on each vertex be its prelabel appended with the prelabel of its parent. The same procedure works for infinite trees with infinite degrees as well.

Problem 2. Given a tree prelabeled with distinct positive integers (imagine an adversary choosing the prelabels), label the vertex prelabeled i with $O(\log i)$ bits, so that, given two vertices, we can determine adjacency quickly.

Solution 2. Root the tree. Each vertex will receive a two-part label. For the first part of its label, each vertex takes on the prelabel of its smallest child or itself, whichever is smaller. There is also a flag bit in the first part to indicate if the prelabel remembered is its own or a child's. The second part of each vertex's label will be the first part of its parent's label. A vertex with prelabel i is labeled with at most $2(\log i + 1)$ bits. Given any two vertices, there is a linear time procedure to tell

* Received by the editors July 30, 1990; accepted for publication (in revised form) January 21, 1992. The research of this paper was supported by National Science Foundation grant DCR-85-13926. A preliminary version of this paper appeared in the Proceedings of the 20th Association for Computing Machinery Symposium Theory of Computing, 1988, Chicago, Illinois.

† Computer Science Department, University of Arizona, Tucson, Arizona 85721.

‡ IBM Research Division, Almaden Research Center, San Jose, California 95120.

§ Computer Science Department, Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213.

adjacency. Two vertices are adjacent if and only if the first part of one vertex's label is the same as the second part of the other vertex's label.

Problem 3. Give an efficient construction of an $O(n^2)$ graph that contains all n-vertex trees as vertex-induced subgraphs.

Solution 3. The following works for forests as well: Let the vertices be the ordered pairs (i, j), where $1 \leq i, j \leq n$. Place an edge between (i, j) and (i', j'), if and only if $i = j'$ or $j = i'$.

For a wide variety of graphs, the above three problems are essentially equivalent. We will formalize the above notions and present them in a more general setting.

2. Definitions. DEFINITION 3. A family F of finite graphs has a k-labeling scheme if there is a polynomial time Turing machine T and a function f, which labels the vertices of each graph G in F with distinct labels of no more than $k \log n$ bits (n is the number of vertices of G) such that, given two vertex labels of a graph G in F, T will correctly decide adjacency of the corresponding vertices in G.

If a family F has a k-labeling scheme for some k, we say that it has a *labeling scheme.*

For a family of graphs to have a labeling scheme is clearly a desirable property. It enables representing graphs that are members of the family implicitly, just by having available the vertex names. For instance, in a distributed system, the graph need not be stored in one place. Whenever a processor needs to determine adjacency, however, it can do so only by looking at the names of the two vertices. If the vertices of a graph were ordered by the frequency of their usage, we would want to assign the more frequent vertices a shorter label. In general, we could have other constraints on the size of the vertex labels. Hence we define constrained labeling.

DEFINITION 4. A family of graphs F has a *constrained k-labeling scheme* if for all $G \in F, |G| = n$, for all p such that p assigns distinct positive integers to the vertices of G, we can associate $k\lceil \log p(v) \rceil$ bits with each vertex v and use these labels to decide adjacency efficiently.

As mentioned above, labeling schemes are related to vertex induced universal graphs, which we define below.

DEFINITION 5. Given a finite set of graphs S, a graph G is *vertex-induced universal* for S if every graph in S is a vertex induced subgraph of G. We say a family F has universal graphs of size $g(n)$ if, for every n, there is a graph of size less than or equal to $g(n)$, which is vertex-induced universal for the set of all graphs in F with n or fewer vertices.

It is possible to define the *edge-induced* universal graph of a set of graphs S as a graph G that contains each graph in S as an edge-induced subgraph. However, in our paper, we focus on vertex-induced universal graphs. So we refer to vertex-induced universal graphs simply as *universal graphs.*

The remainder of the paper deals with the relationship among the three concepts defined above. In §3 we give labeling schemes for a wide variety of graph families. Section 4 discusses constrained labeling and shows that, under a very weak hypothesis, all families that have labeling schemes also have constrained labeling schemes. Section 5 deals with the implications of labeling on the construction of universal graphs. Sections 6 and 7 outline some of the related work and open problems.

