SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 1–27

IMPROVED ALGORITHMS AND ANALYSIS FOR SECRETARY PROBLEMS AND GENERALIZATIONS*

MIKLOS AJTAI†, NIMROD MEGIDDO‡, AND ORLI WAARTS§

Abstract. In the classical secretary problem, n objects from an ordered set arrive in random order, and one has to accept k of them so that the final decision about each object is made only on the basis of its rank relative to the ones already seen. Variants of the problem depend on the goal: either maximize the probability of accepting the best k objects, or minimize the expectation of the sum of the ranks (or powers of ranks) of the accepted objects. The problem and its generalizations are at the core of tasks with a large data set, in which it may be impractical to backtrack and select previous choices.

Optimal algorithms for the special case of $k = 1$ are well known. Partial solutions for the first variant with general k are also known. In contrast, an explicit solution for the second variant with general k has not been known. It seems that the fact that the expected sum of powers of the ranks of selected items is bounded as n tends to infinity has been known to follow from standard results. We derive our results by obtaining explicit algorithms. For each $z \geq 1$, the resulting expected sum of the zth powers of the ranks of the selected objects is at most $k^{z+1}/(z+1) + C(z) \cdot k^{z+0.5} \log k$, where $\log k \equiv \max\{1, \log_2 k\}$, whereas the best possible value at all is $k^{z+1}/(z+1) + O(k^z)$. Our methods are very intuitive and apply to some generalizations. We also derive a lower bound on the trade-off between the probability of selecting the best object and the expected rank of the selected object.

Key words. dynamic programming, optimal stopping, expected rank maximization

AMS subject classifications. 65C50, 90C39

PII. S0895480195290017

1. Introduction.

In the classical *secretary* problem, n items or options are presented one by one in random order (i.e., all $n!$ possible orders being equally likely). If we could observe them all, we could rank them totally with no ties, from best (rank 1) to worst (rank n). However, when the ith object appears, we can observe only its rank relative to the previous $i-1$ objects; the relative rank is equal to 1 plus the number of the predecessors of i which are preferred to i. We must accept or reject each object, irrevocably, on the basis of its rank relative to the objects already seen, and we are required to select k objects. The problem has two main variants. In the first, the goal is to maximize the probability of obtaining the best k objects. In the second, the goal is to minimize the expectation of the sum of the ranks of the selected objects or, more generally, for a given positive integer z, minimize the expectation of the sum of the zth powers of the ranks.

*Received by the editors August 7, 1995; accepted for publication (in revised form) February 28, 2000; published electronically December 28, 2000.

http://www.siam.org/journals/sidma/14-1/29001.html

†IBM Research Division, Almaden Research Center, 650 Harry Road, San Jose, CA 95120-6099 (ajtai@almaden.ibm.com). The research of this author was supported in part by ONR contract N-00014-94-C-0007.

‡IBM Research Division, Almaden Research Center, Department K53/B2/, 650 Harry Road, San Jose, CA 95120-6099 (megiddo@almaden.ibm.com) and School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel. The research of this author was supported in part by ONR contract N-00014-94-C-0007.

§Computer Science Division, University of California, Soda Hall, Berkeley, CA 94720 (waarts@cs.berkeley.edu). The research of this author was supported in part by ONR contract N-00014-94-C-0007 and an NSF postdoctoral fellowship. Part of this work was done while this author was at the IBM Almaden Research Center.

Solutions to the classical problem apply also in a variety of more general situations. Examples include the following cases. (i) Objects are drawn from some probability distribution; the interesting feature of this variant is that the decisions of the algorithms may be based not only on the relative rank of the item but also on an absolute "grade" that the item receives. (ii) The number of objects is not known in advance. (iii) Objects arrive at random times. (iv) Some limited backtracking is allowed: objects that were rejected may be recalled. (v) The acceptance algorithm has limited memory. Solutions to the classical problem also apply to combinations of these situations. In addition to providing intuition and upper and lower bounds for the above important generalizations of the problem, solutions to the classical problem also provide in many cases very good approximations, or even exact solutions (see [6, 15, 16] for survey and also [10]). Our methods can also be directly extended to apply for these generalizations.

The obvious application to choosing a best applicant for a job gives the problem its common name, although the problem (and our results) has a number of other applications in computer science. For any problem with a very large data set, it may be impractical to backtrack and select previous choices such as, in the context of data mining, selecting records with best fit to requirements or retrieving images from digital libraries. In such applications, limited backtracking may be possible, and, in fact, this is one of the generalizations mentioned above. Another important application is when one needs to choose an appropriate sample from a population for the purpose of some study. In other applications the items may be jobs for scheduling, opportunities for investment, objects for fellowships, etc.

1.1. Background and intuition. The problem has been extensively studied in the probability and statistics literature (see [6, 15, 16] for surveys and also [12]).

The case of $k = 1$. Let us first review the case of $k = 1$, i.e., only one object has to be selected. Since the observer cannot go back and choose a previously presented object which, in retrospect, turns out to be the best, it clearly has to balance the risk of stopping too soon and accepting an apparently desirable object when an even better one might still arrive, against the risk of waiting for too long and then finding that the best item had been rejected earlier.

It is easy to see that the optimal probability of selecting the best item does *not* tend to zero as n tends to infinity; consider the following stopping rule. Reject the first half of the objects and then select the first relatively best one (if any). This rule chooses the best object whenever the latter is among the second half of the objects while the second best object is among the first half. Hence, for every n, this rule succeeds with probability greater than $1/4$. Indeed, it has been established [9, 7, 4] (see below) that there exists an optimal rule that has the following form. Reject the first $r - 1$ objects and then select the first relatively best one or, if none has been chosen through the end, accept the last object. When n tends to infinity, the optimal value of r tends to n/e, and the probability of selecting the best is approximately $1/e$. (Lindley showed the above using backward induction [9]. Later, Gilbert and Mosteller provided a slightly more accurate bound for r [7]. Dynkin established the result as an application of the theory of Markov stopping times [4].)

It is not as easy to see that the optimal expected rank of the selected object tends to a finite limit as n tends to infinity. Observe that the above algorithm (for maximizing the probability of selecting the best object) yields an expected rank of $n/(2e)$ for the selected item; the argument is as follows. With probability $1/e$, the best item is among the first n/e items, and in this case the algorithm selects the

last item. The conditional expectation of the rank of the last object in this case is approximately $n/2$. Thus, the expected rank for the selected object in this algorithm tends to infinity with n. Indeed, in this paper we show that, surprisingly, the two goals are in fact in conflict (see section 1.2).

It can be proven by backward induction that there exists an optimal policy for minimizing the expected rank of the selected item that has the following form. Accept an object if and only if its rank relative to the previously seen objects exceeds a certain threshold (depending on the number of objects seen so far). Note that while the optimal algorithm for maximizing the probability of selecting the best has to remember only the best object seen so far, the threshold algorithm has to remember all the previous objects. (See [13] for solutions where the observer is allowed to remember only one of the previously presented items.) This fact suggests that minimizing the expected rank is harder. Thus, not surprisingly, finding an approximate solution for the dynamic programming recurrence for this problem seems significantly harder than in the case of the first variant of the problem, i.e., when the goal is to maximize the probability of selecting the best. Chow et al. [3] showed that the optimal expected rank of the selected object is approximately 3.8695. The question of whether higher powers of the rank of the selected object tend to finite limits as n tends to infinity was resolved in [13]. It has also been shown that if the order of arrivals is determined by an adversary, then no algorithm can yield an expected rank better than $n/2$ [14].

We thank an anonymous referee who pointed out that the case $k = 1$ and general z can be deduced from results of Robbins (see page 389 in [15], and [5]). He showed that if the "loss" of picking a candidate with absolute rank i is $(i+1)\cdots(i+z-1)$, then the value of the optimal rule tends to $z![\Pi_{j=1}^{\infty}((j+z+1)/j)^{1/(j+z)}]^z$ as n tends to infinity.

The case of a general k. There has been much interest in the case where more than one object has to be selected. It is not hard to see that for every fixed k, the maximum probability of selecting the best k objects does not tend to zero as n tends to infinity. The proof is as follows. Partition the sequence of n objects into k disjoint intervals, each containing n/k consecutive items. Apply the algorithm for maximizing the probability of selecting the best object to each set independently. The resulting algorithm selects the best item in each interval with probability e^{-k}. The probability that the best k objects belong to distinct intervals tends to $k!/k^k$ as n tends to infinity. For this first variant of the problem, the case of $k = 2$ was considered in [11]; Vanderbei [18] and, independently, Glasser, Holzsager, and Barron [8] considered the problem for general k. They showed that there is an optimal policy with the following threshold form. Accept an object with a given relative rank if and only if the number of observations exceeds a critical number that depends on the number of items selected so far; in addition, an object which is worse than any of the already rejected objects need not be considered. Notice that this means that not all previously seen items have to be remembered, but only those that were already selected and the best among all those that were already rejected. This property is analogous to what happened in the $k = 1$ case, where the goal was to maximize the probability of selecting the best item. Both papers derive recursive relations using backward induction. General solutions to their recurrences are not known, but the authors give explicit solutions (i.e., critical values and probability) for the case of $n = 2k$ [8, 18] and $n = 2k + 1$ [8]. Vanderbei [18] also presents certain asymptotic results as n tends to infinity and k is fixed and also as both k and n tend to infinity so that $(2k - n)/\sqrt{n}$ remains finite.

In analogy to the case of $k = 1$, bounding the optimal expected sum of ranks of k selected items appears to be considerably harder than minimizing the probability of selecting the best k items. Also, here it is not obvious whether or not this sum tends to a finite limit when n tends to infinity. Backward induction gives recurrences that seem even harder to solve than those derived for the case of maximizing the probability of selecting the best k. Such equations were presented by Henke [10], but he was unable to approximate their general solutions.

An anonymous referee has pointed out to us that the fact that the expected sum of powers of ranks accepted items is bounded as n tends to infinity can be derived from standard principles. However, there has not been an explicit solution for obtaining a bounded expected sum.

1.2. Our results. In this paper we present a family of explicit algorithms for the secretary problem such that for each positive integer z, the family includes an algorithm for accepting items, where for all values of n and k the resulting expected sum of the zth powers of the ranks of the accepted items is at most

$$\frac{k^{z+1}}{z+1} + C(z) \cdot k^{z+0.5} \log k,$$

where $C(z)$ is a constant.[1]

Clearly, the sum of ranks of the zth powers of the best k objects is $k^{z+1}/(z+1) + O(k^z)$. Thus, the sum achieved by our algorithms is not only bounded by a value independent of n, but also differs from the best possible sum only by a relatively small amount. For every fixed k, this expected sum is bounded by a constant. Thus we resolve the above open questions regarding the expected sum of ranks and, in general, zth powers of ranks, of the selected objects.

Our approach is very different from the dynamic programming approach taken in most of the papers mentioned above. It has been more successful in obtaining explicit solutions to this classical problem and can more easily be used to obtain explicit solutions for numerous generalizations.

We remark that our approach does not partition the items into k groups and select one item in each. Such a method is suboptimal since, with high probability, a constant fraction of the best k items appears in groups where they are not the only ones from the best k. Therefore, this method rejects a constant fraction of the best k with high probability, and so the expected value of the sum of the ranks obtained by such an algorithm is greater by at least a constant factor than the optimal.

Since the expected sums achieved by our algorithms depend only on k and z and, in addition, the probability of our algorithms to select an object does not increase with its rank, it will follow that the probabilities of our algorithms to actually select the best k objects depend only on k and z and hence, for fixed k and z, do not tend to zero when n tends to infinity. In particular, this means that for $k = z = 1$, our algorithms will select the best possible object with probability bounded away from zero.

In contrast, for any algorithm for the problem, if the order of arrival of items is the worst possible (i.e., generated by an oblivious adversary), then the algorithm yields an expected sum of at least $kn^z 2^{-(z+1)}$ for the zth powers of the ranks of selected items. Our lower bound holds also for randomized algorithms.

Finally, in section 1.1 we observed that an optimal algorithm for maximizing the probability of selecting the best object results in an unbounded expected rank of

[1] $\log k \equiv \max\{1, \log_2 k\}$.

the selected object. As a second part of this work we show that this fact is not a coincidence; the two goals are in fact in conflict. No algorithm can simultaneously optimize the expected rank and the probability of selecting the best. We derive a lower bound on the trade-off between the probability of accepting the best object and the expected rank of the accepted item.

2. The algorithms. In this section we describe a family of algorithms for the secretary problem, such that for each positive integer z, the family includes an algorithm for accepting objects, where the resulting expected sum of the zth powers of the ranks of accepted objects is

$$\frac{k^{z+1}}{z+1} + O(k^{z+0.5} \log k).$$

In addition, it will follow that the algorithm accepts the best k objects with positive probability that depends only on k and z. Let z be the positive integer that we are given. Denote $p = 64 + \log^2 k$.

For the convenience of exposition, we assume without loss of generality that n is a power of 2. We partition the sequence $[1, \ldots, n]$ (corresponding to the objects in the order of arrival) into $m = \log n + 1$ consecutive intervals $I_i (i = 1, \ldots, m)$, so that

$$I_i = \begin{cases} [1 + n\sum_{j=1}^{i-1} 2^{-j}, n\sum_{j=1}^{i} 2^{-j}] & \text{if } 1 \le i \le m-1, \\ \{n\} & \text{if } i = m. \end{cases}$$

In other words, the first $m-1$ intervals are $[1, \frac{n}{2}]$, $[\frac{n}{2}+1, \frac{3n}{4}], \ldots$, each containing a half of the remaining elements. The mth interval contains the last element. Note that $|I_i| = \lceil n/2^i \rceil$ $(i = 1, \ldots, m-1)$.

Let us refer to the first

$$m' = \max\{0, \lfloor \log(k/p) \rfloor\}$$

intervals as the *opening* ones, and let the rest be the *closing* ones. Note that since $p \ge 64$, the last five intervals are closing. For an opening I_i, the expected number of those of the top k objects in I_i is

$$|I_i| \cdot \frac{k}{n} = k/2^i \quad (i = 1, \ldots, m').$$

(The latter is not necessarily an integer.) Furthermore, for any $d \le \sum_{j=1}^{m'} |I_i|$ (i.e., d is in one of the opening intervals), the expected number of those of the top k objects among the first d to arrive is $d \cdot \frac{k}{n}$.

Let

$$p_i = \begin{cases} k2^{-i} & \text{if } i \le m', \\ k2^{-m'} & \text{if } i = m'+1, \\ 0 & \text{if } m'+1 < i \le m. \end{cases}$$

Observe that $p_{m'+1} = k - \sum_{j=1}^{m'} p_j$.

We will refer to p_i as the *minimum number of acceptances required for* I_i $(i = 1, \ldots, m)$. Observe that for $i \le m'$, $p_i \ge k \cdot 2^{-\log k/p} = p$. On the other hand, $p_{m'+1} = k2^{-m'} \le k2^{-\log k/p+1} = 2p$.

Intuitively, during each interval the algorithm attempts to accept the expected number of top k objects that arrive during this interval and, in addition, to make up for the number of objects that should have been accepted prior to the beginning of this interval but have not. Note that since $p_i = 0$ for $i > m' + 1$, then during such intervals the algorithm only attempts to make up for the number of objects that should have been accepted beforehand and have not.

Let us explain this slightly more formally. During each execution of the algorithm, at the beginning of each interval, the algorithm computes a *threshold* for acceptance, with the goal that by the time the processing of the last object of this interval is completed, the number of accepted objects will be at least the minimum number of acceptances required prior to this time. In particular, recall that for $i = 1, \ldots, m, p_i$ denotes the minimum number of acceptances required for I_i. Given a "prefix" of an execution prior to the beginning of $I_i (i = 1, \ldots, m+1)$, let $Q_j (j = 0, \ldots, i-1)$ be the number of items accepted in I_j. Let $D_{i-1} = \max\{0, \sum_{j=1}^{i-1} p_j - \sum_{j=1}^{i-1} Q_j\}$. Roughly speaking, D_{i-1} is the difference between the minimum number of acceptances required prior to the beginning of I_i and the number of items that were actually accepted during the given prefix. Note that $D_0 = 0$.

Given a prefix of an execution prior to the beginning of I_i, let

$$A_i = \begin{cases} D_{i-1} + p_i & \text{if } \sum_{j=1}^{i-1} Q_j < k, \\ 0 & \text{otherwise.} \end{cases}$$

We refer to A_i computed at the beginning of I_i as the *acceptance threshold for* I_i in this execution. Loosely stated, given a prefix of execution of the algorithm prior to the beginning of I_i, A_i is the number of objects the algorithm has to accept during I_i in order to meet the minimum number required by the end of I_i. The algorithm will aim at accepting at least A_i objects during I_i. To ensure that it accepts that many, it attempts to accept a little more. In particular, during each opening interval I_i, the algorithm attempts to accept an expected number of $A_i + 6(z+1)\sqrt{A_i}\log k$. As we will see, this ensures that the algorithm accepts at least A_i objects during this interval with probability of at least $k^{-5(z+1)}$. During each closing interval I_i, the algorithm attempts to accept an expected number of $32(z+1)A_i$. This ensures that the algorithm accepts at least A_i objects during this interval with probability of at least $2^{-5(z+1)(a_i+1)}$.

We make the distinction between opening and closing intervals in order to restrict the expected rank of the accepted objects. If I_i is closing, then A_i may be much smaller than $\sqrt{A_i}\log k$. Let

$$B_i = \begin{cases} A_i + 6(z+1)\sqrt{A_i}\log k & \text{if } I_i \text{ is opening,} \\ 32(z+1)(A_i) & \text{if } I_i \text{ is closing.} \end{cases}$$

In order to accept an expected number of B_i objects during interval I_i, the algorithm will accept the dth item if it is one of the approximately $B_i \cdot 2^i d/n$ top ones among the first d. Since the order of arrival of the items is random, the rank of the dth object relative to the first d ones is distributed uniformly in the set $\{1, \ldots, d\}$. Therefore, the dth object will be accepted with probability of $B_i 2^i/n$, and hence, since $|I_i| = \lceil n/2^i \rceil$, the expected number of objects accepted during I_i is indeed B_i.

If at some point during the execution of the algorithm, the number of slots that still have to be filled equals the number of items that have not been processed yet, all the remaining items will be accepted regardless of rank. Analogously, if by the time the dth item arrives all slots have already been filled, this item will not be accepted.

Finally, the algorithm does not accept any of the first $\lceil n/(8\sqrt{k})\rceil$ items except in executions during which the number of slots becomes equal to the number of items before $\lceil n/(8\sqrt{k})\rceil$ items have been processed. Roughly speaking, this modification will allow us to bound the expected rank of the dth item in terms of its rank relative to the first d items.

The above leads to our algorithm, which we call *Select*.

ALGORITHM SELECT. *The algorithm processes the items, one at a time, in their order of arrival. At the beginning of each interval I_i, the algorithm computes A_i as described above. When the dth item ($d \in I_i$) arrives, the algorithm proceeds as follows.*

(i) *If all slots have already been filled, then the object is rejected.*

(ii) *Otherwise, if $d > \lceil n/(8\sqrt{k}\rceil$, then the following hold.*
   (a) *If $i \le m'$, the dth item is accepted if it is one of the top $\lfloor (A_i + 6(z+1)\sqrt{A_i}\log k)2^i d/n\rfloor$ items among the first d.*
   (b) *If $i > m'$, the algorithm accepts the dth item if it is one of the top $\lfloor 32(z+1)(A_i)2^i d/n\rfloor$ items among the first d.*

(iii) *Otherwise, if the number of slots that still have to be filled equals the number of items left (i.e., $n-d-1$), the dth item is accepted.*

We refer to acceptances under (iii), i.e., when the number of slots that still have to be filled equals the number of items that remained to be seen, as *mandatory*, and to all other acceptances as *elective*. For example, if the dth item arrives during I_1, and the latter is opening, then the item is accepted electively if and only if it is one of the approximately

$$\lfloor (A_1 + 6(z+1)\sqrt{A_1}\log k)\cdot(2d/n)\rfloor = \lfloor (k + 12(z+1)\sqrt{k/2}\log k)\cdot(d/n)\rfloor$$

top objects among the first d. In general, if the dth object arrives during an opening I_i, then the object is accepted electively if and only if it is one of the approximately

$$\lfloor (2^i A_i + 6(z+1)\cdot 2^i\sqrt{A_i}\log k)\cdot(d/n)\rfloor$$

top objects among the first d.

3. Analysis of Algorithm Select. Very loosely stated, the proof proceeds as follows. In section 3.1 we show that for $i = 1,\dots,m+1$ ($m = \log n + 1$), with high probability, $D_{i-1} = 0$. Observe that this implies that for $i = 1,\dots,m$, with high probability, A_i is approximately p_i, i.e.,

$$A_i \approx \begin{cases} 2^{-i}k & \text{if } i \le m', \\ 2^{-m'}k \le 2p & \text{if } i = m'+1, \\ 0 & \text{if } i > m'+1. \end{cases}$$

In section 3.2 we show that if the dth object arrives during an opening I_i, then the conditional expectation of the zth power of its rank, given that it is accepted electively, is not greater than $2^{iz}\frac{1}{z+1}A_i^z + c_4(z)2^{iz}A_i^{z-0.5}\log k$ for some constant $c_4(z)$ (depending on z); if I_i is closing, this conditional expectation is not greater than $c_6(z)2^{iz}A_i^z$ for some constant $c_6(z)$. In section 3.3 these results of sections 3.1 and 3.2 are combined and it is established that if the dth object arrives during an opening I_i, then its conditional expected zth power of rank, given that it is accepted electively, is at most

$$\frac{k^z}{z+1} + c(z)2^{i/2}k^{z-0.5}\log k$$

for some constant $c(z)$. If I_i is closing, that conditional expected zth power of rank is at most $c'(z)k^z$ for some constant $c'(z)$ if $i = m' + 1$ and is approximately 0 otherwise. From this it will follow that the expected sum of the zth powers of ranks of the electively accepted objects is $\frac{1}{z+1}k^{z+1} + O(k^{z+0.5}\log k)$. In addition, we use the result of section 3.1 to show that the expected sum of the zth powers of ranks of mandatorily accepted objects is $O(k^{z+0.5}\log k)$. Thus the expected sum of the zth powers of ranks of the accepted objects is $\frac{1}{z+1}k^{z+1} + O(k^{z+0.5}\log k)$.

In addition, from the fact that the expected sum of the zth powers of ranks of the accepted objects is bounded by a value that depends only on k and z, it will also follow that the algorithm accepts the top k objects with probability that depends only on k and z.

3.1. Bounding the A_i's. In this section we show that for $i = 1, \ldots, m$, with high probability, A_i is very close to p_i. More precisely, we say that a prefix of execution prior to the end of the ith interval is *smooth* if, for each $j = 1 \ldots, i$, the value computed for A_i in this prefix is $\leq |I_j|$. We distinguish between smooth and nonsmooth executions.

In section 3.1.1 we show that for an opening interval I_i, in executions whose prefix prior to the end of the $(i-1)$th interval is smooth, the probability that $A_i > 2^j p_i$ decreases exponentially with j (part 1 of Lemma 3.3). For a closing I_i, in executions whose prefix prior to the end of the $(i-1)$th interval is smooth, the probability that $A_i > 2^j p_{m'+1}$ decreases exponentially both with j and with i (part 2 of Lemma 3.3). Parts 1 and 2 of Lemma 3.3 will follow, respectively, from Lemmas 3.1 and 3.2, which show that in executions whose prefix prior to the end of the ith interval is smooth, in I_i the algorithm accepts A_i objects with high probability (where A_i is computed for the prefix of the execution). Intuitively, the restriction to smooth executions is necessary since at most $|I_i|$ objects can be selected in I_i. Lemma 3.3 implies that for each $i = 1, \ldots, m$, in executions whose prefix prior to the end of the ith interval is smooth, with high probability, by the end of I_i the number of objects that were already accepted is not smaller than the minimum number of acceptance required prior to this point. The latter holds even if I_i started at a disadvantage in the sense that the minimum number of acceptances required prior to I_i was greater than the number of objects that were actually accepted by that point.

Clearly, Lemma 3.3 implies that in smooth executions, with high probability, A_i is very close to p_i. To complete the proof that A_i is close to p_i, section 3.1.2 shows that nonsmooth executions are rare. In particular, section 3.1.2 uses Lemma 3.3 to show that in executions whose prefix prior to the end of the $(i-1)$st interval is smooth, the probability that $A_i > |I_i|$ is less than $c(z)n^{-2.5(z+1)}$ for some constant $c(z)$ (Lemmas 3.4 and 3.5). The case of $k \geq n/2$ is excluded (Lemma 3.5) and thus handled separately later (section 3.3).

3.1.1. Smooth prefixes. Denote by E_i the prefix of an execution E prior to the end of I_i. Note that E_m is E. We say that E_i is smooth if for $j = 1, \ldots, i, A_j$ computed in E_i is $\leq |I_j|$. Denote by M_{E_i} the event in which E_i is smooth.

LEMMA 3.1. *For every $i \leq m'$ and for any value a_i of A_i,*

$$\mathrm{Prob}\{D_k > 0 \mid \{A_i = a_i\} \cap \mathrm{M}_{E_i}\} < k^{-5(z+1)}.$$

Proof. Note that $D_i > 0$ only if the number of objects accepted in I_i is less than a_i.

Overview. Loosely stated, the algorithm accepts the dth object electively if it is one of the top $\lfloor (A_i + 6(z+1)\sqrt{A_i}\log k)\frac{2^i d}{n} \rfloor$ objects among the first d. Since the

objects arrive in a random order, the rank of the dth object within the set of first d is distributed uniformly and hence it will be accepted electively with probability not less than $\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i d}{n}\rfloor/d$. Moreover, the rank of the dth object within the set of the first d is independent of the arrival order of the first $d-1$, and hence is independent of whether or not any previous object in this interval, say the d_1th one, is one of the top $\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)2^i d_1/n\rfloor$ objects among the first d_1. The rest of the proof follows from computing the expected number of accepted candidates and the Chernoff inequality (see [2]).

We proceed with the actual proof. Suppose $i \le m'$, and let a_i be the acceptance threshold computed for I_i in a given execution. Recall that if the dth object arrives during I_i while there are still empty slots $d > \lceil n/(8\sqrt{k})\rceil$, and $i \le m'$, then the algorithm accepts electively if it is one of the top $\lfloor (A_i + 6(z+1)\sqrt{A_i}\log k)\frac{2' d}{n}\rfloor$ objects among the first d. (If either $d \le \lceil n/(8\sqrt{k})\rceil$ or there are no empty slots when the dth object arrives, it may not be accepted electively.) Since the objects arrive in a random order, the rank of the dth object within the set of first d is distributed uniformly and hence it will be accepted electively with probability not less than $\min\{1, \lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i d}{n}\rfloor/d\}$. Moreover, the rank of the dth object within the set of the first d is independent of the arrival order of the first $d-1$. Hence this rank is independent of whether or not any previous object in this interval, say the d_1th one, is one of the top $\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)2^i d_1/n\rfloor$ objects among the first d_1.

Without loss of generality we may assume that

$$\min\left\{1, \left\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\ \frac{2^i d}{n}\right\rfloor \Big/ d\right\} = \left\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i d}{n}\right\rfloor \Big/ d < 1.$$

For, if for some $d \in I_i$, $\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i d}{n}\rfloor/d \ge 1$, then $a_i + 6(z+1)\sqrt{a_i}\log k \ge \frac{n}{2^i}$, and hence, for each $d \in I_i$, $\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i d}{n}\rfloor \ge 1$. In this case each object in I_i is accepted with probability 1 unless all slots have already been filled. If all slots are filled, then $D_i = 0$, and we are done. Otherwise, $Q_i = |I_i|$. It follows from the definition that $D_i \le a_i - Q_i$, and hence $D_i \le a_i - |I_i|$. Since by the lemma assumption, $a_i \le |I_i|$, it follows that D_i is nonpositive.

The rest of the proof follows directly from Chernoff's inequality. Formally, suppose the gth object is the first in I_i, i.e., $g = 1 + n\sum_{j=1}^{i-1} 2^{-j}$. Define $X_1, \ldots, X_{|I_i|}$ to be independent random $(0,1)$-variables such that

$$\text{Prob}\{X_t = 1\} = \frac{\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i(t+g-1)}{n}\rfloor}{t+g-1}$$

for $t + g - 1 > \lceil n/(8\sqrt{k})\rceil$. It follows from the reasoning above that if the dth object is in an opening I_i, then the probability that the dth object is accepted electively is not less than $\text{Prob}\{X_{d-g+1} = 1\}$. The independence of the order of arrival of the first $d-1$ objects also implies that

$$\text{Prob}\{D_i > 0\} \le\ \text{Prob}\{Q_i < a_i\} \le \text{Prob}\left\{\sum_{t=1}^{|I_i|} X_t < a_i\right\}.$$

Thus, to complete the proof, we will show that $\text{Prob}\{\sum_{t=1}^{|I_i|} X_t < a_i\} < k^{-5(z+1)}$. To this end, we first establish the following claim.

CLAIM 3.1. $\sum_{t=1}^{|I_i|} \text{Prob}\{X_t = 1\} \geq a_i + 5(z+1)\sqrt{a_i}\log k$.

Proof. We distinguish two cases.

Case 1. $i > 1$. In this case we have $g > n/2$ and the first inequality follows since $g > \lceil n/(8\sqrt{k})\rceil$. The same holds for $t + g - 1$ for each $t \in I_i$. It follows that

$$\text{Prob}\{X_t = 1\} = \frac{\lfloor (a_i + 6(z+1)\sqrt{a_i}\log k)\frac{2^i(t+g-1)}{n}\rfloor}{t+g-1}.$$

It follows that $i < m$ because, as noted in section 2, I_m is closing. Also, $|I_i| = n/2^i (i = 1, \ldots, m-1)$. Furthermore, note that since $i \leq m'$, $p_i \geq k2^{-\log k/p} = p$, and hence also k, $a_i \geq p$. Since $p \geq 64$, $\sqrt{a_i}\log k \geq \sqrt{p}\log p \geq 1 \geq 2^{-i+1}$. The above implies

$$\sum_{t=1}^{|I_i|} \text{Prob}\{X_t = 1\} = a_i + 5(z+1)\sqrt{a_i}\log k.$$

Case 2. $i = 1$. Here we have

$$\text{Prob}\{X_t = 1\} = \frac{\lfloor (a_1 + 6(z+1)\sqrt{a_1}\log k)\frac{2t}{n}}{t}$$

for every $t > \lceil n/(8\sqrt{k})\rceil$. Also, since I_m is closing, and since in our case I_1 is opening, we have $1 < m$, and also $|I_i| = n2^{-i}$ $(i = 1, \ldots, m-1)$. Furthermore, $\frac{1}{4\sqrt{k}} \geq \frac{2}{n}$, because (i) as noted above, if $i \leq m'$, then $p_i \geq p$, and since $p_1 = \frac{k}{2}$, we have $k \geq 2p$, and (ii) $p \geq 64 + \log^2 k$, so $8\sqrt{k} \leq \sqrt{p}\sqrt{k} \leq k \leq n$. Other relations to note are: $a_1 = p_1 = \frac{k}{2}$, $k \geq 2p \geq 128$, and $a_1 = k/2$. The above implies

$$\sum_{t=1}^{|I_1|} \text{Prob}\{X_t = 1\} \geq a_1 + 5(z+1)\sqrt{a_1}\log k. \qquad \square$$

An inequality related to Chernoff's states the following. *Let* $X_1, \ldots, X_n$ *be independent random* $(0,1)$*-variables with* $\text{Prob}\{X_i = 1\} = p_i$, $0 < p_i < 1$. *Let* $X = \sum_{i=1}^n X_i$ *and* $\mu = \sum_{i=1}^n p_i$. *Then, for* $\delta \in [0,1]$,

$$\text{Prob}\{X < (1-\delta)\mu\} < \exp\left(-\frac{1}{2}\mu\delta^2\right).$$

Using the claim, we apply Chernoff's inequality to our X_t's to get

$$\text{Prob}\left\{\sum_{t=1}^{|I_i|} X_t < a_i\right\} \exp(-5(z+1)\log k) < k^{-5(z+1)}. \qquad \square$$

LEMMA 3.2. *If* $n \geq 16$, *then for every* $i > m'$,

$$\text{Prob}\{D_i > 0 \mid \{A_i = a_i\} \cap \text{M}_{E_i}\} < 2^{-5(z+1)(a_i+1)}.$$

Proof. Suppose $i > m'$, and let a_i be the acceptance threshold computed for I_i. First, observe that $A_i > 0$ $(i = m'+1, \ldots, m)$ implies $A_i \geq 1$. For, by definition,

$$A_i = \begin{cases} D_{i-1} + p_i & \text{if } \sum_{j=1}^{i-1} Q_j < k, \\ 0 & \text{otherwise.} \end{cases}$$

Hence, for $i = m' + 1$, if $k - \sum_{j=1}^{m'} Q_j \leq 0$, then $A_{m'+1} = \cdots = A_m = 0$, and the observation follows. Assume $k - \sum_{j=1}^{m'} Q_j > 0$. Then

$$A_{m'+1} = D_{m'} + p_{m'+1} = \max\left\{0, \sum_{j=1}^{m'} p_j - \sum_{j=1}^{m'} Q_j\right\} + p_{m'+1}$$
$$= \max\left\{p_{m'+1}, k - \sum_{j=1}^{m'} Q_j\right\} \geq k - \sum_{j=1}^{m'} Q_j.$$

Since k and Q_j are integers, and by our assumption $k - \sum_{j=1}^{m'} Q_j > 0$, this means $A_{m'+1} \geq 1$. For $i > m'+1$, if $k - \sum_{j=1}^{i-1} Q_j \leq 0$, then $A_i = 0$ by definition. Otherwise, $A_{m'+1} > 0$ and hence as reasoned above is $\geq k - \sum_{j=1}^{m'} Q_j$. Thus

$$A_i = D_{i-1} = \max\left\{0, A_{m'+1} - \sum_{j=m'+1}^{i-1} Q_j\right\} \geq k - \sum_{j=1}^{m'} Q_j - \sum_{j=m'+1}^{i-1} Q_j = k - \sum_{j=1}^{i-1} Q_j.$$

Thus, since k and Q_j are integers and $k - \sum_{j=1}^{i-1} Q_j > 0$ by assumption, then $A_j \geq 1$.

If $a_i = 0$, the lemma follows since $D_i \leq a_i$. Thus, assume that $a_i \geq 1$. The proof is analogous to that of Lemma 3.1.

Recall that for $d > \lceil n/(8\sqrt{k})\rceil$, if the dth object arrives during a closing I_i while there are still empty slots, then the object is accepted electively if it is one of the top $\lfloor 32(z+1)A_i 2^i d/n\rfloor$ objects among the first d. (If either $d \leq \lceil n/(8\sqrt{k})\rceil$, or there are no empty slots, this object is not accepted electively.) Since the rank of the dth object in the set of the first d is uniformly distributed, it will be accepted electively with probability not less than $\min\{1, \lfloor 32(z+1)a_i 2^i d/n\rfloor/d\}$.

As in the proof of Lemma 3.1, we may assume that $\min\{1, \lfloor 32(z+1)a_i 2^i d/n\rfloor/d\} = \lfloor 32(z+1)a_i 2^i d/n\rfloor/d < 1$. We apply again Chernoff's inequality. Suppose the gth object is the first to arrive during I_i, i.e., $g = 1 + n\sum_{j=1}^{i-1} 2^{-j}$. Let $X_1, \ldots, X_{|I_i|}$ be independent random $(0,1)$-variables such that

$$\text{Prob}\{X_t = 1\} = \lfloor 32(z+1)a_i 2^i (t+g-1)/n\rfloor/(t+g-1)$$

for $t+g-1 > \lceil n/(8\sqrt{k})\rceil$. It follows that if the dth object arrives during I_i and $i > m'$, then the probability that it is accepted electively is not less than $\text{Prob}\{X_{t-g+1} = 1\}$. It also follows that $\text{Prob}\{D_i > 0\} \leq \text{Prob}\{Q_i < a_i \mid A_i = a_i\} \leq \text{Prob}\{\sum_{t=1}^{|I_i|} X_t < a_i\}$. Thus, to complete the proof, we will show that $\text{Prob}\{\sum_{t=1}^{|I_i|} X_t < a_i\} < 2^{-5(z+1)(a_i+1)}$.

To show this we first prove the following claim.

CLAIM 3.2. $\sum_{t=1}^{|I_i|} \text{Prob}\{X_t = 1\} \geq 16(z+1)a_i$.

Proof. Again, we distinguish two cases.

Case 1. $i > 1$. Here we rely on the following. For $i > 1$, we have $g > n/2$; thus $g > \lceil n/(8\sqrt{k})\rceil$, and hence so is $t + g - 1$ for each t in I_i. It follows that $\text{Prob}\{X_t = 1\} = \lfloor 32(z+1)a_i 2^i (t+g-1)/n\rfloor/(t+g-1)$. Also, we have $|I_i| = \lceil n/2^i\rceil$. Finally, we may assume $a_i \geq 1$, so $8(z+1)a_i \geq 2 \geq 2^{-i+2}$. The above implies

$$\sum_{t=1}^{|I_i|} \text{Prob}\{X_t = 1\} \geq 16(z+1)a_i.$$

Case 2. $i = 1$. Here we rely on $g = 1$ and

$$\text{Prob}\{X_t = 1\} = \lfloor 32(z+1)a_1 2^1 t/n \rfloor /t \quad \text{for} \quad t > \lceil n/(8\sqrt{k}) \rceil.$$

Also, note that $\lceil n/2 \rceil$ objects arrive during I_1. Note that since $1 = i > m'$ in this case, we get $m' = 0$; thus by definition, $a_1 = p_1 = k2^{-m'} = k$. Finally, note that $n \geq 16$ and $z \geq 1$. The above implies

$$\sum_{t=1}^{|I_1|} \text{Prob}\{X_t = 1\} \geq 16(z+1)a_1. \qquad \square$$

The claim and Chernoff's inequality imply

$$\text{Prob}\left\{ \sum_{t=1}^{|I_i|} X_t < a_i \right\} < \exp\left(-\frac{1}{2} \cdot 16(z+1)a_i \cdot 0.9 \right) \leq 2^{-5(z+1)(a_i+1)}.$$

LEMMA 3.3.

(i) *For $i \leq m'$ and for all j,*

$$\text{Prob}\{A_i > k2^{-i}(2^j - 1) \mid \text{M}_{E_{i-1}}\} \leq k^{-5(z+1)j}.$$

(ii) *If $n \geq 16$, then for $i > m'$, $j \geq 0$,*

$$\text{Prob}\{A_i > k2^{-m'}(2^j - 1) \mid \text{M}_{E_{i-1}}\} \leq k^{-5(z+1)j}(2^{-5(z+1)})^{i-m'-1}.$$

Proof. For the proof of (i), suppose $i \leq m'$. Without loss of generality we may assume that $j \geq 1$, because for $j \leq 0$, we have $k^{-5(z+1)j} \geq 1$, and (i) follows.

Recall that the minimum number of acceptances required for an opening interval I_i is $p_i = k2^{-i}$. Thus if $A_i > k2^{-i}$, then $D_{i-1} > 0$. Moreover,

$$\frac{k}{2^i}(2^j - 1) = \frac{k}{2^i}(1 + 2 + \cdots + 2^{j-1}) = p_i + p_{i-1} + \cdots + p_{i-j+1}.$$

By induction, if $A_i > k2^{-i}(2^j - 1)$, then D_{i-1}, $D_{i-2}, \ldots, D_{i-j}$ are positive. Thus, it suffices to bound the probability

$$\begin{aligned}
&\text{Prob}\{(D_{i-1} > 0) \cap (D_{i-2} > 0) \cap \cdots \cap (D_{i-j} > 0) \mid \text{M}_{E_{i-1}}\} \\
&\quad = \text{Prob}\{D_{i-1} > 0 \mid (D_{i-2} > 0) \cap (D_{i-3} > 0) \cap \cdots \cap (D_{i-j} > 0) \cap \text{M}_{E_{i-1}}\} \\
&\quad \cdot \text{Prob}\{(D_{i-2} > 0) \cap (D_{i-3} > 0) \cap \cdots \cap (D_{i-j} > 0) \mid \text{M}_{E_{i-1}}\} \\
&\quad \leq \text{Prob}\{D_{i-1} > 0 \mid \text{M}_{E_{i-1}}\} \\
&\quad \cdot \text{Prob}\{(D_{i-2} > 0) \cap (D_{i-3} > 0) \cap \cdots \cap (D_{i-j} > 0) \mid \text{M}_{E_{i-1}}\},
\end{aligned}$$

where the last inequality follows from the way the algorithm sets the thresholds. Here we use an inductive argument. Thus, Lemma 3.1 implies that each of the underlying events $\{D_q > 0\}$ $(q = 1, \ldots, i-1)$ occurs with probability less than $k^{-5(z+1)}$. Clearly, each of the events $\{D_q > 0\}$ $(q \leq 0)$ occurs with probability 0 and hence less than $k^{-5(z+1)}$. Thus,

$$\text{Prob}\{A_i > k2^{-i}(2^j - 1)\} \leq (k^{-5(z+1)})^j = k^{-5(z+1)j}.$$

For the proof of (ii), suppose $i > m'$. Recall that

$$p_{m'+2} = \cdots = p_m = 0$$

and

$$p_{m'+1} = k2^{-m'}.$$

Thus, if $A_i > k2^{-m'}(2^j - 1)$, then we must have

$$D_{m'} > k2^{-m'}(2^j - 2)$$

and

$$D_{m'+1}, \ldots, D_{i-1} > 0.$$

Lemma 3.2 implies that for each q ($q = m'+1 \ldots, m$), the underlying event $\{D_q > 0\}$ occurs with probability less than $2^{-5(z+1)}$. Again the dependency and the conditioning on $\mathrm{M}_{E_{i-1}}$ are working in our favor. Thus, if $j = 0$, then

$$\mathrm{Prob}\{A_i > k2^{-m'}(2^j - 1) \mid \mathrm{M}_{E_{i-1}}\} \leq (2^{-5(z+1)})^{i-m'-1} = k^{-5(z+1)j}(2^{-5(z+1)})^{i-m'-1}.$$

To complete the proof, assume $j \geq 1$. Then $D_{m'} > k2^{-m'}(2^j - 2) \geq 0$. Lemma 3.2 implies that the underlying event $\{D_{m'} > 0\}$ occurs with probability less than $k^{-5(z+1)}$. Moreover, since $D_{m'} \leq A_{m'}$, it follows that $D_{m'} > k2^{-m'}(2^j - 2)$ implies $A_{m'} > k2^{-m'}(2^j - 2) \geq k2^{-m'}(2^{j-1} - 1)$. The first part of the lemma thus implies that for $j \geq 1$, the underlying event $\{A_{m'} > k2^{-m'}(2^{j-1} - 1)\}$ occurs with probability at most $k^{-5(z+1)(j-1)}$. Hence

$$\mathrm{Prob}\{A_i > k2^{-m'}(2^j - 1) \mid \mathrm{M}_{E_{i-1}}\} \leq k^{-5(z+1)j}(2^{-5(z+1)})^{i-m'-1}. \qquad \square$$

3.1.2. Nonsmooth executions.

LEMMA 3.4. *If $i \leq m'$, then*

$$\mathrm{Prob}\{\neg \mathrm{M}_{E_i} \cap \mathrm{M}_{E_{i-1}}\} \leq 2^{5(z+1)} n^{-2.5(z+1)}.$$

Proof. First note that

$$\mathrm{Prob}\{\neg \mathrm{M}_{E_i} \cap \mathrm{M}_{E_{i-1}}\} \leq \ \mathrm{Prob}\{A_i > n2^{-i} \mid \mathrm{M}_{E_{i-1}}\}.$$

We distinguish two cases.