3. Labelable families. In this section, we show that a wide variety of families of graph have labeling schemes. We start by showing that trees and transitive closures of trees have labeling schemes. These two schemes will be very useful as building blocks in labeling schemes we give for other families. We then give a labeling scheme

for graphs of bounded arboricity, which can be viewed as "uniformly sparse" graphs. This yields a 4-labeling scheme for planar graphs. Next, we present a general technique that applies to many families of intersection graphs. Finally, we give a labeling scheme for c-decomposable graphs, i.e., graphs that have small separators, which decompose the graph.

Not all families of graphs are labelable. Since we allow only $O(n \log n)$ bits to represent an n-vertex graph, we can represent at most $2^{O(n \log n)}$ graphs. Hence we have the following proposition.

PROPOSITION 1. *A family of graphs that contains more than* $2^{\Omega(n \log n)}$ *n-vertex graphs cannot be labeled.*

As a corollary, bipartite graphs and chordal graphs are not labelable.

3.1. Trees and transitive closures of trees. In this section, we give a 2-labeling scheme for trees and transitive closures of trees. These two schemes will be used extensively as building blocks in the rest of §3.

Finite trees. We give a 2-labeling scheme for trees (not necessarily bounded degree): Given an n-vertex tree, arbitrarily choose a root, and arbitrarily prelabel the vertices with the integers from 1 to n. For each vertex, concatenate the prelabel of its parent (if any) to its own prelabel. The new vertex labels have no more than $2 \log n$ bits. We can now quickly decide parenthood given two vertex labels by checking if the first half of one is the second half of the other. Thus we can decide adjacency. This also works for finite forests.

Transitive closures of trees. Often we are interested in the ancestorhood relation in trees rather than the adjacency relation. Let F be the family of transitive closures of rooted finite trees. The adjacency relation in F is the same as the ancestorhood relation in rooted finite trees. We can find a 2-labeling scheme for F: Let T be a tree and let T' be the transitive closure of T. The label we assign each vertex is an interval on the integer line. We traverse T in post-order. To each vertex, we assign the interval between its smallest numbered descendant and its largest numbered descendant. A vertex u is an ancestor of v if and only if the interval for u contains the interval for v.

3.2. Sparse graphs or graphs with bounded arboricity. Let G be a graph and let H range over all possible vertex induced subgraphs of G. Then the *arboricity* of G is defined to be

$$\max_H \frac{|E(H)|}{|V(H)| - 1},$$

where $|E(H)|$ is the number of edges in H and $|V(H)|$ is the number of vertices in H. Nash-Williams [11] showed that the edges of a graph G with arboricity k can be decomposed into k forests. This means that there is a $k+1$ labeling scheme for graphs of arboricity k. Prelabel the vertices arbitrarily with distinct integers from 1 to n. Concatenate to each vertex label, the label of its parent in each of the k forests. The ordered $(k+1)$-tuple is then the label of the vertex. To decide adjacency of two vertices, just check if one is the parent of the other in any of the forests. It is clear that adjacency can be tested efficiently. Using flow techniques, Picard and Queyranne [13] have shown that the forest decomposition of a graph and thus its labels can be computed efficiently.

Several families fall into this category and hence have labeling schemes; see below:

1. Graphs of bounded degree d clearly have arboricity bounded by $\lfloor d/2 \rfloor + 1$;
2. Graphs of bounded genus g fall in this category, since they have at most $6(g-1) + 3n$ edges and hence can be labeled succinctly;

3. In the special case of planar graphs that have arboricity 3, this technique yields a 4-labeling scheme.

3.3. Intersection graphs. In this section, we consider many families of graphs, which are characterized as intersection graphs for certain types of sets. The general method used is to try to find a succinct representation for the sets representing the vertices.

DEFINITION 6. An interval graph is a graph where each vertex can be represented by an interval on the real line such that two vertices are adjacent if and only if the interval representing one vertex intersects the interval representing the other.

The technique for labeling interval graphs is similar to the one for labeling transitive closures of trees. This yields a 2-labeling scheme. Adjacency can be tested in linear time in the length of the labels. In this case, the definition of the sets gives us a labeling scheme. This is so for a number of families: *circle graphs, circular arc graphs, permutation graphs, graphs with bounded interval number.* See Golumbic for definitions of these families [8].