Case 1. $k \leq \sqrt{n}$. Here,

$$\mathrm{Prob}\{A_i > n2^{-i} \mid \mathrm{M}_{E_{i-1}}\} \ \leq \ \mathrm{Prob}\{A_i > (k2^{-i})(2^j - 1) \mid \mathrm{M}_{E_{i-1}}\},$$

where $j = \lceil \frac{1}{2} \log n \rceil - 1$.

From part (i) of Lemma 3.3 it follows that

$$\mathrm{Prob}\{A_i > (k2^{-i})(2^j - 1) \mid \mathrm{M}_{E_{i-1}}\} \leq 2^{5(z+1)} \cdot n^{-2.5(z+1)}.$$

Case 2. $k \geq \sqrt{n}$. Here it follows from part (i) of Lemma 3.3 that

$$\mathrm{Prob}\{A_i > n2^{-i} \mid \mathrm{M}_{E_{i-1}}\} \leq n^{-2.5(z+1)}. \qquad \square$$

LEMMA 3.5. *If $n \geq 16$, $k \leq \frac{1}{2}n$, and $i > m'$, then*

$$\text{Prob}\{\neg \text{M}_{E_i} \cap \text{M}_{E_{i-1}}\} \leq 2^{10(z+1)} n^{-2.5(z+1)}.$$

Proof.

$$\text{Prob}\{\neg \text{M}_{E_i} \cap \text{M}_{E_{i-1}}\} \ \leq \ \text{Prob}\{A_i > n2^{-i} \mid \text{M}_{E_{i-1}}\}.$$

We distinguish two cases.

Case 1. $k \leq \sqrt{n}$. Here,

$$\text{Prob}\{A_i > n2^{-i} \mid \text{M}_{E_{i-1}}\} \ \leq \ \text{Prob}\{A_i > k2^{-m'} \cdot (2^j - 1) \mid \text{M}_{E_{i-1}}\},$$

where $j = \max\{0, \lceil \frac{1}{2} \log n \rceil - i + m' - 1\}$.

We distinguish again two cases.

Case 1.a. $\lceil \frac{1}{2} \log n \rceil - i + m' - 1 \leq 0$. In this case, $i \geq \lceil \frac{1}{2} \log n \rceil + m' - 1$. From part (ii) of Lemma 3.3 it follows that

$$\text{Prob}\{A_i > k2^{-m'} \cdot (2^j - 1) \mid \text{M}_{E_{i-1}}\} \leq 2^{10(z+1)} n^{-2.5(z+1)}.$$

Case 1.b. $\lceil \frac{1}{2} \log n \rceil - i + m' - 1 \geq 0$. From part (ii) of Lemma 3.3 it follows that

$$\text{Prob}\{A_i > k2^{-m'} \cdot (2^j - 1) \mid \text{M}_{E_{i-1}}\} \leq k^{-5(z+1)j} 2^{-5(z+1)(i-m'-1)}.$$

We distinguish two cases.

Case 1.b.1. $k \leq 2$.

By definition

$$A_i \leq p_i + D_{i-1} = p_i + \max\left\{0, \sum_{j=1}^{i-1} p_j - \sum_{j=1}^{i-1} Q_j\right\} \leq \sum_{j=1}^{i} p_j \leq 2.$$

On the other hand, by our assumption $A_i \geq n2^{-i}$. Thus, $i \geq \log n - 1$. In addition, $m' = 0$ since by definition, $m' = \max\{0, \lfloor \log(k/p) \rfloor\}$ and $p \geq 64$. Thus,

$$k^{-5(z+1)j} 2^{-5(z+1)(i-m'-1)} \leq 2^{10(z+1)} n^{-5(z+1)}.$$

Case 1.b.2. $k \geq 2$.

$$k^{-5(z+1)j} 2^{-5(z+1)(i-m'-1)} \leq 2^{10(z+1)} \cdot n^{-2.5(z+1)}.$$

Case 2. $k \geq \sqrt{n}$. We distinguish again two cases.

Case 2.a. $i = m' + 1$. Since $k \leq \frac{1}{2}n$ and in view of part (ii) of Lemma 3.3,

$$\text{Prob}\{A_i > n2^{-i} \mid \text{M}_{E_{i-1}}\} \leq n^{-2.5(z+1)}.$$

Case 2.b. $i > m' + 1$. In this case we can prove that

$$\text{Prob}\{A_i > n2^{-i} \mid \text{M}_{E_{i-1}}\} \leq n^{-2.5(z+1)}$$

based on Lemma 3.2, the definition of $A_{m'+1}$, and the fact that $p \geq \log^2 k$. □

The case of $k \geq n/2$ is excluded (Lemma 3.5) and thus handled separately later (section 3.3).

3.2. Expected zth powers of ranks. Let us denote by R_d the random variable of the rank of the dth object. We define the *arrival rank* of the dth object as its rank within the set of the first d objects, i.e., one plus the number of better objects seen so far. Denote by S_d the random variable of the arrival rank. Denote by NA_d the event in which the dth object is accepted electively. In this section we show that there exist constants $c_4(z)$, $c_5(z)$, and $c_6(z)$ such that if the dth object arrives during an opening interval I_i, then

$$\mathcal{E}(R_d^z \mid \mathrm{NA}_d \cap \{A_i = a_i\}) \le 2^{iz} \cdot \frac{a_i^z}{z+1} + c_4(z) 2^{iz} a_i^{z-\frac{1}{2}} \log k + c_5(z) \sqrt{\frac{n}{D}} 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k$$

(Lemma 3.9); and if I_i is closing, then

$$\mathcal{E}(R_d^z \mid \mathrm{NA}_d \cap \{A_i = a_i\}) \le c_6(z) 2^{iz} a_i^z$$

(Lemma 3.10). To prove the above, we first prove a technical lemma (Lemma 3.6) showing that for fixed d and s, if $r \ge \frac{n}{d}s + \frac{n}{d}\sqrt{s}$, then $\mathrm{Prob}\{R_d = r \mid S_d = s\}$ decreases exponentially with r. This lemma will be used to prove Lemma 3.7, which states roughly that there exists a constant $c_2(z)$ such for every s

$$\mathcal{E}(R_d^z \mid S_d = s) \le \left(\frac{n}{d}\right)^z s^z + c_2(z) \left(\frac{n}{d}\right)^z s^{z-\frac{1}{2}} \log k.$$

Lemma 3.9 will follow by combining the result of Lemma 3.7 with the fact that given that the object is accepted electively during an opening interval I_i and $A_i = a_i$, then S_d is distributed uniformly in the set $\{1, 2, \ldots, \lfloor (a_i + 6(z+1)\sqrt{a_i} \log k) 2^i d/n \rfloor\}$. Lemma 3.10 will follow analogously by combining the result of Lemma 3.7 with the fact that given that the object is accepted electively during a closing interval I_i and $A_i = a_i$, then S_d is distributed uniformly in the set $\{1, 2, \ldots, \lfloor 32(z+1) a_i 2^i d/n \rfloor\}$.

LEMMA 3.6. *For all s and $j \ge \frac{n}{d}\sqrt{s}$,*

$$\mathrm{Prob}\left\{R_d = \frac{n}{d}s + j \mid S_d = s\right\} \le \exp\left(-\frac{jd}{8n\sqrt{s}}\right).$$

Proof. Clearly,

$$\mathrm{Prob}\left\{R_d = \frac{n}{d}s + j \mid S_d = s\right\} = \frac{\binom{\frac{n}{d}s+j-1}{s-1}\binom{n-\frac{n}{d}s-j}{d-s}}{\binom{n-1}{d-1}}.$$

Denote

$$\alpha = \frac{\binom{\frac{n}{d}s-1}{s-1}\binom{n-\frac{n}{d}s}{d-s}}{\binom{n-1}{d-1}}.$$

It follows that

$$\begin{aligned}
&\mathrm{Prob}\left\{R_d = \tfrac{n}{d}s + j \mid S_d = s\right\} \\
&\le \alpha \left(\frac{1}{1-\frac{d}{n}}\right)^{\frac{j}{2}} \cdot \left(\frac{1}{1-\frac{d}{n}\frac{s}{s+\frac{d}{n}\frac{j}{2}}}\right)^{\frac{1}{2}} \cdot \left(1 - \frac{d}{n} \cdot \frac{1-\frac{s}{d}}{1-\frac{s}{d}}\right)^{\frac{j}{2}} \\
&\cdot \left(1 - \frac{d}{n} \cdot \frac{1-\frac{s}{d}}{1-\frac{s}{d}-\frac{j}{2n}}\right)^{\frac{j}{2}}.
\end{aligned}$$

Let us denote

$$\rho(n,d,s,j) = -\ln\left(1-\frac{d}{n}\cdot\frac{s}{s+\frac{d}{n}\frac{j}{2}}\right)+\ln\left(1-\frac{d}{n}\cdot\frac{1-\frac{s}{d}}{1-\frac{s}{d}-\frac{j}{2n}}\right),$$

so we can write

$$\text{Prob}\left\{R_d = \frac{n}{d}s+j \mid S_d = s\right\} \le \alpha e^{\rho(n,d,s,j)\frac{j}{2}}.$$

Observe that $\alpha \le 1$. Thus, to complete the proof, we show that $\rho(n,d,s,j) \le -\frac{d}{4n\sqrt{s}}$.

However,

$$\rho(n,d,s,j) = \sum_{t=1}^{\infty}\left(\left(\frac{s}{s+\frac{d}{n}\frac{j}{2}}\right)^t - \left(\frac{1-\frac{s}{d}}{1-\frac{s}{d}-\frac{j}{2n}}\right)^t\right)\left(\frac{d}{n}\right)^t\cdot\frac{1}{t}.$$

Thus, to complete the proof, it suffices to show that

(i) $$\frac{s}{s+\frac{d}{n}\frac{j}{2}} - \frac{1-\frac{s}{d}}{1-\frac{s}{d}-\frac{j}{2n}} \le -\frac{1}{4\sqrt{s}}$$

and

(ii) $$\text{for each } t>1,\ \left(\frac{s}{s+\frac{d}{n}\frac{j}{2}}\right)^t \le \left(\frac{1-\frac{s}{d}}{1-\frac{s}{d}-\frac{j}{2n}}\right)^t.$$

For the proof of (i),

$$\frac{s}{s+\frac{d}{n}\frac{j}{2}} - \frac{1-\frac{s}{d}}{1-\frac{s}{d}-\frac{j}{2n}} \le -\frac{\frac{dj}{2n}}{s+\frac{dj}{2n}} \le -\frac{\frac{\sqrt{s}}{2}}{s+\frac{\sqrt{s}}{2}} < -\frac{1}{4\sqrt{s}}.$$

Clearly, (ii) follows from (i). □

LEMMA 3.7. *There exist constants $c_2(z)$, $c_3(z)$, and $c_{18}(z)$ such that for all $d \ge \frac{n}{k}$ and s,*

$$\mathcal{E}(R_d^2 \mid S_d = s) \le \left(\frac{n}{d}\right)^z\left(1+\frac{c_3(z)}{k}\frac{d}{n}\right)s^z + c_2(z)\left(\frac{n}{d}\right)^z s^{z-\frac{1}{2}}\log k + c_{18}(z)\left(\frac{n}{d}\right)^z s^{z/2}\log^z k.$$

The proof is omitted.

LEMMA 3.8. *For every $x \ge 1$, there exists a constant $c_{20}(x)$, such that for all intervals I_i and for all values a_i of A_i, if the dth object arrives during I_i, and $d \ge n/\sqrt{k}$, then*

$$\left(\frac{n}{d}\right)^{x/2} 2^{ix/2} a_i^{x/2}\log^x k \le c_{20}(x)\sqrt{\frac{n}{d}}2^{i(x-\frac{1}{2})}a_i^{x-\frac{1}{2}}\log k.$$

Proof. If $a_i = 0$, the lemma follows. Thus assume $a_i > 0$. It suffices to prove

$$\left(\frac{n}{d}\right)^{(x-1)/2}\log^{x-1} k \le c_{20}(x)2^{i(x-1)/2}a_i^{(x-1)/2},$$

where $c_{20}(x)$ is a constant. We distinguish two cases.

Case 1. $i = 1$. Based on the relations $d \geq n/\sqrt{k}$, $\log k \leq ck^{1/4}$ for some constant c, and $a_1 \geq k/2$ (in particular, if I_1 is opening, then $a_1 = k/2$, and otherwise we get that $m' = 0$; thus by definition, $a_1 = p_1 = k2^{-m'} = k$), we get

$$\left(\frac{n}{d}\right)^{(x-1)/2} \log^{x-1} k \leq c_{20}(x) 2^{i(x-1)/2} a_1^{(x-1)/2},$$

where c and $c_{20}(x)$ are constants.

Case 2. $i > 1$. We distinguish two cases.

Case 2.a. I_i is opening, i.e., $i \leq m'$. In this case $d > n/2$. Also, since I_i is an opening interval, we have (as mentioned in section 2) that $p_i \geq p$, and hence also $a_i \geq p$, and $p > \log^2 k$ by definition. It follows that

$$\left(\frac{n}{d}\right)^{(x-1)/2} \log^{x-1} k \leq c_{20}(x) 2^{i(x-1)/2} a_i^{(x-1)/2},$$

where $c_{20}(x)$ is a constant.

Case 2.b. I_i is closing i.e., $i > m'$. Here, $p = \log^2 k + 64$, and hence $\frac{k}{p} \geq c \log^2 k$ for some constant c. Also, (i) as observed in the beginning of the proof of Lemma 3.2, for closing I_i, $A_i > 0$ implies that $A_i \geq 1$, and (ii) by our assumption in the beginning of the proof, $a_i \neq 0$. It follows that

$$\left(\frac{n}{d}\right)^{(x-1)/2} \log^{x-1} k \leq c_{20}(x) 2^{i(x-1)/2} a_i^{(x-1)/2},$$

where c and $c_{20}(x)$ are constants. □

LEMMA 3.9. *There exist constants $c_4(z)$ and $c_5(z)$ such that for all opening intervals I_i* (i.e., $i \leq m'$) *and for every value a_i of A_i, if the dth object arrives during I_i and $d \geq \frac{n}{k}$, then*

$$\mathcal{E}(R_d^z \mid \mathrm{NA}_d \cap \{A_i = a_i\}) \leq 2^{iz} \frac{1}{z+1} a_i^z + c_4(z) 2^{iz} a_i^{z-\frac{1}{2}} \log k + c_5(z) \sqrt{\frac{n}{d}} 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k.$$

Proof. Recall that if the dth object arrives during an opening interval I_i, it is accepted electively only if it is one of the top $\lfloor (a_i + 6(z+1)\sqrt{a_i} \log k) 2^i d/n \rfloor$ among the first d. Obviously, S_d is distributed uniformly in $\{1, \ldots, d\}$, so given $\mathrm{NA}_d \cap \{A_i = a_i\}$, S_d takes on any of the values in the set $\{1, \ldots, \lfloor (a_i + 6(z+1)\sqrt{a_i} \log k) 2^i d/n \rfloor\}$ with equal probability of $(\lfloor (a_i + 6(z+1)\sqrt{a_i} \log k) 2^i d/n \rfloor)^{-1}$. It follows that there exist constants $c(z)$, $c'(z)$, $c''(z)$, and $c'''(z)$ such that

$$\begin{aligned}
&\mathcal{E}(R_d^z \mid \mathrm{NA}_d \cap \{A_i = a_i\}) \\
&\quad \leq \left(1 + \frac{c_3(z)}{k} \frac{d}{n}\right) \cdot 2^{iz} \frac{1}{z+1} a_i^z + c(z) 2^{iz} a_i^{z-\frac{1}{2}} \log k + c'(z) 2^{iz} a_i^{z/2} \log^z k \\
&\qquad + c''(z) \sqrt{\frac{n}{d}} 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k + c'''(z) \left(\frac{n}{d}\right)^{z/2} 2^{iz/2} a_i^{z/2} \log^z k.
\end{aligned}$$

Thus, to complete the proof, it suffices to show that there exists a constant c such that

(i) $a_i^z / k \leq a_i^{z-\frac{1}{2}} \log k$,

(ii) $a_i^{z/2} \log^z k \leq a_i^{z-\frac{1}{2}} \log k$, and

(iii) $\left(\frac{n}{d}\right)^{z/2} 2^{iz/2} a_i^{z/2} \log^z k \le c\sqrt{\frac{n}{d}} 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k$.

For the proof of (i), since $a_i \le k$ we have that $a_i^z/k \le a_i^{z-1}$. Since, as noted above, $a_i \ge p$ and $k \ge p$, we have that a_i and $k \ge \log^2 k + 64$, and thus $a_i^{z-1} \le a_i^{z-\frac{1}{2}} \log k$, and (i) follows. (ii) follows from $z \ge 1$ and $a_i \ge \log^2 k$. (iii) follows from Lemma 3.8. □

LEMMA 3.10. *There exists a constant $c_6(z)$, such that for all closing intervals I_i (*i.e., *$i > m'$) and for all values a_i of A_i, if the dth object arrives during I_i, and $d \ge \frac{n}{\sqrt{k}}$, then*

$$\mathcal{E}(R_k^z \mid \mathrm{NA}_d \cap \{A_i = a_i\}) \le c_6(z) 2^{iz} a_i^z.$$

Proof. The proof is analogous to that of Lemma 3.9. Recall that if the dth object arrives during a closing I_i, then it is accepted electively if it is one of the top $\lfloor 32(z+1)a_i 2^i d/n \rfloor$ among the first d. Given $\mathrm{NA}_d \cap \{A_i = a_i\}$, R_d is uniformly distributed in the set $\{1, \ldots, \lfloor 32(z+1)a_i 2^i d/n \rfloor\}$. Thus there exist constants $c(z)$, $c'(z)$, $c''(z)$, and $c_6(z)$ such that

$$\begin{aligned} &E\mathcal{E}(R_d^z \mid \mathrm{NA}_d \cap \{A_i = a_i\}) \\ &\le c(z) 2^{iz} a_i^z + c'(z)\sqrt{\frac{n}{d}} 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k + c''(z) \left(\frac{n}{d}\right)^{z/2} 2^{iz/2} a_i^{z/2} \log^z k \\ &\le c_6(z) 2^{iz} a_i^z. \qquad \square \end{aligned}$$

3.3. Expected sum of ranks. In this section we show that the expected sum of the zth powers of ranks of the k accepted objects is

$$\frac{1}{z+1} k^{z+1} + O(k^{z+0.5} \log k)$$

(Theorem 3.1). This will follow by adding up the expected sum of the zth powers of ranks of electively accepted objects (Lemma 3.15), and the expected sum of the zth powers of ranks of mandatorily accepted objects (Lemma 3.17).

In section 3.3.1 we bound the expected sum of the zth powers of ranks of electively accepted objects. In particular, denote by SUMZ_i the sum of the zth powers of ranks of objects that are accepted electively during I_i. We first use Lemmas 3.9 and 3.10 of section 3.2 to show that there exist constants $c_7(z)$ and $c_8(z)$ such that if I_i is opening, then

$$\mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a_i) \le 2^{iz} \frac{1}{z+1} a_i^{z+1} + c_7(z) 2^{iz} a_i^{z+\frac{1}{2}} \log k$$

(Lemma 3.11); and if I_i is closing, then

$$\mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a_i) \le c_8(z) 2^{iz} a_i^{z+1}$$

(Lemma 3.12). Lemma 3.11 is then combined with part (i) of Lemma 3.3 and with Lemma 3.4, to show that there exists a constant $c_9(z)$ such that if I_i is opening, then

$$\mathcal{E}(\mathrm{SUMZ}_i) \le 2^{-i} \frac{k^{z+1}}{z+1} + c_9(z) 2^{-i/2} k^{z+\frac{1}{2}} \log k$$

(Lemma 3.15). Lemma 3.12 is combined with part (ii) of Lemma 3.3 and with Lemma 3.5, to show that there exists a constant $c_{10}(z)$ such that if I_i is closing, then

$$\mathcal{E}(\mathrm{SUMZ}_i) \le c_{10}(z) 2^{-i} k^{z+1}$$

(Lemma 3.14). The expected sum of the zth powers of ranks of electively accepted objects is obtained by summing up these results over all intervals (Lemma 3.15).