Path graphs. Path graphs are graphs where each vertex is represented by a path in a tree. Two vertices are adjacent if and only if the paths representing them intersect. There are several kinds of path graphs mentioned in the literature. See Monma and Wei [10] for details. We will show how to label path graphs where the paths are undirected and two paths intersect if they have a vertex in common. Similar methods will work for all path graph families, provided the path representation is given. To label path graphs, we first label the transitive closure of the tree containing all the paths. Now we can decide ancestorhood in the tree quickly.

To label the vertices of the path graph, we concatenate the labels of the beginning, the apex, and the end of the path representing that vertex. Here the apex of a path is the vertex along the path closest to the root. To test adjacency of two vertices, we must test if the apex of one vertex is sandwiched between the apex and an end vertex of the other path. This requires, at most, six adjacency tests in a transitive closure of a tree. Finding the path representation and thus the labeling of a graph can be done efficiently [7]. It is interesting to note that the closely related family of *chordal graphs,* which can be characterized as the intersection of subtrees in a tree, cannot be labeled.

3.4. c-decomposable graphs. DEFINITION 7. A graph G is c-decomposable if, for all subgraphs H with more than c vertices, there exist c vertices such that their removal causes H to be disconnected with no component containing more than $2|H|/3$ vertices [6].

To label a c-decomposable graph G, we construct a tree decomposition T of G in the following manner. Let S be a c-separator for G. Then S is assigned to the root of T. If H_1 and H_2 are the two components obtained by removing S from G, the left subtree will be the tree for H_1, and the right subtree will be the tree for H_2.

Every vertex v of G occurs in a vertex $t(v)$ in T. T is binary, has depth no greater than $\log_{3/2} n$, and each vertex of T is assigned at most c vertices of G. Assume an arbitrary ordering among the vertices of the graph assigned to the same vertex of T. Two vertices in G can be adjacent only if they are assigned to two vertices of T such that one is an ancestor of the other.

To label G, for each vertex v, we remember the following:

1. The path in the tree from the root to $t(v)$, which is of length at most $\log_{3/2} n$,
2. The rank of v in $t(v)$, which is an arbitrary number from 1 to c,

3. For each vertex of the tree s, along the path from the root to $t(v)$, a c-bit vector giving adjacency information with all the vertices in G assigned to s. This is at most $c \cdot \log_{3/2} n$ bits.

Given two vertices u and v of G, we determine if $t(u)$ is an ancestor of $t(v)$ (or vice versa). If so, we determine the depth i of the ancestor, say u. The ith c-bit vector in v's label has a bit at the position corresponding to u's rank, which tells if u and v are adjacent. This is an efficient procedure to determine adjacency.

4. Constrained labeling schemes. In this section, we prove that any family of graphs that has a labeling scheme also has a constrained labeling scheme as long as it is closed under vertex deletion. For example, let F be the family of graphs bounded by degree k. A constrained k-labeling scheme for F is obtained by associating with each vertex its own prelabel concatenated with the prelabels of its neighbors with smaller prelabels. We have already seen a constrained 2-labeling scheme for trees. Since graphs of bounded arboricity can be decomposed into a constant number of forests, we also have a constrained labeling scheme for these graphs. In fact, all the families of graphs for which we have given labeling schemes also have constrained labeling schemes.

THEOREM 1. *A family of finite graphs that is closed under vertex deletion has a labeling scheme if and only if it has a constrained labeling scheme.*

Proof. We will call a family F of graphs *hereditary* if it is closed under vertex-induced subgraphs, i.e., under vertex deletion. Let $G = (V, E)$ be a graph belonging to a family F and let $S = (V', E')$ be some subgraph of G. An *extension* X of S is defined to be a subset of $V - V'$, such that, for all $x, y \in X$, the neighbor sets of x and y in G intersect V' in distinct sets.

For a family F, we define an *extension function* f_F as follows: $f_F(n)$ is the cardinality of the largest extension of any n-vertex, vertex-induced subgraph of any graph $G \in F$. With this definition, the proof of the theorem reduces to the proof of the following two lemmas.