Section 3.3.2 bounds the expected sum of the zth powers of ranks of mandatorily accepted objects. It first shows that if in execution E some object $d \in I_i$ is accepted mandatorily, then the prefix of E prior to the end of I_{i+1} is not smooth (Lemma 3.16). Lemmas 3.4 and 3.5 of section 3.1.2 imply that, for each I_i, the probability that a prefix of execution E prior to the end of I_i is not smooth and is at most $c(z)n^{-2.5(z+1)} \log n$, where $c(z)$ is a constant. This bound applies thus also for the probability that objects will be mandatorily accepted in I_i. Lemma 3.17 combines this bound with the facts that the rank of an object never exceeds n, and the number of accepted objects is at most $k \le n$, to show that the expected sum of the zth powers of ranks of mandatorily accepted objects is $O(k^{z+0.5} \log k)$. The case of $k \ge \frac{1}{2}n$ is handled without the use of Lemma 3.5, since this lemma excludes it.

In addition, the fact that the expected sum of the zth powers of ranks of accepted objects is bounded by a value that does not depend on n will imply that the algorithm accepts the top k objects with positive probability that does not depend on n (Corollary 3.1).

3.3.1. Elective acceptances.

LEMMA 3.11. *There exists a constant $c_7(z)$ such that for all opening intervals I_i and for all values a_i of A_i,*

$$\mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a_i) \le 2^{iz} \cdot \frac{a_i^{z+1}}{z+1} + c_7(z) 2^{iz} a_i^{z+\frac{1}{2}} \log k.$$

Proof. It can be shown that if I_i is opening, then

$$\begin{aligned}
&\mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a_i) \\
&\quad \le 2^{iz} \cdot \frac{a_i^{z+1}}{z+1} + (c_4(z) + 6 + c_4(z)) 2^{iz} a_i^{z+\frac{1}{2}} \log k \\
&\qquad + (a_i + 6(z+1)\sqrt{a_i} \log k) \cdot \frac{2^i}{n} \cdot c_5(z) 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k \cdot \sum_{\substack{d \in I_i \\ d > \lceil n/(8\sqrt{k}) \rceil}} \sqrt{\frac{n}{d}}.
\end{aligned}$$

To complete the proof, it suffices to show that there is a constant $c(z)$ such that

$$(a_i + 6(z+1)\sqrt{a_i} \log k) \cdot \frac{2^i}{n} \cdot 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k \cdot \sum_{\substack{d \in I_i \\ d > \lceil n/(8\sqrt{k}) \rceil}} \sqrt{\frac{n}{d}} \le c(z) 2^{iz} a_i^{z+\frac{1}{2}} \log k.$$

For $i > 1$, we have $\frac{n}{d} \le 2$, and hence

$$\begin{aligned}
&(a_i + 6(z+1)\sqrt{a_i} \log k) \cdot \frac{2^i}{n} \cdot 2^{i(z-\frac{1}{2})} a_i^{z-\frac{1}{2}} \log k \cdot \sum_{\substack{d \in I_i \\ d > \lceil n/(8\sqrt{k}) \rceil}} \sqrt{\frac{n}{d}} \\
&\quad \le c(z) 2^{i(z-\frac{1}{2})} a_i^{z+\frac{1}{2}} \log k.
\end{aligned}$$

For $i = 1$, it can be proved that

$$\sum_{\substack{d \in I_1 \\ d > \lceil n/(8\sqrt{k}) \rceil}} \sqrt{\frac{n}{d}} \le \frac{n}{\sqrt{k}} \cdot \sqrt{\sqrt{k}} \cdot \sum_{i=1}^{\sqrt{k}} \sqrt{\frac{1}{i}} \le cn. \qquad \square$$

LEMMA 3.12. *There exists a constant $c_8(z)$ such that for all closing intervals I_i and for all acceptance thresholds a_i computed for I_i,*

$$\mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a_i) \leq c_8(z) 2^{iz} a_i^{z+1}.$$

Proof. The proof is analogous to that of Lemma 3.11. It can be shown that if I_i is closing, then there is a constant $c_8(z)$ such that

$$\mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a_i) \leq 32(z+1) a_i \frac{2^i}{n} \cdot \left\lceil \frac{n}{2^i} \right\rceil \cdot c_6(z) 2^{iz} a_i^z \leq c_8(z) 2^{iz} a_i^{z+1},$$

where $c_8(z)$ is a constant. □

Lemma 3.11 is combined with part (i) of Lemma 3.3 and with Lemma 3.4 to show the following lemma.

LEMMA 3.13. *There exists a constant $c_9(z)$ such that for all opening intervals I_i,*

$$\mathcal{E}(\mathrm{SUMZ}_i) \leq 2^{-i} \frac{k^{z+1}}{z+1} + c_9(z) 2^{-i/2} k^{z+\frac{1}{2}} \log k.$$

Proof. It can be shown that if I_i is opening, then

$$\mathcal{E}(\mathrm{SUMZ}_i \leq \mathrm{Prob}\{\neg \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = k) + \sum_{a=1}^{\infty} \mathrm{Prob}\{A_i = a \cap \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a).$$

To complete the proof, we can show that there exist constants $c(z)$ and $c'(z)$ such that

(1) $\mathrm{Prob}\{\neg \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = k) \leq c(z) 2^{-i/2} k^{z+\frac{1}{2}} \log k$,

(2) $\sum_{a=1}^{\infty} \mathrm{Prob}\{A_i = a \mid \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a) \leq 2^{-i} \frac{k^{z+1}}{z+1} + c'(z) 2^{-i/2} k^{z+\frac{1}{2}} \log k$.

LEMMA 3.14. *If $n \geq 16$, then there exists a constant $c_{10}(z)$ such that for any closing interval I_i,*

$$\mathcal{E}(\mathrm{SUM}_i) \leq c_{10}(z) 2^{-i} k^{z+1}.$$

Proof. The proof is analogous to that of Lemma 3.13. Suppose I_i is closing. Then

$$\mathcal{E}(\mathrm{SUMZ}_i) \leq \mathrm{Prob}\{\neg \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = k) + \sum_{a=1}^{\infty} \mathrm{Prob}\{A_i = a \mid \mathrm{M}_{E_{i-1}}\}.$$

To complete the proof, we can prove that there exist constants $c(z)$ and $c'(z)$ such that

(1) $\mathrm{Prob}\{\neg \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = k) \leq c(z) 2^{-i} k^{z+1}$,

(2) $\sum_{a=1}^{\infty} \mathrm{Prob}\{A_i = a \mid \mathrm{M}_{E_{i-1}}\} \cdot \mathcal{E}(\mathrm{SUMZ}_i \mid A_i = a) \leq c'(z) 2^{-i} k^{z+1}$. □

The following lemma completes the proof of the upper bound on the sum of the ranks of the electively accepted objects. It sums up the expected sum of ranks of electively accepted objects over all intervals.

LEMMA 3.15.

$$\sum_{i=1}^{m} \mathcal{E}(\mathrm{SUMZ}_i) \leq \frac{k^{z+1}}{z+1} + O(k^{z+0.5} \log k).$$

Proof. If $n \le 16$, the assertion is immediate. For $n > 16$, we can prove that there exists a constant $c_{11}(z)$ such that

$$\begin{aligned}\sum_{i=1}^{m} \mathcal{E}(\mathrm{SUMZ}_i) &\le \frac{k^{z+1}}{z+1}(1 - 2^{-\log(k/p)}) + c_9(z)2k^{z+\frac{1}{2}} \log k + c_{10}(z)k^z(\log^2 k + 64) \\ &\le \frac{k^{z+1}}{z+1} + c_{11}(z)k^{z+0.5}(\log k + 1) \\ &= \frac{k^{z+1}}{z+1} + O(k^{z+0.5} \log k). \qquad \square\end{aligned}$$

3.3.2. Mandatory acceptances. This section bounds the expected sum of mandatorily accepted objects. We first observe the following lemma.

LEMMA 3.16. *If the dth object is mandatorily accepted in execution E during I_i, then $\neg \mathrm{M}_{E_{i+1}}$.* *Proof.* First we claim that $i < m - 1$. For, as noted in section 2, I_{m-1}, I_m are closing. Thus, if there is still an empty slot in E by the time the dth object arrives during I_j $(j = m-1, m)$, then $a_j > 0$. As observed at the beginning of the proof of Lemma 3.2, in this case $a_j \ge 1$, since for closing I_j, $A_j > 0$ implies $A_j > 1$. Hence, this object will be electively accepted if it is among the top $\lfloor 32(z+1)a_j 2^j d/n \rfloor$ objects seen so far, and hence, it will be electively accepted if it is among the top

$$\lfloor 32(z+1)a_j 2^j d/n \rfloor \ge \lfloor 32(z+1)2^{\log n}(n-1)/n \rfloor \ge 32(z+1)n(n-1)/n \rfloor \ge n$$

objects seen so far. Thus, it will be electively accepted and hence not mandatorily accepted.

Assume the dth object is mandatorily accepted in E during I_i. By definition of mandatorily accepted, this implies that the number of open slots just before the dth object's arrival equals to the total number of objects remaining to be seen, i.e., $n - d + 1$. Since, as shown above, $i < m - 1$, it follows that I_{i+1} exists. Thus, at the beginning of I_{i+1}, the number of open slots equals to the total number of objects remaining to be seen. Moreover, since $i < m - 1$, the number of objects that remain to be seen just before the beginning of I_{i+1} is exactly $2|I_{i+1}|$. Thus,

$$\sum_{j=1}^{i} Q_j = k - 2|I_{i+1}|.$$

Therefore,

$$D_i = \sum_{j=1}^{i} p_j - \sum_{j=1}^{i} Q_j = \sum_{j=1}^{i} p_j - (k - 2|I_{i+1}|).$$

But

$$a_{i+1} = D_i + p_{i+1} = \sum_{j=1}^{i} p_j - (k - 2|I_{i+1}|) + p_{i+1} = 2|I_{i+1}| - \sum_{j=i+2}^{m} p_j.$$

To complete the proof, it suffices to show that $2|I_{i+1}| - \sum_{j=i+2}^{m} p_j > |I_{i+1}|$. We distinguish between two cases.

Case 1. $i \ge m'$. In this case, $\sum_{j=i+2}^{m} p_j = 0$, and hence

$$2|I_{i+1}| - \sum_{j=i+2}^{m} p_j = 2|I_{i+1}| > |I_{i+1}|.$$

Case 2. $i < m'$. In this case, it follows directly from the definition of p_j that $\sum_{j=i+2}^{m} p_j = p_{i+1} = k2^{-i-1}$. If $k = n$, then clearly all objects are electively accepted and none is mandatorily accepted. Thus, assume $k < n$. Then $k2^{-i-1} < n2^{-i-1} = |I_{i+1}|$. Thus

$$2|I_{i+1}| - \sum_{j=i+2}^{m} p_j > 2|I_{i+1}| - |I_{i+1}| = |I_{i+1}|. \qquad \square$$

Denote by SUMDZ_i the sum of the zth powers of ranks of objects that are accepted mandatorily during I_i.

LEMMA 3.17. *There exist constants $c_{21}(z)$ and $c_{22}(z)$ such that*

$$\sum_{i=1}^{m} \mathcal{E}(\mathrm{SUMDZ}_i) \le c_{21}(z)k^{z+\frac{1}{2}} \log k + c_{22}(z).$$

Proof. Again, if $n \le 16$, the assertion is immediate. For $n > 16$, we argue as follows. The number of accepted objects in I_i at most $|I_i|$, and the rank of any object is of course not greater than n. Thus,

$$\sum_{i=1}^{m} \mathcal{E}(\mathrm{SUMDZ}_i) \le \sum_{i=1}^{m'-1} \mathrm{Prob}\{\neg \mathrm{M}_{E_{i+1}}\} \cdot n2^{-i} \cdot n^z + \sum_{i=m'-1}^{m-1} \mathrm{Prob}\{\neg \mathrm{M}_{E_{i+1}}\} \cdot n2^{-i} \cdot n^z.$$

To complete the proof, we can show that there exist constants $c(z)$, $c'(z)$, and $c''(z)$ such that

(1) $\sum_{i=1}^{m'=-1} \mathrm{Prob}\{\neg \mathrm{M}_{E_{i+1}}\} \cdot n2^{-i} \cdot n^z \le c(z)$,

(2) $\sum_{i=m'-1}^{m-1} \mathrm{Prob}\{\neg \mathrm{M}_{E_{i+1}}\} \cdot n2^{-i} \cdot n^z \le c'(z)k^{z+\frac{1}{2}} \log k + c''(z)$. $\square$

Lemmas 3.15 and 3.17 imply the following theorem.

THEOREM 3.1. *The expected sum of ranks of accepted objects is at most*

$$\frac{1}{z+1}k^{z+1} + O(k^{z+0.5} \log k).$$

COROLLARY 3.1. *Algorithm Select accepts the best k objects with positive probability that depends only on k and z.* *Proof.* Theorem 3.1 implies that the expected sum of the zth powers of ranks of accepted objects is bounded by a value that is independent of n. Thus, there is some value r that does not depend on n such that with probability $\ge 1/2$, all accepted objects are of rank $\le r$. Clearly, the probability of acceptance decreases monotonically with the object's rank. Therefore, the probability that the k accepted objects are the best k objects is at least $\frac{1}{2}/\binom{r}{k}$. $\square$

4. Trade-off between small expected rank and large probability of accepting the best.

THEOREM 4.1. *Let p_0 be the maximum possible probability of selecting the best object. There is a $c > 0$ so that for all $\epsilon > 0$ and all sufficiently large n, if A is an algorithm that selects one of n objects, and the probability p_A that A selects the best one is greater than $p_0 - \epsilon$, then the expected rank of the selected object is at least c/ϵ.*

Proof. Suppose that contrary to our assertion there is an algorithm A that selects the best object with probability of at least $p_0 - \epsilon$ and yet the expected value of the rank of the selected object is less than c/ϵ. Starting from A, we construct another algorithm R so that R selects the best object with a probability $> p_0$.

Denote by OPT the following algorithm. Let n/e objects pass, and then accept the first object that is better than any one seen so far. If no object was accepted by the time the last object arrives, accept the last object. For n sufficiently large, this algorithm accepts the best object with the highest possible probability, and hence with probability p_0 [9].[2]

We define R by modifying A. The definition will depend on parameters $c_1 > d > 0$. We will assume that d is a sufficiently large absolute constant and c_1 is sufficiently large with respect to d. R will accept an object if at least one of the following conditions is satisfied.

(i) A accepts the object after time n/d and by time $n - c_1\epsilon n$ and the object is better than any one seen earlier.
(ii) OPT accepts the object whereas A accepted earlier some object which, at the time of acceptance, was known not to be the best one (that is, there was a better one before).
(iii) OPT accepts the object and A has already accepted some object by time n/d.
(iv) The object comes after time $n - c_1\epsilon n$, it is better than any object else seen before, and R has not yet accepted any object based on the rules (i), (ii), and (iii).
(v) The object is the nth object and R has not accepted yet any object.

Notation. Denote by BA, BR, and BOPT the events in which A, R, and OPT, respectively, accept the best object. Denote by B1, B2, and B3 the events in which the best object appears in the intervals $[1, n/d]$, $(n/d, t_0 = n - c_1\epsilon]$, and $(t_0, n]$, respectively. Denote by IA1, IA2, and IA3 the events in which A makes a selection in the intervals $[1, n/d]$, $(n/d, t_0 - n - c_1\epsilon]$, and $(t_0, n]$, respectively.

We distinguish between two cases.

Case 1. $\text{Prob}\{\text{IA1}\} \geq 3\epsilon/p_0$.

Claim 4.1.

$$\text{Prob}\{\text{BR} \mid IA1\} \geq P\{\text{BA} \mid IA1\} + p_0/2.$$

Proof. Suppose that A made a selection by time n/d. According to rule (iii), in this case R will accept an object that arrives after time n/d if and only if OPT accepts this object. By choosing d sufficiently large, we have that objects are accepted by OPT only after time n/d. Thus, if A made a selection by time n/d, R will accept the object if and only if OPT accepts it. Thus,

$$\text{Prob}\{\text{BR} \mid \text{IA1}\} = \text{Prob}\{\text{BOPT} \mid \text{IA1}\} = \text{Prob}\{\text{BOPT}\} \geq p_0.$$

The second inequality follows since the probability that OPT accepts the best object is independent of the order of arrival of the first n/d objects, and hence independent of whether or not A makes a selection by time n/d. On the other hand,

$$\text{Prob}\{\text{BA} \mid \text{IA1}\} \leq \text{Prob}\{\text{B1}\} \leq 1/d.$$

Thus, by choosing d to be sufficiently large, the claim follows. □

Claim 4.2.

$$\text{Prob}\{\text{BR} \mid \text{IA2}\} \geq \text{Prob}\{\text{BA} \mid \text{IA2}\}.$$

[2] In fact, $r = [(n - \frac{1}{2})e^{-1} + \frac{1}{2}]$ is a better approximation to r than ne^{-1}, although the difference is never more than 1 [7]. We ignore this difference for the sake of simplicity.

Proof. The claim follows immediately from the fact that if A picks the best object between n/d and t_0, then this object must be the best seen so far, and hence by rule (i), R picks the same object. □

CLAIM 4.3

$$\text{Prob}\{\text{BR} \mid \text{IA3}\} \geq \text{Prob}\{\text{BA} \mid \text{IA3}\}.$$

Proof. If IA3 holds, then neither A nor R have accepted any object until time t_0. Let X be the event when A chooses no later than R. By the definition of R we have that if $X \cap \text{IA3}$ holds, then either A accepts an object that already at the moment of acceptance is known not to be the best, or A and R accept the same object. Thus

$$\text{Prob}\{\text{BR} \mid \text{IA3} \cap X\} \geq \text{Prob}\{\text{BA} \mid \text{IA3} \cap X\}.$$

To complete the proof, it suffices to show that

$$\text{Prob}\{\text{BR} \mid \text{IA3} \cap \neg X\} \geq \text{Prob}\{\text{BA} \mid \text{IA3} \cap \neg X\}.$$

Suppose that $\text{IA3} \cap \neg X$ holds and R accepts an object at some time $t > t_0$. By definition, A has not accepted any object yet, and the object accepted by R at t is better than any object else seen earlier. Thus, if a better object than the one accepted by R arrives after time t, this means that the best object arrives after time t. Since the objects arrive in a random order, the rank of each dth arriving object within the set of first d is distributed uniformly. Hence, the probability that the best object will arrive after time t is at most $(n-t)/n \leq c_1\epsilon n$. Notice that this probability is independent of the ordering of the first t objects, and hence is independent of the fact that R has accepted the tth object. Therefore the probability that the object accepted by R is indeed the best object is at least $1 - c_1\epsilon n$, while the probability that A accepts the best one later is smaller than $c_1\epsilon n$. Thus, for any fixed choice of t and fixed order of the first t objects (with the property $\text{IA3} \cap \neg X$), the probability of BR is larger than BA, and hence $\text{Prob}\{\text{BR} \mid \text{IA3} \cap \neg X\} \geq \text{Prob}\{\text{BR} \mid \text{IA3} \cap \neg X\}$. □

Now we can complete the proof of Case 1.

$$\begin{aligned}
&\text{Prob}\{\text{BR}\} \\
&= \text{Prob}\{\text{BR} \mid \text{IA1}\} \cdot \text{Prob}\{\text{IA1}\} \\
&\quad + \text{Prob}\{\text{BR} \mid \text{IA2}\} \cdot \text{Prob}\{\text{IA2}\} \\
&\quad + \text{Prob}\{\text{BR} \mid \text{IA3}\} \cdot \text{Prob}\{\text{IA3}\} \\
&\geq \text{Prob}\{\text{BA} \mid \text{IA1}\} + p_0/2) \cdot \text{Prob}\{\text{IA1}\} \\
&\quad + \text{Prob}\{\text{BA} \mid \text{IA2}\} \cdot \text{Prob}\{\text{IA2}\} \\
&\quad + \text{Prob}\{\text{BA} \mid \text{IA3}\} \cdot \text{Prob}\{\text{IA3}\} \\
&= \text{Prob}\{\text{BA}\} + (p_0/2) \cdot \text{Prob}\{\text{IA1}\} \\
&\geq p_0 - \epsilon + (p_0/2) \cdot 3\epsilon/p_0 = p_0 + \epsilon/2.
\end{aligned}$$

The second inequality follows from Claims 4.1, 4.2, and 4.3. The fourth inequality follows from (i) $\text{Prob}\{\text{BA}\} \geq p_0 - \epsilon$ by the theorem assumption and (ii) $\text{Prob}\{\text{IA1}\} \geq 3\epsilon/p_0$ by Case I assumption.

Case II: $\text{Prob}\{\text{IA1}\} < 3\epsilon/p_0$.

Denote by BR1, BR2, and BR3 the events when R picks the best object and its selections are in the interval $[1, n/d]$, $(n/d, t_0]$ and $(t_0, n]$, respectively. Denote by BA1, BA2, and BA3 the corresponding events for A.

Since by the assumption of this case $\text{Prob}\{\text{IA1}\} < 3\epsilon/p_0$, we have

$$\text{Prob}\{\text{BA1}\} < 3\epsilon/p_0. \tag{1}$$

If A picks the best object between n/d and t_0, then this object must be the best seen so far, and hence by rule (1), R picks the same object. Thus

$$\text{Prob}\{\text{BR2}\} \geq \text{Prob}\{\text{BA2}\}. \tag{2}$$

By choosing d sufficiently large, we have that objects are accepted by OPT only after time n/d. Observe that in that case, if the second best comes by time n/d and the best comes after time t_0, then R accepts the best object. The probability that the second best object arrives by time n/d is $1/d$, and the conditional probability that the best object comes after time t_0, given that the second best comes by time n/d, is at least $c_1\epsilon$. It thus follows that

$$\text{Prob}\{\text{BR3}\} \geq c_1\epsilon/d. \tag{3}$$

For bounding $\text{Prob}\{\text{BA3}\}$, we first use the assumption that the expected rank of the object selected by A is less than c/ϵ to show the following.

Claim 4.4.

$$\text{Prob}\{\text{IA3}\} \leq 1/(2d).$$

Proof. Each of the $1/(10dc_1\epsilon)$ objects with a rank smaller than $1/(10dc_1\epsilon)$ arrives after time $t_0 = n - c_1\epsilon n$ with probability of at most $c_1\epsilon$. Therefore, with probability of at least $1 - 1/(10d)$, all objects that arrive after time t_0 are of rank larger than $1/(10dc_1\epsilon)$. Hence, if the probability of IA3 had been greater than $1/(2d)$, then the expected value of the rank would have been larger than c'/ϵ for some absolute constant $c' > 0$. Take the c of the theorem to be equal to c', and we get a contradiction to the assumption that the expected rank of the selected object is at most c/ϵ. □

Let B3 denote the event in which the best object arrives in interval $(t_0, n]$. Then $\text{Prob}\{\text{BA3}\} \leq \text{Prob}\{\text{IA3}\} \cdot \text{Prob}\{\text{B3} \mid \text{IA3}\}$. But B3 is independent of the order of arrival of the first t_0 objects and hence independent on whether or not A has accepted an object by time t_0. Thus, Claim 4.4 implies that $\text{Prob}\{\text{IA3}\} \cdot \text{Prob}\{\text{B3} \mid \text{IA3}\} = \text{Prob}\{\text{IA3}\} \cdot \text{Prob}\{\text{B3} \leq \frac{1}{2d} \cdot c_2\epsilon$. Thus,

$$\text{Prob}\{\text{BA3}\} \leq c_1\epsilon/(2d). \tag{4}$$

Equations (1) to (4) imply

$$\begin{aligned}
&\text{Prob}\{\text{BR}\} - \text{Prob}\{\text{BA}\} \\
&= \text{Prob}\{\text{BR1}\} - \text{Prob}\{\text{BA1}\} + \text{Prob}\{\text{BR2}\} - \text{Prob}\{\text{BA2}\} + \text{Prob}\{\text{BR3}\} - \text{Prob}\{\text{BA3}\} \\
&\geq -3\epsilon/p_0 + c_1\epsilon/d - c_1\epsilon(2d) \\
&= c_1\epsilon/(2d) - 3\epsilon/p_0 \geq 2\epsilon.
\end{aligned}$$

(The last inequality follows from our assumption that c_1 is sufficiently large with respect to d.) Therefore,

$$\text{Prob}\{\text{BR}\} \geq \text{prob}\{\text{BA}\} + 2\epsilon \geq p_0 - \epsilon + 2\epsilon > p_0. \qquad \square$$

5. Deterministic arrivals. In this section we consider the case where the order of arrivals is not random but is determined by an adversary that knows the algorithm, i.e., an oblivious adversary. We show that against such an adversary, no algorithm can obtain an expected sum of the zth powers of ranks of selected items that is less than $kn^z/2^{z+1}$. In particular, this expected sum tends to infinity with n. This lower bound holds also for randomized algorithms.

Given an algorithm A, we construct a sequence over which the expected sum of zth powers of ranks of objects selected by A is at least $kn^z/2^{z+1}$. Without loss of generality assume that n is even. Let p be the expected number of acceptances prior to the time the $(n/2)$th object is seen (inclusive), in case the ranks of the arriving objects are monotonically increasing. If $p \leq k/2$, then construct a sequence of objects such that the best $n/2$ objects are the first to arrive, and they arrive in order of increasing rank. Clearly, the expected number of objects accepted during the second half is at least $k/2$, and each such object is of rank larger than $n/2$. It thus follows that the average rank of accepted objects is at least $(k/2) \cdot (n/2)^z = kn^z/2^{z+1}$. The case of $p > k/2$ is analogous. The sequence is constructed so that the worst $n/2$ objects are the first to arrive, and they arrive in order of increasing rank. It again follows that the expected sum of z-powers of ranks of accepted objects is at least $kn^z/2^{z+1}$.

TABLE 1

Slots \ Candidates	128	256	512	1024	2048	4096	8192	16384	32768
1	65	127	256	312	317	320	323	324	322
	(35)	(38)	(40)	(40)	(41)	(42)	(42)	(42)	(42)
5	65	127	257	464	473	476	482	489	484
	(16)	(33)	(67)	(87)	(91)	(95)	(97)	(99)	(96)
10	65	129	256	511	633	635	643	647	649
	(11)	(22)	(47)	(91)	(124)	(135)	(139)	(142)	(145)

6. Simulation results. We have performed some simulations of the algorithm. Since our algorithm was designed for the purpose of proving asymptotic results, it does not necessarily give the best values possible. In particular, when the number of candidates is small it seems that much better results could be achieved. Following are some results with numbers of candidates taken as powers of 2, and the number of slots to be filled are 1, 5, and 10, and $z = 1$. In Table 1, we display simulation estimates of the expectation and the standard deviation (in parentheses) of the mean rank of candidates selected by the algorithm for various pairs of candidate and slot numbers.

Acknowledgments. We are indebted to Eugen Dynkin, John Preater, Yossi Rinott, Mike Saks, Steve Samuels, and Robert Banderbei for helpful references.

REFERENCES

[1] M. AJTAI, N. MEGIDDO, AND O. WAARTS *Improved Algorithms and Analysis for Secretary Problems and Generalizations*, manuscript.

[2] N. ALON AND J. H. SPENCER, *The Probabilistic Method*, Wiley-Intersci. Series Discrete Math. Optim., Wiley, New York, 1992.

[3] Y. S. CHOW, H. ROBBINS, S. MORIGUTI, AND S. M. SAMUELS, *Optimal selection based on relative rank (the "secretary problem")*, Israel J. Math., 2 (1964), pp. 81–90.

[4] E. B. DYNKIN, *The optimum choice of the instant for stopping a Markov process*, Sov. Math. Dokl., 4 (1963), pp. 627–629.

[5] T. S. Ferguson, *Who solved the secretary problem?*, Statist. Sci., 4 (1989), pp. 282–296.
[6] P. R. Freeman, *The secretary problem and its extensions: A review*, Internat. Statist. Rev., 51 (1983), pp. 189–206.
[7] J. Gilbert and F. Mosteller, *Recognizing the maximum of a sequence*, J. Amer. Statist. Assoc., 61 (1966), pp. 35–73.
[8] K. S. Glasser, R. Holzsager, and A. Barron, *The d choice secretary problem*, Comm. Statist. C—Sequential Anal., 2 (1983), pp. 177–179.
[9] D. V. Lindley, *Dynamic programming and decision theory*, Appl. Statist., 10 (1961), pp. 39–52.
[10] M. Henke, *Sequentialle Auswahlprobleme bei Unsicherheit*, Anton Hain Verlag, Meisenheim, 1970.
[11] M. L. Nikolaev, *A generalization of the best choice problem*, Theory Probab. Appl., 22 (1977), pp. 187–190.
[12] J. Preater, *On multiple choice secretary problems*, Math. Oper. Res., 19 (1994), pp. 597–602.
[13] H. Rubin and S. M. Samuels, *The finite memory secretary problem*, Ann. Probab., 5 (1977), pp. 627–635.
[14] S. M. Samuels, *Optimal counter strategies for the secretary problem*, IMS Bull. 7, 93 (Abstract 78t-43).
[15] S. M. Samuels, *Secretary problems*, in Handbook of Sequential Analysis, B. K. Gosh and P. K. Sen, eds., Marcel Dekker, New York, 1991, pp. 381–405.
[16] S. M. Samuels, *Secretary problems as a source of benchmark sounds*, in Stochastic Inequalities IMS Lecture Notes Monogr. Ser. 22, Inst. Math. Statist., Hayward, CA, 1992, pp. 371–387.
[17] D. D. Sleator and R. E. Tarjan, *Amortized efficiency of list updates and paging rules*, Comm. ACM, 28 (1985), pp. 202–208.
[18] R. J. Vanderbei, *The optimal choice of a subset of a population*, Math. Oper. Res., 5 (1980), pp. 482–486.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 28–35

LABELING PRODUCTS OF COMPLETE GRAPHS WITH A CONDITION AT DISTANCE TWO*

JOHN P. GEORGES†, DAVID W. MAURO†, AND MELANIE I. STEIN†

Abstract. For integers $j \geq k$, an $L(j,k)$-labeling of a graph G is an integer labeling of the vertices in $V(G)$ such that adjacent vertices receive integers which differ by at least j, and vertices which are distance two apart receive labels which differ by at least k. We determine $\lambda_k^j(K_n \times K_m)$ for all j, k, m, n, and $\lambda_1^2(K_{p^r}^q)$ for $3 \leq q < p$, p prime.

Key words. vertex labeling, λ_k^j-labeling, product of complete graphs, Cayley graph

AMS subject classification. 05C

PII. S0895480199351859

1. Introduction.

An *$L(2,1)$-labeling of a graph G* is a mapping L from $V(G)$ into the integers such that $|L(v_2) - L(v_1)| \geq 2$ if v_1 and v_2 are adjacent in G and $|L(v_2) - L(v_1)| \geq 1$ if v_1 and v_2 are distance two apart in G. Elements of the image of L are called *labels*, and the *span of L* is the difference between the largest and smallest labels of L. The minimum span taken over all $L(2,1)$-labelings of G, denoted $\lambda(G)$, is called the *λ-number of G*, and if L is a labeling with minimum span, then L is called a *λ-labeling of G*. We shall assume with no loss of generality that the minimum label of $L(2,1)$-labelings of G is 0.

A variation of Hale's channel assignment problem [8] in which the difference between frequencies assigned to transmitters is inversely related to the distance between the transmitters, $L(2,1)$-labelings were first studied by Griggs and Yeh [7]. They derived formulas for the λ-number of paths and cycles and established bounds on the λ-number of trees. They also investigated the relationship between $\lambda(G)$ and other graph invariants such as $\chi(G)$ and $\Delta(G)$. Other authors have subsequently contributed to the literature of $L(2,1)$-labelings; see [1], [2], [3], [4], [5], [6], [9], [10], [11], [12], and [13].

In [3], Georges and Mauro considered $L(j,k)$-labelings, a generalization of $L(2,1)$-labelings, where j and k are positive integers, $j \geq k$. Formally, an *$L(j,k)$ labeling of G* is a mapping L from $V(G)$ into the integers such that

(i) $|L(v_2) - L(v_1)| \geq j$ if $d_G(v_1, v_2) = 1$, and

(ii) $|L(v_2) - L(v_1)| \geq k$ if $d_G(v_1, v_2) = 2$.

The *λ_k^j-number of G*, denoted $\lambda_k^j(G)$, is the minimum span over all $L(j,k)$-labelings of G. As before, we shall assume with no loss of generality that the minimum label of $L(j,k)$-labelings is 0. Georges and Mauro considered $L(j,k)$-labelings of Cartesian products of complete graphs in [6].

In this paper, we extend a result of Georges, Mauro, and Whittlesey [5] on λ-labelings (hereafter denoted λ_1^2-labelings) of products of complete graphs. In section 2, we introduce terminology and notation and review existing results. In section 3, we take a group theoretic approach to prove the following theorems.

*Received by the editors December 12, 1999; accepted for publication (in revised form) September 1, 2000; published electronically December 28, 2000.
http://www.siam.org/journals/sidma/14-1/35185.html

†Department of Mathematics, Trinity College, Hartford, CT 06106 (John.Georges@trincoll.edu, David.Mauro@trincoll.edu, Melanie.Stein@trincoll.edu).

THEOREM 3.1. *If $3 \leq q$ and p is prime, then*
(1) $\lambda_1^2(K_{p^r}^q) = p^{2r} - 1$ *if* $r > 1$ *and* $q \leq p$, *and*
(2) $\lambda_1^2(K_p^q) = p^2 - 1$ *if* $q < p$.

THEOREM 3.2. $\lambda_1^1(K_p^p) = \lambda_1^1(K_p^{p+1}) = p^2 - 1$ *if* p *is prime,* $p \geq 3$.

Finally, in section 4, we prove the following theorems.

THEOREM 4.4. *Let j, k, n and m be integers where $2 \leq n < m$ and $j \geq k$. Then*
(1) $\lambda_k^j(K_n \times K_m) = (m-1)j + (n-1)k$ *if* $\frac{j}{k} > n$, *and*
(2) $\lambda_k^j(K_n \times K_m) = (mn-1)k$ *if* $\frac{j}{k} \leq n$.

THEOREM 4.5. *Let j, k, and n be integers where $2 \leq n$ and $j \geq k$. Then*
(1) $\lambda_k^j(K_n^2) = (n-1)j + (2n-2)k$ *if* $\frac{j}{k} > n-1$, *and*
(2) $\lambda_k^j(K_n^2) = (n^2-1)k$ *if* $\frac{j}{k} \leq n-1$.

We also relate λ_k^j-labelings of diameter 2 graphs to r-path coverings.

2. Notation and results. For $1 \leq i \leq q$, let K_{t_i} be a complete graph on t_i vertices. Then the *Cartesian product of* $K_{t_1}, K_{t_2}, \ldots, K_{t_q}$, denoted $\Pi_{i=1}^q K_{t_i} \equiv K_{t_1} \times K_{t_2} \times \cdots \times K_{t_q}$, is the graph whose vertex set is the set of ordered q-tuples whose hth component is an element of $\{0, 1, 2, 3, \ldots, t_h - 1\}$, and whose edge set is the set of pairs of vertices which differ in exactly one component. If $t_i = t$ for all i, then $\Pi_{i=1}^q K_{t_i}$ will be denoted K_t^q.

Let $(G, *)$ be a group and let $\mathcal{A}$ be a subset of G. We call $\mathcal{A}$ a *generating set for* G (alternatively, we say $\mathcal{A}$ *generates* G), if and only if it has the following property: for all $g \in G$, $\exists a_1, \ldots, a_q \in \mathcal{A}$ such that $g = a_1 * a_2 * \ldots * a_q \in G$. (We henceforth notationally suppress the binary operator in group operations.) We define the *Cayley graph of G with respect to the generating set $\mathcal{A}$* to be the graph with vertex set $V = \{g | g \in G\}$ and edge set $E = \{\{g, a_i g\} | a_i \in \mathcal{A} \text{ and } g \in G\}$. We observe that K_t^q is the Cayley graph for the group $G = \mathbf{Z}_t^q$ with respect to the set of generators $\mathcal{A} = \{(a_1, \ldots, a_q) |\ a_i \in \mathbf{Z}_t \text{ and } a_i \neq 0 \text{ for exactly one } i\}$, where the vertices of K_t^q are identified with the elements of G. Recalling that the length $l_{\mathcal{A}}(g)$ of nonidentity element g in G is the minimum number of elements of $\mathcal{A}$ whose product is g (by convention, 0 if g is the identity), we have that $l_{\mathcal{A}}(g)$ equals the distance in K_t^q from the identity vertex to the vertex g. Similarly, defining $d(g, h)$ to be $l_{\mathcal{A}}(hg^{-1})$ for $g, h \in G$, then this distance is simply the distance between the vertices g and h in the graph.

We next note the following theorem from [3].

THEOREM 2.1. *For all graphs G,*
(1) *there exists a λ_k^j-labeling of G such that each label is of the form $\alpha j + \beta k$, $\alpha, \beta \geq 0$, and*
(2) *for any positive integer c, $c\lambda_k^j(G) = \lambda_{ck}^{cj}(G)$.*

An $L(j, k)$-labeling L is called *k-separated* if and only if any two distinct vertices receive labels under L which differ by at least k. As well, a set of integers is called *k-separated* if and only if any two distinct elements of the set differ by at least k.

For $r \geq 2$, we define the *r-path on n vertices*, denoted ${}_rP_n$, to be the graph with vertex set $V({}_rP_n) = \{x_1, x_2, \ldots, x_n\}$ and edge set $E({}_rP_n) = \{x_i x_j | 1 \leq |i-j| \leq r-1\}$. We note that if $n \leq r$, then ${}_rP_n$ is isomorphic to K_n.

Let H be a graph with vertex set $V(H) = \{v_1, v_2, \ldots, v_n\}$ and edge set $E(H)$. Then the *complement of H*, denoted H^c, is the graph with vertex set $V(H)$ and edge set $\{\{v_i, v_j\} | \{v_i, v_j\} \notin E(H)\}$. Additionally, a partition $\mathcal{C} = \{X_1, X_2, \ldots, X_d\}$ of $V(H)$ is called an *r-path covering* of H if and only if for each i, $1 \leq i \leq d$, the subgraph of H induced by X_i contains a spanning r-path. A *minimum r-path covering*

of H is an r-path covering of H of minimum cardinality, and if $c_r(H)$ is the cardinality of a minimum r-path covering of H, then $c_r(H)$ is called the *r-path covering number of* H. We note that if $c_r(H) = 1$, then H is said to have a *Hamilton r-path*. For the case $r = 2$, if $c_2(H) = 1$, then H is simply said to have a Hamilton path.

In [5], Georges, Mauro, and Whittlesey proved the following result on general 2-path coverings.

THEOREM 2.2. *Let G be a graph with $|V(G)| = n$. Then*

(1) $\lambda_1^2(G) \leq n-1$ *if and only if* $c_2(G^c) = 1$, *and*

(2) $\lambda_1^2(G) = n + c_2(G^c) - 2$ *if and only if* $c_2(G^c) \geq 2$.

This result, coupled with the observation that $\lambda_1^2(G) \geq |V(G)| - 1$ if G has diameter 2, led to the following theorem also in [5].

THEOREM 2.3. *For $2 \leq m \leq n$, $\lambda_1^2(K_n \times K_m) = mn - 1$, except for the case $m = n = 2$, in which case $\lambda_1^2(K_2 \times K_2) = 4$.*

3. On $\lambda_1^2(K_{p^r}^q)$: A Cayley graph approach. As noted in section 2, $K_{p^r}^q$ may be identified with the Cayley graph of the group $G = Z_{p^r}^q$ with respect to the set of generators $\mathcal{A} = \{(a_1, a_2, \ldots, a_q) | a_i \in Z_{p^r} \text{ and } a_i \neq 0 \text{ for exactly one } i\}$. We use this identification to prove the following theorem.

THEOREM 3.1. *If $3 \leq q$ and p is prime, then*

(1) $\lambda_1^2(K_{p^r}^q) = p^{2r} - 1$ *if $r > 1$ and $q \leq p$, and*

(2) $\lambda_1^2(K_p^q) = p^2 - 1$ *if $q < p$.*

Proof of (1). To obtain a lower bound for $\lambda_1^2(K_{p^r}^q)$, notice that the subgraph induced by the set of vertices of the form $(a_1, a_2, 0, \ldots, 0)$ has diameter 2. Thus every $L(2,1)$-labeling of $K_{p^r}^q$ must assign distinct labels to the p^{2r} vertices of the subgraph, implying $\lambda_1^2(K_{p^r}^q) \geq p^{2r} - 1$ for $q \geq 2$. We therefore need only establish that $p^{2r} - 1$ is an upper bound for $\lambda_1^2(K_{p^r}^q)$. We show this by constructing a labeling with span $p^{2r} - 1$.

Our strategy will be to find a subgroup $H \subset G$ satisfying the following properties:

(i) $|H| = (p^r)^{q-2}$;

(ii) $h \in H \Rightarrow l_{\mathcal{A}}(h) \geq 3$ or $h = (0, \ldots, 0)$;

(iii) there exist two cosets $g_1 H$ and $g_2 H$ which generate the quotient group G/H, and $g_1 H \cap \mathcal{A} = g_2 H \cap \mathcal{A} = \emptyset$.

We claim that given such a subgroup H, we can label the vertices of the graph, which are identified with the elements of G, using the labels $\{0, \ldots, p^{2r} - 1\}$ by assigning the same label to all group elements within one H-coset. We note that if gh and gh' are in the same coset with $h \neq h'$, then $d(gh, gh') = l_{\mathcal{A}}(gh'(gh)^{-1}) = l_{\mathcal{A}}(h'h^{-1}) \geq 3$ by property (ii), and hence all group elements in a given coset may be assigned the same label. We next assign labels from 0 to $p^{2r} - 1$, in order, to the cosets by ordering them as follows:

$$H,\ g_1 H,\ g_1^2 H,\ \ldots, g_1^{p^r - 1} H;$$

$$g_2 g_1^{-1} H,\ g_2 H,\ g_2 g_1 H,\ \ldots, g_2 g_1^{p^r - 2} H;$$

$$g_2^2 g_1^{-2} H,\ g_2^2 g_1^{-1} H,\ \ldots, g_2^2 g_1^{p^r - 3} H;$$

$$\vdots$$

$$g_2^{p^r-1}g_1H,\ g_2^{p^r-1}g_1^2H,\ \ldots, g_2^{p^r-1}H.$$

We must check that group elements in adjacent cosets in this ordering are not adjacent vertices in the graph. Choose $a \in aH$ and $b \in bH$ where aH and bH are adjacent cosets in the list above. Then $d(a,b) = l_{\mathcal{A}}(g_1h)$ or $d(a,b) = l_{\mathcal{A}}(g_2h)$ for some $h \in H$. However, by property (iii), $l_{\mathcal{A}}(g_1h) \geq 2$ and $l_{\mathcal{A}}(g_2h) \geq 2$, implying that a and b are not adjacent.

To complete the proof of (1), we construct a subgroup H which satisfies all three conditions as long as $q \leq p$.