LEMMA 1. *If F is hereditary, k_1-labelable, and $f_F(n) \in O(n^{k_2})$, then F is constrained $4k_1k_2$-labelable. Note that the constant is independent of the prelabeling function and is entirely determined by graph-theoretic properties of the family F.*

Proof. Let $G \in F$, a k_1-labelable family. Let $p(v)$ denote the prelabel associated with vertex v. Let G_i denote the induced subgraph on the vertex set, $\{v : 2^{2^{i-1}} \leq p(v) < 2^{2^i}\}$. Define G_0 to be the singleton set consisting of the vertex with prelabel 1. The idea is that each vertex in G_j will remember its neighbors in G_i for all $i \leq j$. In fact, the label for a vertex in G_j will be a $(j+1)$-tuple, the ith component of which contains information about neighborhood with vertices in G_i, $i \in [0, 1, \ldots, j]$. Starting with zero, we consider the graphs G_i in turn. We label G_i together with a maximal extension E_i. The labels assigned here will be the ith component of the vertex labels. For any vertex v, there is a vertex $u \in E_i$ such that v and u have the same neighborhood in G_i, since E_i is maximal. The ith component of v's label will then be the same as the ith component of u's label. We now delete the vertices of G_i and proceed to G_{i+1}.

It remains to prove that the labels assigned above satisfy the requirements of a constrained labeling scheme. The ith component of a label comes from the labeling of G_i with its extension. If G_i has n vertices, then, by our hypothesis on F, the maximal extension contains at most n^{k_2} vertices. Labeling a graph in F with $n + n^{k_2}$ vertices requires $k_1(k_2 + \epsilon) \log n$ bits, where $\epsilon \to 0$ as $n \to \infty$, provided that $k_2 \geq 1$. By the definition of the G_i, n is less than 2^{2^i}. Hence the ith component of a label contains

approximately $k_1k_2 2^i$ bits. Hence a vertex in G_j has in all $\sum_{i=0}^{j} k_1k_2 2^i$ bits. This sum is bounded by $k_1k_2 2^{j+1}$. The smallest prelabel in G_j, however, is $2^{2^{j-1}}$, and the log of this value is 2^{j-1}. Hence the label size is bounded by $4k_1k_2$ times the size of the prelabel.

Finally, these labels can be used to determine adjacency efficiently. Given two labels, let the vertex with the smaller prelabel belong to G_i. Then adjacency can be tested by looking at the ith components of both labels. □

Conversely, the following lemma shows that, if the extension function is superpolynomial, then the family of graphs does not even have a labeling scheme.

LEMMA 2. *If F is hereditary and the extension function is superpolynomial, then F is not labelable.*

Proof. Let F be a family of graphs with superpolynomial extension. Let $f_F(n)$ be $n^{\alpha(n)}$, where $\alpha(n) \to \infty$ as $n \to \infty$. We have a sequence of graphs, $G_1, G_2, \ldots$ such that G_n contains a subgraph H_n with n vertices, and H_n has an extension of cardinality at least $n^{\alpha(n)}$. We will prove that there are more than $2^{O(n \log n)}$ n vertex graphs in F, thereby proving that F cannot be labeled.

To H_n, we can adjoin n vertices arbitrarily chosen from the extension. The induced subgraph on these $2n$ vertices is in F. We can choose the n vertices from the extension in

$$\begin{pmatrix} n^{\alpha(n)} \\ n \end{pmatrix}$$

different ways. This is approximately $n^{n\alpha(n)}$ choices of graphs. Some of the graphs we obtain may be isomorphic. However, each isomorphic copy can occur at most $\binom{2n}{n}n!$ times, since fixing the vertices of H_n determines the other vertices as well. Thus, in all, we obtain approximately $n^{n\alpha(n)}/(2n)^n$ distinct graphs with $2n$ vertices. This number is $n^{\Omega(n\alpha(n))}$, which is $2^{\Omega(n \log n\alpha(n))}$. □

This concludes the proof of Theorem 1. □

As a corollary, we have constrained labeling schemes for all families we have discussed in §2. Theorem 1 has consequences for the labeling of infinite graphs.