Consider the map $\varphi : G \to (\mathbf{Z}_{p^r})^2$, defined by

$$\varphi((a_1,\ldots,a_q)) = \left(\sum_{i=1}^{q} a_i, \sum_{i=1}^{q} ia_i\right).$$

We set $H = \mathrm{Ker}(\varphi)$. Since $\varphi((2x-y, y-x, 0, \ldots, 0)) = (x,y)$, then φ is surjective, implying that $|\mathrm{Ker}(\varphi)| = (p^r)^{q-2}$; therefore H satisfies property (i).

Now suppose that $g \in \mathcal{A} \cap H$. Since g has length 1, $\varphi(g) = (a_j, ja_j)$ for some j. However, since $g \in H$, then $a_j = 0 \Rightarrow g = (0,\ldots,0)$. Thus $H \cap \mathcal{A} = \emptyset$, and H contains no elements with length 1.

To establish the second property, we must also show that H is free of not only length 1 elements, but of length 2 elements as well, to ensure that no two vertices at distance two receive the same label. First note that no two distinct elements of $\mathcal{A}$ can have the same image under φ; otherwise, if g and g' are distinct elements in $\mathcal{A}$ such that $\varphi(g) = \varphi(g')$, then $(a_i, ia_i) = (a_j, ja_j)$, implying that $a_i = a_j$ and $a_i(i-j) = 0 \bmod p^r$. Since a_i is nonzero, $i-j$ must be divisible by p. Now $|i-j| \leq q-1$, so if $q \leq p$, then $|i-j| < p$. This implies $i = j$, in which case $g = g'$.

Now suppose $h \in H$ and $l_{\mathcal{A}}(h) = 2$. Then $h = gg'$ where g and g' each have length 1, and $\varphi(h) = 0 \Rightarrow \varphi(g) = \varphi((g')^{-1})$. Now since no two length one elements have the same image, $g = (g')^{-1}$, so $h = (0,\ldots,0)$. Therefore if $h \in H$ and $h \neq (0,\ldots,0)$, then $l_{\mathcal{A}}(h) \geq 3$, thus satisfying property (ii).

Property (iii) holds by a counting argument. Note that $|\mathcal{A}| = (p^r-1)q$ and $|G/H| = p^{2r}$. No two elements of $\mathcal{A}$ can be in the same coset, since H contains no elements of length 2. Therefore if $\mathcal{S}$ is the set of cosets which are distinct from H and contain no elements of $\mathcal{A}$, then $|\mathcal{S}| = p^{2r} - (p^r-1)q - 1$.

To ensure that the cosets in property (iii) exist, we must show that there are two cosets in $\mathcal{S}$, each of order p^r in G/H, which generate G/H. We verify that this must be true by counting the number of cosets which cannot generate G/H together with a fixed coset of order p^r. Let x be the fixed coset, and let y be any other coset which, together with x, generates G/H. Then the cosets which cannot generate G/H together with x are those of the form $x^a y^b$ where $0 \leq a \leq p^r - 1$ and $0 \leq b \leq p^r - 1$ and b is relatively prime to p. Hence we have p^r choices for a and p^{r-1} choices for b, and the number of cosets not suitable for generating G/H together with x is $p^r p^{r-1}$.

If the number of cosets in $\mathcal{S}$ is greater than the number of cosets which can never be part of a generating pair together with a given coset, then there will always be a coset in $\mathcal{S}$ left to complete a generating pair. Thus we require

$$p^{2r} - (p^r-1)q - 1 > p^r p^{r-1} - 1$$

or alternatively,

$$q < \frac{p^{2r} - p^{2r-1}}{p^r - 1}.$$

If $r > 1$, we have

$$q < p^r - p^{r-1} + \frac{p^r - p^{r-1}}{p^r - 1},$$

a number greater than p. Hence if $q \leq p$, the requirement is satisfied, and H satisfies property (iii).

Proof of (2). We note that the above proof applies until the final counting argument, and that if $r = 1$, the required inequality simplifies to $q < p$. □

Theorem 3.1 can be used to establish bounds on $\lambda_1^2(K_t^q)$ for arbitrary t and q. If $w = \max\{q, t\}$ and if $p > w$ where p is a prime, then $\lambda_1^2(K_t^q) \leq \lambda_1^2(K_w^w) \leq \lambda_1^2(K_p^w) = p^2 - 1$. For the particular case $p_1^{r_1} \leq t \leq p_2^{r_2}$ and $3 \leq q < \min\{p_1, p_2\}$, where p_1 and p_2 are primes, we have

$$p_1^{2r_1} - 1 = \lambda_1^2(K_{p_1{}^{r_1}}^q) \leq \lambda_1^2(K_t^q) \leq \lambda_1^2(K_{p_2{}^{r_2}}^q) = p_2^{2r_2} - 1.$$

Noting that $p^2 - 1 \leq \lambda_1^1(K_p^q) \leq \lambda_1^2(K_p^q) = p^2 - 1$ for p prime and $3 \leq q < p$, we can use the strategy of Theorem 3.1 to show the following theorem.

THEOREM 3.2. $\lambda_1^1(K_p^p) = \lambda_1^1(K_p^{p+1}) = p^2 - 1$ *if* p *is prime,* $p \geq 3$.

Proof. To show that $\lambda_1^1(K_p^p) = p^2 - 1$, we proceed as in the previous proof, recalling that property (iii) is not relevant in this case.

To show that $\lambda_1^1(K_p^{p+1}) = p^2 - 1$, proceed again as in the previous proof, but define φ via

$$\varphi((a_1, \ldots, a_{p+1})) = \left(a_p + \sum_{i=1}^{p-1} a_i, a_{p+1} + \sum_{i=1}^{p-1} i a_i \right).$$

One verifies as before that φ is surjective, that no length 1 element is in $\mathrm{Ker}(\varphi)$, and that no two length 1 elements have the same image. It follows that $\mathrm{Ker}(\varphi)$ is also free of length 2 elements. Therefore labeling each coset with a different number between 0 and $p^2 - 1$ gives a λ_1^1-labeling. □

4. On $\lambda_k^j(K_n \times K_m)$. In this section we determine the λ_k^j-number of the product of two complete graphs. We shall also explore the relationship between λ_k^j-labelings of diameter 2 graphs and r-path covering numbers of their complements.

We shall begin by presenting several lemmas which establish lower bounds on λ_k^j-numbers of these products.

LEMMA 4.1. $\lambda_k^j(K_n \times K_m) \geq (mn - 1)k$.

Proof. Since the order of $K_n \times K_m$ is mn and since every (j, k)-labeling of $K_n \times K_m$ is k-separated, the result follows. □

LEMMA 4.2. *If* $2 \leq n \leq m$, *then* $\lambda_k^j(K_n \times K_m) \geq (m-1)j + (n-1)k$.

Proof. Suppose to the contrary that $\lambda_k^j(K_n \times K_m) < \alpha \equiv (m-1)j + (n-1)k$. Let L be a (j, k)-labeling of $K_n \times K_m$ with span $s(L) \leq \alpha - 1$, and consider the partition $\{X_1, X_2, \ldots, X_m\}$ of $[0, \alpha - 1]$, where $X_i = [(i-1)j, ij - 1]$ for $1 \leq i \leq m - 1$, and $X_m = [(m-1)j, \alpha - 1]$. Since L is necessarily k-separated, we observe that at most

$n-1$ vertices of $K_n \times K_m$ are assigned labels from X_m. It follows that at least $n(m-1)+1$ vertices are given labels from $X_1 \bigcup X_2 \bigcup \cdots \bigcup X_{m-1}$, and that there exists some i_0, $1 \leq i_0 \leq m-1$ such that at least $n+1$ vertices are assigned labels from X_{i_0} under L. However, in any collection of $n+1$ vertices of $K_n \times K_m$, there exists a pair of adjacent vertices, implying that their respective labels differ by at least j. This, however, is impossible since the span of interval X_{i_0} is $j-1$. □

LEMMA 4.3. $\lambda_k^j(K_n^2) \geq (n-1)j + (2n-2)k$.

Proof. Suppose to the contrary that $\lambda_k^j(K_n^2) < \beta \equiv (n-1)j + (2n-2)k$. Let L be a (j,k)-labeling of K_n^2 with span less than or equal to $\beta - 1$, and let $\{X_1, X_2, \ldots, X_n\}$ be a partition of the interval $[0, \beta-1]$ such that for $1 \leq i \leq n-1$, $X_i = [(i-1)(j+k), i(j+k)-1]$ and $X_n = [(n-1)(j+k), (n-1)j + (2n-2)k - 1]$. Since L is a k-separated labeling, at most $n-1$ vertices may receive labels from X_n. It follows that at least $n^2 - n + 1$ vertices are given labels from $X_1 \bigcup X_2 \bigcup \cdots \bigcup X_{n-1}$, implying that there exists $i_0, 1 \leq i_0 \leq n-1$, such that at least $n+1$ vertices receive labels from interval X_{i_0} under L. However, in any collection of $n+1$ vertices of K_n^2, there are at least two vertices on a common row and at least two vertices on a common column. These vertices induce a subgraph W isomorphic to either P_3 or two copies of K_2. Thus the k-separation of L requires the span of L restricted to W to be at least $j+k$. Since the length of X_{i_0} is $j+k-1$, we have that not all labels used on W can come from X_{i_0}, a contradiction. □

We are now ready to present the two main results of this section.

THEOREM 4.4. *Let j, k, n, and m be integers where $2 \leq n < m$ and $j \geq k$. Then*

(1) $\lambda_k^j(K_n \times K_m) = (m-1)j + (n-1)k$ *if* $\frac{j}{k} > n$, *and*

(2) $\lambda_k^j(K_n \times K_m) = (mn-1)k$ *if* $\frac{j}{k} \leq n$.

Proof of (1). By Lemma 4.2, it suffices to exhibit a (j,k)-labeling of $K_n \times K_m$ with span $(m-1)j + (n-1)k$. To that end, let labeling L be given by $L((a,b)) = j[(a-b) \bmod m] + ak$. Then clearly the span of L is $(m-1)j + (n-1)k$; hence it suffices to show that L satisfies the distance conditions.

Let $v_1 = (a_1, b_1)$ and $v_2 = (a_2, b_2)$ be elements of $V(K_n \times K_m)$ and let d_1 and d_2 denote $(a_1 - b_1) \bmod m$ and $(a_2 - b_2) \bmod m$, respectively, where with no loss of generality, $d_2 - d_1 \geq 0$.

To show that the distance two condition is satisfied, we suppose $a_1 \neq a_2$ and $b_1 \neq b_2$. If $d_2 - d_1 = 0$, then $|L((a_2,b_2)) - L((a_1,b_1))| = |j(d_2 - d_1) + k(a_2 - a_1)| = k|a_2 - a_1| \geq k$. If $d_2 - d_1 > 0$, then $L((a_2,b_2)) - L((a_1,b_1)) = j(d_2 - d_1) + k(a_2 - a_1) \geq j + k(a_2 - a_1) \geq j + k(1-n) \geq kn + k(1-n) = k$.

To show that the distance one condition is satisfied, we assume that $a_1 = a_2$ and $b_1 \neq b_2$, or $a_1 \neq a_2$ and $b_1 = b_2$.

If $a_1 = a_2$ and $b_1 \neq b_2$, then $L((a_2,b_2)) - L((a_1,b_1)) = j(d_2 - d_1) \geq j$.

If $a_1 \neq a_2$ and $b_1 = b_2$, then $L((a_2,b_2)) - L((a_1,b_1)) = j(d_2 - d_1) + k(a_2 - a_1)$, where, by an argument similar to that above, $d_2 - d_1 \geq 1$. If $d_2 - d_1 = 1$, then $a_2 - a_1 = 1$ or $a_2 - a_1 = 1 \pm m$, the latter of which is impossible since $1 - m < 1 - n \leq a_2 - a_1 \leq n - 1 < m - 1$. Hence $L((a_2,b_2)) - L((a_1,b_1)) = j + k$. If $d_2 - d_1 > 1$, then $L((a_2,b_2)) - L((a_1,b_1)) \geq 2j + k(a_2 - a_1) \geq 2j + k(1-n) \geq j + nk + k(1-n) = j + k$.

Proof of (2). By Lemma 4.1, it suffices to exhibit a (j,k)-labeling of $K_n \times K_m$ with span $(mn-1)k$. However, it can be easily verified that the image of $f(a,b) = ((n+1)a - nb) \bmod mn$ has cardinality mn as a and b range over the integers in $[0, n-1]$ and $[0, m-1]$, respectively. Thus the labeling L given by $L((a,b)) = k[((n+1)a - nb) \bmod mn]$ is a (j,k)-labeling with span $k(mn-1)$. □

THEOREM 4.5. *Let j, k and n be integers where $2 \leq n$ and $j \geq k$. Then*
(1) $\lambda_k^j(K_n^2) = (n-1)j + (2n-2)k$ *if* $\frac{j}{k} > n-1$, *and*
(2) $\lambda_k^j(K_n^2) = (n^2-1)k$ *if* $\frac{j}{k} \leq n-1$.

Proof. To prove (1), we appeal to Lemma 4.3 and let $L((a,b))$ be the vertex labeling given by $j[(b-a) \bmod n] + ak + k[(b-a) \bmod n]$. To prove (2), we appeal to Lemma 4.1 and let $L((a,b))$ be the vertex labeling given by $k[(na-(n-1)b) \bmod n^2]$. Details are omitted. □

From Theorem 4.5, $\lambda_3^5(K_3^2) = 24$ and $\lambda_3^8(K_3^2) = 28$, and from Theorem 4.4, $\lambda_3^8(K_3 \times K_4) = 33$ and $\lambda_3^{10}(K_3 \times K_4) = 36$. In Figure 4.1(1)–(4) we exhibit labelings with these spans.

0	9	18
21	3	12
15	24	6

(1) λ_3^5-labeling of K_3^2

0	11	22
25	3	14
17	28	6

(2) λ_3^8-labeling of K_3^2

0	27	18	9
12	3	30	21
24	15	6	33

(3) λ_3^8-labeling of $K_3 \times K_4$

0	30	20	10
13	3	33	23
26	16	6	36

(4) λ_3^{10}-labeling of $K_3 \times K_4$

FIG. 4.1.

For each labeling exhibited in Figure 4.1, tracking labels in ascending order reveals a general relationship between λ_k^j-labelings of $G = K_n \times K_m$ and r-path covering numbers of G^c. Noting that the independence number $\alpha(K_n \times K_m)$ of $K_n \times K_m$ is $\min\{m,n\}$, we have the following results.

THEOREM 4.6. *Let $2 \leq n < m$. Then*
(1) *if $r > n$, then $c_r((K_n \times K_m)^c) = m$, and*
(2) *if $2 \leq r \leq n$, then $c_r((K_n \times K_m)^c) = 1$.*

Proof of (1). Since $\alpha(K_n \times K_m) = n$, the largest clique in $(K_n \times K_m)^c$ has n vertices. It thus follows that for any r-path covering $\mathcal{C}$ of $(K_n \times K_m)^c$, the length of its largest r-path is at most n, implying that $|\mathcal{C}| \geq \frac{mn}{n} = m$. We now construct an r-path covering $\mathcal{C}' = \{X_0, X_1, \ldots, X_{m-1}\}$ of $(K_n \times K_m)^c$ with cardinality m by using the labeling which appears in Theorem 4.4: for all i, $0 \leq i \leq m-1$, $(a,b) \in X_i$ if and only if $(a-b) \bmod m = i$.

Proof of (2). We construct a Hamilton r-path $v_0, v_1, v_2, \ldots, v_{mn-1}$ as follows: $v_i = (a,b)$, where $i = ((n+1)a - nb) \bmod mn$. □

In a manner similar to that used in the proof of the previous theorem, we can apply the labelings used in the proof of Theorem 4.5 to establish the following result.

THEOREM 4.7. *Let $2 \leq n$.*
(1) *If $r > n-1$, then $c_r((K_n^2)^c) = n$, and*
(2) *If $r \leq n-1$, then $c_r((K_n^2)^c) = 1$.*

In closing, we present several results which hold true for arbitrary diameter 2 graphs.

LEMMA 4.8. *Let G be a graph with order n and diameter 2. Then $\lambda_k^{\lceil \frac{j}{k} \rceil k}(G) = (n-1)k$ if $\lambda_k^j(G) = (n-1)k$.*

Proof. Let $j = ak + t$, $0 \leq t < k$. The lemma is clearly true if $t = 0$. Thus let $t > 0$ and let L be a λ_k^j-labeling of G. Since G has diameter 2, L is k-separated and each label under L is necessarily of the form αk for some α between 0 and $n-1$. If

L is not a $\lambda_k^{(a+1)k}$-labeling of G, then there exist adjacent vertices u and v in $V(G)$ such that $|L(u) - L(v)| < (a+1)k$.

Let $L(u) = b_1 k$ and $L(v) = b_2 k$. Since L is an $L(j,k)$-labeling of G, we have

$$ak < j \leq |L(u) - L(v)| = |b_1 - b_2|k < (a+1)k,$$

implying the contradiction that $a < |b_1 - b_2| < a + 1$. □

We point out that in accord with Lemma 4.8, the labelings appearing in Figures 4.1(1) and 4.1(3) are, respectively, a λ_3^6-labeling of K_3^2 and a λ_3^9-labeling of $K_3 \times K_4$.

THEOREM 4.9. *Let G be a graph with order n and diameter 2.*

(1) *If $\lambda_k^j(G) = (n-1)k$, then for all r, $2 \leq r \leq \lceil \frac{j}{k} \rceil$, G^c has a Hamilton r-path, and*

(2) *if G^c has a Hamilton r-path, then for all j, k, where $\frac{j}{k} \leq r$, $\lambda_k^j(G) = (n-1)k$.*

Proof of (1). Let $d = \lceil \frac{j}{k} \rceil$. Then by Lemma 4.8, $\lambda_k^{dk}(G) = (n-1)k$.

Let L be a λ_k^{dk}-labeling of G. We construct a Hamilton d-path ${}_dP_n = v_0, v_1, \ldots, v_{n-1}$ in G^c by letting v_i denote the vertex such that $L(v_i) = ik$. However, for all r, $2 \leq r \leq d$, a Hamilton d-path has a Hamilton r-path as a subgraph.

Proof of (2). Since G has diameter 2, every $L(j,k)$-labeling of G is k-separated, and therefore, $\lambda_k^j(G) \geq (n-1)k$. Let ${}_rP_n = v_0, v_1, \ldots, v_{n-1}$ be a Hamilton r-path in G^c. We construct the $L(j,k)$-labeling L given by $L(v_i) = ik$, $0 \leq i \leq n-1$. □

Acknowledgments. The authors extend their appreciation to the referees for their many suggestions which resulted in an improved paper.

REFERENCES

[1] G. J. CHANG AND D. KUO, *The L(2,1)-labeling problem on graphs*, SIAM J. Discrete Math., 9 (1996), pp. 309–316.

[2] J. P. GEORGES AND D. W. MAURO, *On the criticality of graphs labeled with a condition at distance two*, Congr. Numer., 101 (1994), pp. 33–49.

[3] J. P. GEORGES AND D. W. MAURO, *Generalized vertex labelings with a condition at distance two*, Congr. Numer., 109 (1995), pp. 141–159.

[4] J. P. GEORGES AND D. W. MAURO, *On the size of graphs labeled with a condition at distance two*, J. Graph Theory, 22 (1996), pp. 47–57.

[5] J. P. GEORGES, D. W. MAURO, AND M. A. WHITTLESEY, *Relating path coverings to vertex labelings with a condition at distance two*, Discrete Math., 135 (1994), pp. 103–111.

[6] J. P. GEORGES AND D. W. MAURO, *Some results on λ_k^j-numbers of the products of complete graphs*, Congr. Numer., 140 (1999), pp. 141–160.

[7] J. R. GRIGGS AND R. K. YEH, *Labelling graphs with a condition at distance* 2, SIAM J. Discrete Math., 5 (1992), pp. 586–595.

[8] W. K HALE, *Frequency assignment: Theory and application*, Proc. IEEE, 68 (1980), pp. 1497–1514.

[9] J. VAN DEN HEUVEL, R. A. LEESE, AND M. A. SHEPHERD, *Graph labeling and radio channel assignment*, J. Graph Theory, 29 (1998), pp. 263–283.

[10] K. JONAS, *Graph Coloring Analogues with a Condition at Distance Two: L(2,1)-Labelings and List λ-Labelings*, Ph.D. thesis, University of South Carolina, Columbia, SC, 1993.

[11] D. LIU AND R. K. YEH, *On distance two labelings of graphs*, Ars Combin., 47 (1997), pp. 13–22.

[12] D. SAKAI, *Labeling chordal graphs: Distance two condition*, SIAM J. Discrete Math., 7 (1994), pp. 133–140.

[13] M. A. WHITTLESEY, J. P. GEORGES, AND D. W. MAURO, *On the λ-number of Q_n and related graphs*, SIAM J. Discrete Math., 8 (1995), pp. 499–506.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 36–48

SUFFICIENT CONDITIONS FOR TWO TREE RECONSTRUCTION TECHNIQUES TO SUCCEED ON SUFFICIENTLY LONG SEQUENCES*

MIKE STEEL†

Abstract. The reconstruction of evolutionary trees (phylogenies) from DNA sequence data is a central problem in biology. We describe simple sufficient conditions for two tree reconstruction methods (maximum parsimony and maximum compatibility) to correctly reconstruct a tree when applied to sufficiently many sequence sites generated under a simple stochastic model.

Key words. trees, genetic sequences, maximum parsimony method, stochastic models

AMS subject classifications. 05C05, 92D15

PII. S0895480198343571

1. Introduction.

In biology, (graph-theoretic) trees are widely used to represent the evolutionary relationship between a group of extant species. Such a tree is sometimes called a "phylogeny" or "evolutionary tree." The extant species comprise the set $\mathcal{L}$ of leaves (vertices of degree 1) and the tree T describes the evolutionary history of the species from some hypothetical ancestor (located on some edge of the tree). Ideally, each vertex of T that is not in $\mathcal{L}$ has degree 3, in which case T is said to be *fully resolved*.

An important task in biology is to reconstruct such trees from observed features or data describing the extant species. We can regard each item of data as a function from $\mathcal{L}$ into some set R of r states (where $r = |R|$). Such functions are called r-state *characters* and they correspond to characteristics (morphological, physiological, genetic) on which the extant species differ. For example, in genetics, each site in a collection of aligned DNA sequences (one for each extant species, and with the same number of aligned sites for each species) provides a 4-state (or 2-state) character. For further biological details the interested reader is referred to [13].

The *maximum parsimony* method (abbreviated $\mathcal{MP}$) is a very popular technique for reconstructing evolutionary trees from collections of characters. To each character f and each tree T we associate a value $L(f,T)$ which is the minimum number of edges that must be assigned different states to its endpoints in order to extend f to assign states from R to all the vertices of T (we call the corresponding function $g : V \to R$ an *extension* of f). These concepts are illustrated in Figure 1 and will be defined more rigorously in the next section.

The $\mathcal{MP}$ method selects the tree (or trees) T that minimizes the sum of $L(f,T)$ over the characters f in the data. Informally, such a tree minimizes the number of "mutations" (changes of state across the edges of the tree) that need to be hypothesized in order to explain how the characters could have all evolved on tree T from some ancestral vertex.

*Received by the editors August 14, 1998; accepted for publication (in revised form) September 5, 2000; published electronically December 28, 2000. This research was supported by the New Zealand Marsden Fund.

http://www.siam.org/journals/sidma/14-1/34357.html

†Biomathematics Research Centre, University of Canterbury, Private Bag 4800, Christchurch, New Zealand (m.steel@math.canterbury.ac.nz).

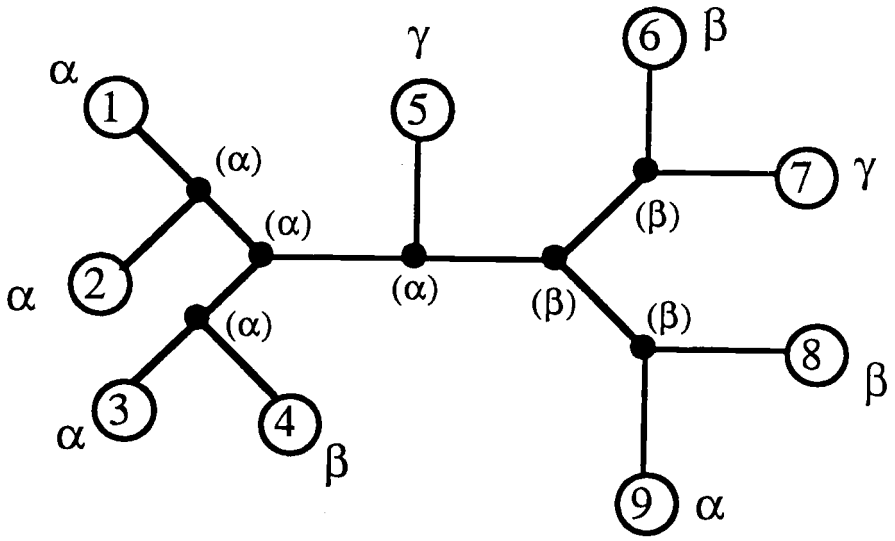


FIG. 1. *A fully resolved tree on leaf set* $\mathcal{L} = \{1, \ldots, 9\}$*, together with a* 3*-state character* $f : \mathcal{L} \to \{\alpha, \beta, \gamma\}$ *having* $L(f, T) = 5$*. An example of an extension* g *of* f *with* $\mathrm{ch}(g, T) = 5$ *is given by the additional assignment of states shown in brackets.*

The problem of finding an $\mathcal{MP}$ tree for a given sequence of characters is NP-hard and is a special case of the Steiner problem in graph theory (see [5] for details).

We will also consider a related method, called *maximum compatibility* denoted $\mathcal{MC}$. Informally, this method selects a tree that maximizes the number of characters in the data that could have evolved on the tree from the hypothetical ancestor without any parallel or reverse mutations (a more precise definition is given in the next section).

A fundamental theoretical question is to determine conditions under which $\mathcal{MP}$ or $\mathcal{MC}$ would recover a tree when applied to a large number of characters that evolved independently on that tree, according to some stochastic model. A variety of Markov-style models have been proposed for modeling and analyzing the evolution of DNA sequences (see [13]). The simplest such model, which we will call the N_r model, assigns equal probability to all possible transitions amongst the r states in R. This is as familiar to geneticists as the *Jukes–Cantor model* in the case $r = 4$ [7].

In a landmark paper [4], Felsenstein investigated whether $\mathcal{MP}$ and $\mathcal{MC}$ would tend to select the correct tree if the underlying characters were generated by this model. He showed that there exist various parameter settings in the N_2 model for which $\mathcal{MP}$ and $\mathcal{MC}$ will select an incorrect tree with probability tending to 1 as the sequence length tends to infinity. Indeed, this occurs even when T has just four leaves (in which case $\mathcal{MP}$=$\mathcal{MC}$). This statistical inconsistency phenomena has subsequently been refined and extended by others [6], [8], [11]. Sufficient conditions for the statistically consistency of $\mathcal{MP}$ or $\mathcal{MC}$ have only been described when either T has just four leaves [10], or for special cases [6], [8], [11], [12].

In this paper we provide the first explicit sufficient conditions for the statistical consistency of $\mathcal{MP}$ (resp., $\mathcal{MC}$) that are applicable to any tree on any number of leaves under the N_r (resp., N_2) models. Essentially, our results are of the form that

if the mutation probabilities associated to the edges of a tree under these models are sufficiently small, and not too unequal across the tree, then the two methods are statistically consistent. We now formalize some of the concepts described above.

2. Preliminaries. We begin by formalizing the definition of the parsimony score of a character $f : \mathcal{L} \to R$ on a tree $T = (V, E)$, where $\mathcal{L}$ is set of leaves (degree 1 vertices) of T.

Given a function $g : V \to R$, the *changing number* of g is defined by $\mathrm{ch}(g, T) := |\{e = \{u, v\} \in E : g(u) \neq g(v)\}|$. Given a character $f : \mathcal{L} \to R$, the *parsimony score* of f on T is defined by

$$L(f, T) := \min_g \{\mathrm{ch}(g, T) : g|\mathcal{L} = f\},$$

where we adopt the convention throughout this paper that $g|\mathcal{L}$ denotes the restriction of g to $\mathcal{L}$. It is easily shown that $L(f, T) \geq |f(\mathcal{L})| - 1$ and thus the quantity

$$H(f, T) := L(f, T) - |f(\mathcal{L})| + 1 \tag{2.1}$$

is nonnegative. $H(f, T)$ is sometimes called the *homoplasy* (or number of "extra steps") of f relative to T. This quantity turns out to be useful in the analysis of maximum parsimony, and it is also the basis of another method we describe now.

First, note that $H(f, T) = 0$ precisely if there is an extension g of f to V for which the subset of V that is mapped to any state in R by g forms a connected subtree of T (in biology this corresponds to being able to describe the evolution of f on T without any reverse or parallel changes of states). The maximum compatibility method, $\mathcal{MC}$, selects the tree (or trees) T that maximizes the number of characters f in that data for which $H(f, T) = 0$.

To investigate the statistical properties of these two methods we also need to specify a model by which characters can "evolve" on trees. In this paper we consider the simplest such model, namely, the symmetric r-state model, due to Neyman [9] (see also, [1], [3], [4]) and abbreviated hereafter as the N_r model. In this model R has cardinality r and the model has as its underlying parameters a fully resolved tree $T = (V, E)$ and a map $p : E \to (0, \frac{r-1}{r})$ that associates to each edge e of T a corresponding *mutation probability* $p(e)$. We now describe how this model assigns states to the vertices of T.

First we select an arbitrary fixed vertex v_0 of T and direct all edges of T away from v_0. We then randomly assign, with uniform probability, an element of R to v_0, and then assign states to the remaining vertices recursively as follows: for any arc (u, v) for which u has been assigned a state but v has not yet been assigned a state, randomly assign v the same state as u with probability $1 - p(e)$ (where $e = \{u, v\}$) or assign v one of the other states in R with equal probability (viz, $\frac{p(e)}{r-1}$).

In this way we generate a random function $G : V \to R$. Under the N_r model, with parameters (T, p), let $\mathbb{P}(G = g)$ be the probability that $G = g$, and let $\mathbb{P}(f, T)$ denote the probability that $G|\mathcal{L} = f$ (i.e., that G restricted to $\mathcal{L}$ equals f). By definition, and the assumptions of the model,

$$\mathbb{P}(f, T) = \sum_{\{g: V \to R: g|\mathcal{L} = f\}} \mathbb{P}(G = g)$$

and

$$\mathbb{P}(G = g) = \frac{1}{r} \prod_{\{e=\{u,v\}: g(u) \neq g(v)\}} \frac{p(e)}{r-1} \prod_{\{e=\{u,v\}: g(u) = g(v)\}} (1 - p(e)),$$

from which we immediately see that the probability distribution on characters f (and extensions g) is independent of our choice of v_0.

A tree reconstruction method is *statistically consistent* under this N_r model with underlying parameters (T, p) if the probability that the method reconstructs T when applied to k independently generated characters converges to 1 as k tends to infinity. An example of such a method is the maximum likelihood technique, as Chang [2] recently established (for the N_r model and generalizations thereof).

3. Sufficient conditions for correct tree reconstruction. We begin with some definitions leading to a simple combinatorial sufficient condition for the two methods described to return a given tree.

An *interior edge* of T is an edge that is not incident with a leaf. We will let $\mathring{E}$ denote the set of interior edges of T. Deleting an edge $e \in E$ from T produces a partition π_e of the leaves of T into two subsets. Note also that each character f induces a partition of $\mathcal{L}$ by grouping together those leaves that are assigned the same state by f. If the partition induced by f equals π_e, we say that f *corresponds to* edge e. Note that f corresponds to some edge of T if and only if $L(f, T) = 1$. Let $c(e)$ denote the set of those $r(r-1)$ characters that correspond to edge e and let $c(T) = \cup_{e \in \mathring{E}} c(e)$.

Suppose we are given a sequence C of characters and a character f. Let $n(C, f)$ denote the number of occurrences of character f in C, and let $n(C, \hat{f})$ denote the total number of occurrences in C of all characters that induce the same partition of $\mathcal{L}$ as f. For an edge e of T let $n_e(C) = \sum_{f \in c(e)} n(C, f) = n(C, \hat{f}_e)$, where f_e is any character that corresponds to edge e. Let

$$n_-(C, T) := \min_e \{n_e(C) : e \in \mathring{E}\},$$

$$n_+(C, T) := \max_f \{n(C, \hat{f}) : L(f, T) > 1\},$$

and

$$H(C, T) := \sum_f n(C, f) H(f, T),$$

where $H(f, T)$ is described by (2.1).

We pause to briefly provide some interpretation of these definitions. The quantity $n_-(C, T)$ is a measure of the minimum support for any edge of T by the characters in C, while $n_+(C, T)$ is the total number of characters that support some split of the species set $\mathcal{L}$ that does not correspond to any edge of T. The quantity $H(C, T)$ is sometimes called the *homoplasy* (or "number of extra steps") of C relative to T. Note that $H(C, T) \geq 0$.

The following result gives sufficient conditions for $\mathcal{MP}$ and $\mathcal{MC}$ to return a given tree from some sequence C of characters, regardless of how these characters arise.

LEMMA 1. *Let T be any fully resolved tree.*

(A) *$\mathcal{MP}$ selects tree T for a sequence C of r-state characters if*

$$n_-(C, T) > H(C, T).$$

(B) *$\mathcal{MC}$ selects tree T for a sequence C of 2-state characters if*

$$n_-(C, T) > n_+(C, T).$$

Proof. For brevity, we let $n(f) = n(C, f), n_- = n_-(C, T), n_+ = n_+(C, T)$.

Part (A). For a tree T_1, let $L(T_1) := \sum_f L(f, T_1)n(f)$. It suffices to show that $L(T_1)$ is strictly minimized when $T_1 = T$. First note that

$$L(T_1) = \Delta + H(C, T_1),$$

where $\Delta = \sum_f (|f(\mathcal{L})| - 1)n(f)$. Now, if $T_1 \neq T$ and since T is fully resolved, T has at least one interior edge e for which, for all $f \in c(e)$, we have $L(f, T_1) \geq 2$, and since $|f(\mathcal{L})| = 2$ for each $f \in c(e)$, this implies that $H(f, T_1) = L(f, T_1) - |f(\mathcal{L})| + 1 \geq 1$. Consequently, $H(C, T_1) \geq n_-$, and so, by the assumption in the lemma,

$$L(T_1) \geq \Delta + n_- > \Delta + H(C, T) = L(T),$$

which establishes the claim.

Part (B). Let $\nu(T_1)$ denote the number of occurrences in C of a character f with $L(f, T_1) = 1$. It suffices to show that $\nu(T_1)$ is strictly maximized when $T_1 = T$. First note that

$$\nu(T_1) = \sum_{\{f: L(f,T_1)=1\}} n(f) = \sum_{e \in E(T_1)} n_e(C).$$

Now, for any tree $T_1 \neq T$ let E' denote the subset of interior edges e of T for which $\pi_e \neq \pi_{e'}$ for any edge e' of T_1. Since T is fully resolved, $E' \neq \emptyset$, and the number of edges e in T_1 for which $\pi_e \cap \{\pi_{e'} : e' \in E(T)\} = \emptyset$ is at most $|E'|$ and for each such edge $n_e(C) \leq n_+$. Thus,

$$\nu(T) - \nu(T_1) \geq \sum_{e \in E'} n_e(C) - |E'|n_+ > 0$$

as required. □

When C is generated under the N_r model, we can apply this lemma to obtain sufficient conditions for the statistical consistency of $\mathcal{M}C$ and $\mathcal{M}P$ (Corollary 1). First we introduce the following terminology.

Definition 1. *Under the N_r model with parameters (T, p), let*

$$m_- := \min_f \{\mathbb{P}(f, T) : f \in c(T)\};\ m_+ := \max_f \{\mathbb{P}(f, T) : L(f, T) > 1\}$$

and

$$\overline{H} := \sum_f \mathbb{P}(f, T)H(f, T).$$

The quantity $\overline{H}$ is the expected homoplasy of the (random) character f on the underlying tree T.

Corollary 1.

(A) *$\mathcal{M}P$ is statistically consistent under the N_r model if*

$$m_- > \frac{\overline{H}}{r(r-1)}.$$

(B) *$\mathcal{M}C$ is statistically consistent under the N_2 model if $m_- > m_+$.*

Proof. Note that, under the N_r model, if f and f' induce the same partition of $\mathcal{L}$, then $\mathbb{P}(f,T) = \mathbb{P}(f',T)$. Suppose that we have a sequence C of c characters which evolve identically and independently under the N_r model. Then, by the weak law of large numbers, as c tends to infinity,

$$\frac{n_-(C,T)}{c} \to_p r(r-1)m_-,$$

where $\to_p$ denotes convergence in probability, since for each $f \in c(T)$ there are precisely $r(r-1)$ r-state characters that induce the same partition of $\mathcal{L}$ as f.

Also, we have

$$\frac{H(C,T)}{c} \to_p \overline{H},$$

while, for $r = 2$, we have

$$\frac{n_+(C,T)}{c} \to_p 2m_+.$$

The result now follows by Lemma 1. □

We can now state our main result.

THEOREM 1. *Under the N_r model with parameters (T,p) let*

$$p_{\text{sum}} := \sum_e p(e),$$

$$p_- := \min\{p(e) : e \in \mathring{E}\}; p_+ := \max\{p(e) : e \in E\},$$

and

$$p_\pm := p_- + p_+.$$

Then,

(A) *$\mathcal{M}P$ is statistically consistent if $p_{\text{sum}} < 1$ and*

$$p_- \geq \frac{p_{\text{sum}}^2}{1 - p_{\text{sum}}}; \tag{3.1}$$

(B) *when $r = 2$, $\mathcal{M}C$ is statistically consistent if*

$$p_- \geq p_+ p_\pm + 2p_\pm^2(1 + p_\pm), \tag{3.2}$$

and, in particular, this is satisfied whenever

$$p_- \geq 12p_+^2. \tag{3.3}$$

Remarks.

- Informally, the condition for the consistency of $\mathcal{M}P$ is that the mutation probability $p(e)$ associated to any interior edge should be at least (approximately) the square of the total expected number of mutations in the tree. Thus it assumes the $p(e)$ values are small and not too unequal.
  For $\mathcal{M}C$ the condition described states, informally, that the smallest mutation probability on any interior edge is of the order of the square of the largest

mutation probability. In particular, $\mathcal{MC}$ is statistically consistent under the N_2 model whenever $p(e)$ is a constant, p, across the edges of the tree and p takes a value at most $\frac{-5+\sqrt{41}}{16} \approx 0.087$, since in that case $p \geq p(2p) + 2(2p)^2(1+2p)$ and so (noting that $p_- = p_+ = p$ and $p_\pm = 2p$) we see that inequality (3.2) is satisfied. More generally for any bound on the ratio of the $p(e)$ values, Theorem 1(B) implies that there exists an upper bound on p_+ (dependent on that bound) for which $\mathcal{MC}$ is statistically consistent under the N_2 model.

- Note also that the size r of the state space does not enter into inequality (3.2). In fact it can be shown that, for any fixed values of $p_- > 0$ and $p_+ < 1$, if r is sufficiently large, then $\mathcal{MP}$ will be consistent (this is a special case of Theorem 3 of [12]).

Proof of Theorem 1 (A). Let

$$\overline{L} := \sum_f \mathbb{P}(f,T)L(f,T).$$

Letting $\mathbb{E}$ denote expectation, we have $\overline{L} = \mathbb{E}[L(G|\mathcal{L},T)]$ for an extension G randomly generated on T under the N_r model. Now, $L(G|\mathcal{L},T) \leq \text{ ch}(G,T)$, and so,

$$\overline{L} \leq \mathbb{E}[\text{ch}(G,T)]. \tag{3.4}$$

However, $\text{ch}(G,T)$ is simply a sum of independent 0/1 random variables as follows: to each edge $e = \{u,v\}$ independently assign the value 1 if and only if $G(u) \neq G(v)$ (which has probability $p(e)$), and assign 0 otherwise. Consequently,

$$\mathbb{E}[\text{ch}(G,T)] = p_{\text{sum}}. \tag{3.5}$$

Let

$$Q := \prod_e (1-p(e)),$$

which is the probability that that there is no mutation on any edge e of T (i.e., for each edge $e = \{u,v\}$ we have $G(u) = G(v)$). Now,

$$\overline{H} = \overline{L} - \sum_f \mathbb{P}(f,T)(|f(\mathcal{L})| - 1), \tag{3.6}$$

and

$$\sum_f \mathbb{P}(f,T)(|f(\mathcal{L})| - 1) \geq \mathbb{P}(L(G|\mathcal{L},T) = 1) \geq \mathbb{P}(\text{ch}(G,T) = 1). \tag{3.7}$$

Furthermore, $\mathbb{P}(\text{ch}(G,T) = 1) = \sum_e p(e) \prod_{e' \neq e}(1-p(e')) \geq p_{\text{sum}}Q$. Thus, by (3.6) and inequality (3.7) we have $\overline{H} \leq \overline{L} - p_{\text{sum}}Q$, while inequality (3.4) and (3.5) give $\overline{L} \leq p_{\text{sum}}$, and hence

$$\overline{H} \leq p_{\text{sum}}(1-Q). \tag{3.8}$$

Now, for a character $f \in c(T)$, let e_f denote the edge of T to which f corresponds. An extension g of f to V can be obtained by assigning a leaf v_0 the value $f(v_0)$ (with

probability $\frac{1}{r}$), assigning (appropriate) different states to the ends of e_f, and for each edge $e \neq e_f$ assigning the same state to each end of e. Consequently,

$$\mathbb{P}(f,T) \geq \mathbb{P}(G=g) = \frac{1}{r} \times \frac{p(e_0)}{r-1} \prod_{e \neq e_f} (1-p(e)) > \frac{1}{r(r-1)} Q p_-.$$

Thus,

$$m_- \geq \frac{1}{r(r-1)} p_-. \tag{3.9}$$

Now, our hypothesis is that $p_- \geq \frac{p_{\text{sum}}^2}{1-p_{\text{sum}}}$. Then, $1 - p_{\text{sum}} \geq \frac{p_{\text{sum}}}{p_{\text{sum}}+p_-}$ which together with the purely algebraic inequality

$$Q > 1 - p_{\text{sum}}$$

implies that $Q > \frac{p_{\text{sum}}}{p_{\text{sum}}+p_-}$. Rearranging gives $Qp_- > p_{\text{sum}}(1-Q)$ and thus, in view of the inequalities (3.8), (3.9) we have

$$m_- > \frac{\overline{H}}{r(r-1)Q} > \frac{\overline{H}}{r(r-1)}.$$

Part (A) of the theorem now follows from Corollary 1(A).