THEOREM 2. *Any infinite graph with the property that the family consisting of all its finite, vertex-induced subgraphs has a labeling scheme, has a constrained labeling scheme.*

Proof. Let G be an infinite graph and let F be the family of all finite, vertex-induced subgraphs of G. Clearly, F is hereditary. Therefore, by Theorem 1, F has a constrained labeling scheme. Consider the sequence S_0 of subgraphs of G, $G_1, G_2, \ldots$, where G_i is the induced subgraph on all vertices with prelabels $\leq i$. We have constrained labelings for all the G_i; any vertex has a finite prelabel, and therefore only finitely many possibilities for a label. Thus the vertex with prelabel 1 is labeled with the same label l_1 infinitely often in S_0. Define S_1 to be the infinite subsequence where vertex 1 gets label l_1. Vertex 1 is assigned the label l_1. Continuing in this manner, for the vertex prelabeled i, we can find a subsequence in S_{i-1}, where it is labeled with some l_i infinitely often. Vertex i gets the label l_i, and S_i is the corresponding subsequence of S_{i-1}. □

COROLLARY 1. *The infinite version of all the classes discussed above has a constrained labeling scheme.*

5. Labeling schemes and universal graphs. Labeling schemes and vertex-induced universal graphs are closely related.

THEOREM 3. *If a family F has a k-labeling scheme, then it has universal graphs of size n^k constructible in polynomial time.*

Proof. Consider the polynomial time Turing machine T to determine adjacency given two vertex labels of a graph in F. Form the graph U with vertices labeled from 1 to n^k, placing an edge between a and b if and only if $T(a, b)$ says "adjacent." U must contain all graphs in F with n or fewer vertices, since all these graphs receive labels no greater than n^k.

Note that we can determine adjacency in U in time polynomial in the length of the vertex names. Hence families with labeling schemes have efficiently constructible universal graphs. Furthermore, the labels on the vertices of the universal graph make the embedding of a graph in the universal graph easy to find. To embed a graph G in its universal graph, all that is required is to label G. The labels then give the information about the embedding. As a nontrivial example, planar graphs have a 4-labeling scheme and hence have n^4-sized universal graphs. □

6. Related work. Several authors have worked on related problems. Breuer [3] and Breuer and Folkman [4] considered the problem of labeling vertices such that adjacency would be determined by the Hamming distance of the labels. This is a restricted labeling scheme. They prove that this can be done for any graph; however, the length of the labels could be very large compared to $\log n$.

Turan [15] considered the problem of representing a graph as succinctly as possible. The representation is global, however; it gives an efficient representation for the whole graph without necessarily partitioning the information into small chunks for each vertex. In this context, it is clear that our labeling schemes always yield $O(n \log n)$-bit representations of all graphs that belong to labelable families. Furthermore, from the representation of any graph, it is easy to derive the representation of any of its vertex induced subgraphs.

Fredrickson and Janardan [6], Santoro and Khatib [14], and Peleg and Upfal [12] considered the question of storing routing information at the vertices of a packet switching network so as to compute near-optimal routes. Again, they look at the overall storage requirements rather than the requirements on a per-vertex basis. This work has recently been extended to limit the amount of storage in every processor [1].

Many papers have considered the question of universal graphs for a family of graphs. Most of these papers have focused on *edge-induced universal graphs*, which have often just been called *universal graphs*. In this paper, we reserve the term "universal graph" for vertex-induced universal graphs, and we refer to edge-induced universal graphs explicitly. A graph G is edge-induced universal for a set of graphs S if every graph in S is a subgraph of G.

The important issue in constructing edge-induced universal graphs is minimizing the number of edges. Bhatt et al. [2] consider the question of constructing n-vertex edge-induced universal graphs for n-vertex bounded-degree trees and planar graphs. They show that, for bounded-degree trees, there is an edge-induced universal graph with cn edges, where c is a constant depending only on the degree bound. For bounded degree planar graphs, they show an edge-induced universal graph with $O(n \log n)$ edges.

Recently, Chung [5] has shown how to construct an edge-induced universal graph for n vertex trees having n vertices and $O(n \log n)$ edges. For planar graphs, [5] gives a bound of $O(n^3/2)$ on the number of edges in the edge-induced universal graph. Reference [5] also combines these results with the techniques of our paper to improve the bounds on the sizes of (vertex-induced) universal graphs for trees and planar

graphs. Specifically, [5] proves that there is a graph with $O(n \log n)$ vertices and $O(n^2)$ edges that contains all n-vertex trees as induced subgraphs and that there is a graph with $O((n \log n)^3)$ vertices and $O(n^6)$ edges that contains all planar graphs on n vertices.

7. Open problems. We know from Proposition 1 that any labelable family has at most $2^{O(n \log n)}$ n-vertex graphs. The natural question is whether all such families that are hereditary have a labeling scheme.

Although this paper produces "small" universal graphs for various families, the question of matching upper and lower bounds for the sizes of universal graphs for these families still remains open.

All our results for labeling graphs are "static." It is often useful to consider dynamically changing graphs where vertices get added and deleted. Can labeling be done in such a manner as to permit quick updates of the labels?

Can the ideas for labeling graphs be extended to hypergraphs? If this could be done dynamically as well, it would have applications to databases.

Acknowledgments. The authors thank Manuel Blum for his many suggestions to the paper. Russell Impagliazzo provided many insights. Helpful remarks on the research and on the paper were received from Ronitt Rubinfeld, Raimund Seidel, and Alejandro Schaffer. The third author especially thanks Ashok Chandra, Larry Carter, Don Coppersmith, and the other researchers at IBM Yorktown for crucial discussions in the embryonic stage of this research.

REFERENCES

[1] B. Awerbuch, A. Bar-Noy, N. Linial, and D. Peleg, *Compact distributed data structures for adaptive routing*, in Proc. 21st ACM Sympos. Theory of Computing, 1989, pp. 479–489.

[2] S. N. Bhatt, F. R. K. Chung, F. T. Leighton, and A. Rosenberg, *Universal graphs for bounded-degree trees and planar graphs*, SIAM J. Discrete Math., 2 (1989), pp. 145–155.

[3] M. Breuer, *Coding vertexes of a graph*, IEEE Trans. Inform. Theory, 12 (1966), pp. 148–153.

[4] M. Breuer and J. Folkman, *An unexpected result on coding vertices of a graph*, J. Math. Anal. Appl., 20 (1967), pp. 583–600.

[5] F. R. K. Chung, *Universal graphs and induced universal graphs*, J. Graph Theory, 14 (1990), pp. 443–454.

[6] G. N. Fredrickson and R. Janardan, *Separator-based strategies for efficient message routing*, in Proc. 27th IEEE Sympos. on Foundation of Computer Science, 1986, Toronto, Ontario, Canada, pp. 428–437.

[7] F. Gavril, *A recognition algorithm for the intersection of paths in trees*, Discrete Math., 23 (1978), pp. 211–227.

[8] M. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[9] S. Kannan, M. Naor, and S. Rudich, *Implicit representation of graphs*, in Proc. 20th ACM Sympos. Theory of Computing, 1988, Chicago, Illinois, pp. 334–343.

[10] C. L. Monma and V. K. Wei, *Intersection graphs of paths in a tree*, J. Combin. Theory B, 41 (1986), pp. 141–181.

[11] C. Nash-Williams, *Edge-disjoint spanning trees of finite graphs*, J. London Math. Soc., 36 (1961), pp. 445–450.

[12] D. Peleg and E. Upfal, *A tradeoff between space and efficiency for routing tables*, J. Assoc. Comput. Mach., 36 (1989), pp. 510–530.

[13] J. C. Picard and M. Queyranne, *A network flow solution to some non-linear* 0-1 *programming problems, with applications to graph theory*, Networks, 12 (1982), pp. 141–160.

[14] N. Santoro and R. Khatib, *Labeling and implicit routing in networks*, Comput. J., 28 (1985), pp. 5–8.

[15] G. Turan, *Succinct representation of graphs*, Discrete Appl. Math., 8 (1984), pp. 289–294.