Part (B). Throughout this proof we will make extensive use of the following two properties of the N_r model with underlying tree T:

- The conditional probability of generating a character f given that a leaf $l \in \mathcal{L}$ of T is in a particular state $\mu \in R$ is precisely $r\mathbb{P}(f,T)$ if $f(l) = \mu$ (and is 0 otherwise).
- Let t_1 and t_2 be two subtrees of T that share one nonleaf vertex, v. Let f_1 and f_2 denote the restrictions of f to the leaves of t_1 and t_2, respectively. Then f_1 and f_2 are conditionally independent once the state of vertex v is specified.

Throughout the rest of this proof we will take (without loss of generality) $R = \{0,1\}$. Let P_0 denote the probability of generating the character that maps all leaves to state 0. We first establish the following inequality. Suppose that $f \in c(T)$ corresponds to edge $e \in \mathring{E}$. Then,

$$\mathbb{P}(f,T) > \frac{p(e)}{1-p(e)} P_0. \tag{3.10}$$

To establish (3.10) let T_1, T_2 denote the two rooted subtrees of T whose roots are the ends of edge e. Without loss of generality we may suppose all the leaves of T_1 (resp., T_2) are mapped by f to 0 (resp., 1). For $i = 1, 2$, and $\mu, \nu \in \{0,1\}$, let $P_i(\mu,\nu)$ denote the conditional probability, under the N_2 model (restricted to T_i) that all the leaves in T_i are in state μ given that the root vertex (which we take as our v_0) is in state ν. Let $\alpha = \frac{1}{2}(P_1(0,1)P_2(1,0) + P_1(0,0)P_2(1,1)); \beta = \frac{1}{2}(P_1(0,0)P_2(1,0) + P_1(0,1)P_2(1,1))$. Then,

$$\mathbb{P}(f,T) = \alpha p(e) + \beta(1-p(e)), \tag{3.11}$$

and by virtue of the symmetry in the N_2 model which implies that

$$P_i(0,0) = P_i(1,1) \text{ and } P_i(0,1) = P_i(1,0),$$

we see that

$$P_0 = \alpha(1 - p(e)) + \beta p(e).$$

Straightforward algebraic manipulation then shows that (since $p(e) < \frac{1}{2}$), $\frac{\mathbb{P}(f,T)}{P_0} > \frac{p(e)}{1-p(e)}$, as required to establish (3.10). Actually all we shall require is the following corollary of inequality (3.10), namely for any $f \in c(T)$,

$$\mathbb{P}(f, T) > P_0 p_-. \tag{3.12}$$

Most of the remainder of the proof is devoted to establishing the following upper bounds on $\mathbb{P}(f, T)$.

Claim.

- *If $L(f, T) = 1$, then*

$$\mathbb{P}(f, T) < p_\pm P_0, \tag{3.13}$$

- *while, if $L(f, T) > 1$, then*

$$\mathbb{P}(f, T) < 2p_\pm^2(1 + p_\pm)P_0 < p_\pm P_0. \tag{3.14}$$

(Note that inequality (3.13) is required purely to justify the proof of inequality (3.14).) The proof of inequality (3.13) is by induction on the number n of leaves of T. The inequality holds for $n = 2$ since then there is just one edge e and $p(e) = p_- = p_+; P_0 = \frac{1}{2}(1 - p(e))$ and so, since $p(e) \in (0, 0.5)$, $\mathbb{P}(f, T) = \frac{1}{2}p(e) < p(e)(1 - p(e)) = p_\pm P_0$, as required. Now, suppose that $n > 2$. We distinguish two subcases.

- $f \in c(T)$,
- f corresponds to a noninterior edge of T.

In the first subcase let T_1, T_2 be as described above. Consider the two subtrees $\{t_i^a, t_i^b\}$ of T_i that intersect precisely on the vertex v_i, where $e = \{v_1, v_2\}$ is the edge associated with f (see Figure 2(a)). For $\theta = a, b$ let $P_i^\theta(\mu, \nu)$ denote the conditional probability, under the N_2 model restricted to t_i^θ, that the leaves t_i^θ are all in state μ given that v_i is in state ν. Then

$$P_i(\mu, \nu) = P_i^a(\mu, \nu)P_i^b(\mu, \nu). \tag{3.15}$$

Now, if $\mu \neq \nu$, then the restriction of f to the leaves of each subtree has L value of 1 on each subtree, so by the inductive hypothesis,

$$P_i^\theta(\mu, \nu) < p_\pm P_i^\theta(0, 0), (\mu \neq \nu). \tag{3.16}$$

Now, recalling (3.11) we have $\mathbb{P}(f, T) = \alpha p(e) + \beta(1 - p(e))$ and so, substituting (3.15) and inequality (3.16) into the definitions of α and β, we deduce that

$$\mathbb{P}(f, T) < K[p(e) + 2p_\pm^2(1 - p(e)) + p(e)p_\pm^4], \tag{3.17}$$

where $K = \frac{1}{2}P_1^a(0, 0)P_1^b(0, 0)P_2^a(0, 0)P_2^b(0, 0)$.

Also, we have

$$P_0 \geq (1 - p(e))K \geq (1 - p_+)K. \tag{3.18}$$

Now, we can bound the term in brackets in (3.17) by noting that $p(e) + 2p_\pm^2(1 - p(e)) + p(e)p_\pm^4 < p_+ + 2p_\pm^2 + p_\pm^3$ (since $p_\pm < 1$), and then, by our assumption (3.2),

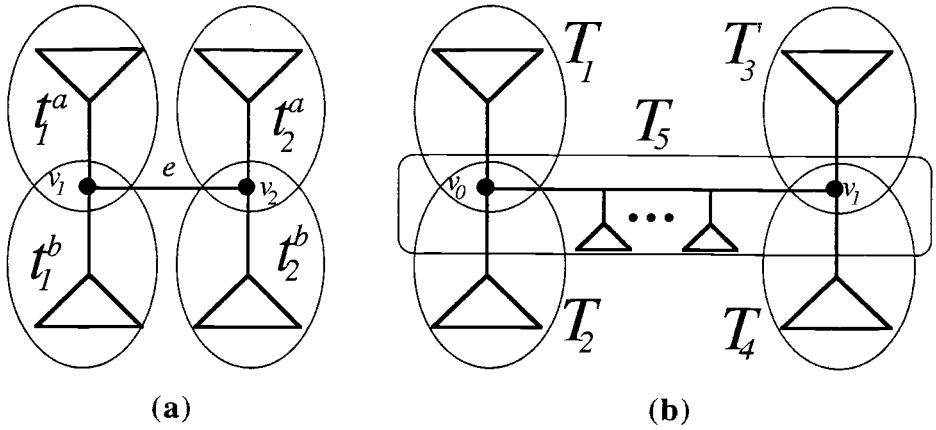


FIG. 2. *Representations of T for the proof of the upper bounds* (3.13) *and* (3.14).

we have $p_+ + 2p_\pm^2 + p_\pm^3 \leq p_\pm(1-p_+)$. Substituting this into (3.17) and comparing the result to (3.18) establishes inequality (3.13) in the first subcase.

For the second subcase, we may assume that f maps some leaf, incident with an edge e, to state 0, and all other leaves of T to state 1.

Let t^a, t^b denote the other two subtrees of T which intersect precisely on the vertex at the other end of edge e from the leaf. Then, defining $P^a(\mu,\nu), P^b(\mu,\nu)$ analogously as before, we have

$$\mathbb{P}(f,T) = \frac{1}{2}(P^a(1,1)P^b(1,1)p(e) + P^a(1,0)P^b(1,0)(1-p(e))),$$

and so

$$\mathbb{P}(f,T) < \frac{1}{2}P^a(0,0)P^b(0,0)[p(e) + p_\pm^2(1-p(e))].$$

Consequently, since $p(e) + p_\pm^2(1-p(e)) < p_+ + p_\pm^2 < p_\pm(1-p_+)$, we have

$$\mathbb{P}(f,T) < \frac{1}{2}P^a(0,0)P^b(0,0)p_\pm(1-p_+).$$

Furthermore, since $P_0 \geq \frac{1}{2}P^a(0,0)P^b(0,0)(1-p(e)) \geq \frac{1}{2}P^a(0,0)P^b(0,0)(1-p_+)$, we deduce that inequality (3.13) holds in this subcase also.

We now establish inequality (3.14). We first observe that our condition (3.2) forces $2p_\pm^2(1+p_\pm) < p_\pm$ so it is only the first inequality in (3.14) we need to establish.

The proof again is by induction on n. For $n = 2, 3$ there is nothing to prove. Suppose $n \geq 4$ and $L(f,T) > 1$. Then, a standard application of Menger's theorem from graph theory shows that there are two edge-disjoint paths in T, each of which connects leaves assigned different states by f [14, Lemma 1]. Thus, we may represent T as in Figure 2(b), with five subtrees trees $T_1, \ldots, T_5$ as shown, each pair of which is disjoint or overlaps at one of two (generally nonadjacent) vertices v_0 and v_1 as shown. We call these two vertices *reference vertices*, and note that each of $T_1, \ldots, T_4$

has exactly one reference vertex (and it is a leaf of that subtree) while T_5 has both reference vertices as leaves.

For $i = 1, \ldots, 4$ and $\mu \in \{0, 1\}$ let f^i_μ be the character defined on the leaf set of T_i which maps its reference vertex to μ and all other leaves of T_i to the element that f specifies. Let $f_{\mu,\nu}$ be the character defined on the leaf set of T_5 which maps v_0 to μ, v_1 to ν, and every other leaf in T_5 to the element that f specifies. For $i = 1, \ldots, 4$ let

$$P_i(\mu) := 2\mathbb{P}(f^i_\mu, T_i); L_i(\mu) := L(f^i_\mu, T_i)$$

and

$$P_5(\mu, \nu) := 2\mathbb{P}(f_{\mu,\nu}, T_5); L_5(\mu, \nu) := L(f_{\mu,\nu}, T_5).$$

For $i = 1, \ldots, 5$ let P^i_0 equal twice the probability of generating on T_i the character which maps all leaves to 0. Then,

$$\mathbb{P}(f, T) = \frac{1}{2} \sum_{\mu,\nu} P_5(\mu, \nu) \prod_{i=1,2} P_i(\mu) \prod_{i=3,4} P_i(\nu). \tag{3.19}$$

Now, by induction, we may assume that inequality (3.14) holds for all five subtrees, and invoking inequality (3.13) if necessary we deduce that, for $i = 1, \ldots, 4$, $P_i(\mu) < p_\pm P^i_0$ when $L_i(\mu) > 0$ (while $P_i(\mu) = P^i_0$ otherwise). Similarly, $P_5(\mu, \nu) < p_\pm P^5_0$ when $L_5(\mu, \nu) > 0$, and $P_i(\mu) = P^i_5$ otherwise. Consequently for when $\mu = \nu \in \{0, 1\}$ we introduce at least two powers of $p_\pm$ into the product terms of (3.19), while for $\mu \neq \nu$ we introduce at least three powers of $p_\pm$. Thus, we deduce that

$$\mathbb{P}(f, T) < 2p^2_\pm(1 + p_\pm) \cdot \frac{1}{2} \cdot \prod_{i=1,\ldots,5} P^i_0$$

and inequality (3.14) now follows by observing that

$$P_0 > \frac{1}{2} \cdot \prod_{i=1,\ldots,5} P^i_0.$$

Finally we establish part (B) of the theorem. In view of inequalities (3.12) and (3.14) we have

$$m_- > P_0 p_-$$

and

$$m_+ < 2p^2_\pm(1 + p_\pm)P_0.$$

Now, by our condition (3.2), $p_- > p_+p_\pm + 2p^2_\pm(1 + p_\pm) > 2p^2_\pm(1 + p_\pm)$, we see that $m_- > m_+$. The first claim in part (B) of the theorem now follows from Corollary 1(B).

Finally we show that inequality (3.3) implies inequality (3.2). Suppose that $p_- \geq 12p^2_+$. Then, $p_+ \leq \frac{1}{12}$, and so, we have $p_- \geq 12p^2_+ \geq 2p^2_+ + 8p^2_+(1 + 2p_+) \geq p_+p_\pm + 2p^2_\pm(1 + p_\pm)$ (since $p_+ \geq \frac{1}{2}p_\pm$), as required. □

4. Remarks. An interesting theoretical question is whether $\mathcal{MP}$ is statistically consistent under the N_r model when $p(e) = p$ (for all edges e) and p is less than some value $p^{(r)} > 0$ that is independent of n. This question is open even for the case $r = 2$ (however, from [11], if such a positive value of $p^{(2)}$ exists, it must be less than $\frac{1}{8}$). Note that the sufficient condition (3.2) described in Theorem 1(B) requires that the $p(e)$ values to converge to 0 at least as fast as n^{-2}, where $n = |\mathcal{L}|$, so in a certain sense the sufficient condition described for $\mathcal{MC}$ is much stronger than that for $\mathcal{MP}$ (in the case $r = 2$).

It is also instructive to compare the strengths of the two parts of Theorem 1 for the case when $n = 4$. In [4] Felsenstein considered the N_2 model on a resolved tree on four leaves, with two nonadjacent edges having $p(e) = s$, and the remaining three edges having $p(e) = q$. He showed that $\mathcal{MP}$ is statistically inconsistent precisely when $q(1-q) < s^2$, which, for q small, amounts, approximately, to $q < s^2$. By contrast, the sufficient condition described in the above theorem for $\mathcal{MP}$ would require $q \geq \frac{(2s+3q)^2}{1-2s-3q}$ which for $q \ll s \ll 1$ amounts, approximately, to $q > 4s^2$. For $\mathcal{MC}$ (which agrees with $\mathcal{MP}$ on trees with four leaves) the analogous sufficient condition described by inequality (3.2) reduces, approximately, to $q > 3s^2$. In either case we see a gap between sufficiency and necessity conditions for statistical consistency. In fact, for the case of four leaves it is possible to characterize precisely the conditions on the five $p(e)$ values for the statistical consistency of $\mathcal{MP}$ (see [10]), however, in general, this appears to be difficult. Thus a challenge for the future would be to narrow the gap between necessary and sufficient conditions for the statistical consistency of $\mathcal{MP}$ and $\mathcal{MC}$. An extension of Theorem 1(B) to $r > 2$ would also be interesting.

Acknowledgments. The author thanks the Isaac Newton Institute (Cambridge) for its hospitality, Jotun Hein for offering helpful comments on an earlier version of this manuscript, and the anonymous referee for several suggestions for improving the presentation of this paper.

REFERENCES

[1] J.A. Cavender, *Taxonomy with confidence*, Math. Biosci., 40 (1978), pp. 271–280.
[2] J.T. Chang, *Full reconstruction of Markov models on evolutionary trees: Identifiability and consistency*, Math. Biosci., 134 (1996), pp. 189–215.
[3] J.S. Farris, *A probability model for inferring evolutionary trees*, Syst. Zool., 22 (1973), pp. 250–256.
[4] J. Felsenstein, *Cases in which parsimony or compatibility will be positively misleading*, Syst. Zool., 27 (1978), pp. 401–410.
[5] L.R. Foulds and R.L. Graham, *The Steiner problem in phylogeny is NP-complete*, Adv. in Appl. Math., 3 (1982), pp. 43–49.
[6] M.D. Hendy and D. Penny, *A framework for the quantitative study of evolutionary trees*, Syst. Biol., 38 (1986), pp. 297–309.
[7] J.T. Jukes and C.R. Cantor, *Evolution of protein molecules*, in Mammalian Protein Metabolism, H.N. Munro ed., Academic Press, New York, 1996, pp. 21–132.
[8] J. Kim, *General inconsistency conditions for maximum parsimony: Effects of branch length and increasing the number of taxa*, Syst. Biol., 45 (1996), pp. 363–374.
[9] J. Neyman, *Molecular studies of evolution: A source of novel statistical problems*, in Statistical Decision Theory and Related Topics, S.S. Gupta and J. Yackel, eds., Academic Press, New York, 1971, pp. 1–27.
[10] D. Penny, M.D. Hendy, and M.A. Steel, *Testing the theory of descent*, in Phylogenetic Analysis of DNA sequences, M.M. Miyamoto and J. Cracraft, eds., Oxford University Press, Oxford, UK, 1991, pp. 155–183.
[11] M.A. Steel, *Distributions on Bicoloured Evolutionary Trees*, PhD thesis, Massey University, Palmerston North, New Zealand, 1989.

[12] M.A. Steel and D. Penny, *Parsimony, likelihood, and the role of models in molecular phylogenetics*, Mol. Biol. Evol., 17 (2000), pp. 839–850.
[13] D.L. Swofford, G.J. Olsen, P.J. Waddell, and D.M. Hillis, *Phylogenetic inference*, in Molecular Systematics, 2nd ed., D.M. Hillis, C. Moritz, and B.K. Marble, eds., Sinauer Associates, Sunderland, MA, 1996, pp. 407–514.
[14] C. Tuffley and M. Steel, *Links between maximum likelihood and maximum parsimony under a simple model of site substitution*, Bull. Math. Biol., 59 (1997), pp. 581–607.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 49–66

COMPACT REPRESENTATIONS OF CUTS*

DAVID HARTVIGSEN†

Abstract. Consider the $\binom{n}{2}$ (or $O(n^2)$) min-cut problems on a graph with n nodes and nonnegative edge weights. Gomory and Hu [J. Soc. Indust. Appl. Math., 9 (1961), pp. 551–570] showed (essentially) that there are at most $n-1$ different min-cuts. They also described a compact structure (the flow equivalent tree) of size $O(n)$ with the following property: for any pair of nodes, the value of a min-cut can be obtained from this structure. Furthermore, they showed how this structure can be found by solving only $n-1$ min-cut problems. This paper contains generalizations of these results. For example, consider a *k-terminal cut problem* on a graph: for a given set of k nodes, delete a minimum weight set of edges (called a *k-cut*) so that each of the k nodes is in a different component. There are $\binom{n}{k}$ (or $O(n^k)$) such problems. Hassin [Math. Oper. Res., 13 (1988), pp. 535–542] showed (essentially) that there are at most $\binom{n-1}{k-1}$ (or $O(n^{k-1})$) different min k-cuts. We describe a compact structure of size $O(n^{k-1})$ with the following property: for any k nodes, the value of a min k-cut can be obtained from this structure. We also show how this structure can be found by solving only $\binom{n-1}{k-1}$ k-terminal cut problems. This work builds upon the results of Hassin [Math. Oper. Res., 13 (1988), pp. 535–542], [Oper. Res. Lett., 9 (1990), pp. 315–318], and [Lecture Notes in Comput. Sci. 450, 1990, pp. 228–299].

Key words. min-cut, flow tree, k-cut

AMS subject classifications. 05, 68, 90

PII. S0895480196312334

1. Introduction.

1.1. Overview of results. We consider two generalized cut problems in this paper. A notion common to both problems is the *k-cut*, which is a partition of a given set V into k nonempty sets. Each k-cut is given a real *weight*.

The first problem we consider is the *k-terminal cut problem*. The objective of a k-terminal cut problem is to find a minimum weight k-cut that separates k specified elements of V (i.e., such that each of the k elements is in a different set of the partition). The second problem is the *k-pair cut problem*. The objective of this problem is to find a 2-cut that simultaneously separates each of k specified pairs of elements of V.

These two cut problems generalized problems typically considered on graphs. For example, a 2-cut problem and a 1-pair cut problem, where the weights derive from the edges in a graph with node set V, are classical *min-cut problems*. (See section 1.2 for a more detailed discussion of related problems on graphs.) These generalizations lead to new and simple proof techniques and algorithms for several classical problems on graphs.

Observe that for a given set V there are a number of different problems of each type. For example, if we let $n = |V|$, then there are $\binom{n}{2}$ or $O(n^2)$ different 2-terminal cut problems. Regarding when the weights on the 2-cuts derive from nonnegative weights on the edges of a graph on V, Gomory and Hu [10] showed the following, somewhat surprising, results.

*Received by the editors November 22, 1996; accepted for publication (in revised form) October 17, 2000; published electronically January 5, 2001.
http://www.siam.org/journals/sidma/14-1/31233.html
†College of Business Administration, University of Notre Dame, P.O. Box 399, Notre Dame, IN 46556-0399 (David.Hartvigsen.1@nd.edu).

1. There exists a set of at most $n-1$ 2-cuts that contains a min weight solution for each 2-terminal cut problem (and this bound is the best possible).
2. There exists a structure (the *flow equivalent tree*) of size $O(n)$ from which the solution *value* of any 2-terminal cut problem can be found (in polynomial time).
3. There exists an algorithm that simultaneously constructs the set in 1 and the structure in 2; it requires solving only $n-1$ 2-terminal cut problems.

Result 1 shows there are significantly fewer min weight solutions than problems. This is described by a *best possible upper bound* ($n-1$ in this case). Result 2 shows that there exists a structure whose size is the same order as the best possible upper bound and that, nevertheless, contains all the information necessary to obtain the value of a min weight solution to any problem in polynomial time. We refer to such structures as *compact representations* of the cuts. Result 3 shows that it is sufficient to solve a number of problems equal to the best possible upper bound in order to find all the min weight 2-cut solutions and to construct the compact representation.

Note that result 2 assumes that cut values can be stored in space bounded by a constant. We make a similar assumption in general: values of k-cuts can be stored in space bounded by a constant.

Gomory and Hu [10] proved stronger results than results 2 and 3 above: There exists a special flow equivalent tree, the *cut tree*, from which the actual min 2-cut for each 2-terminal cut problem can be found in polynomial time; furthermore, this tree can be found by solving only $n-1$ 2-terminal cut problems.

In this paper we present generalizations of results 2 and 3 for the k-terminal cut and k-pair cut problems. Let us describe our results in the context of the work that has already been done.

Hassin [19] proved the best possible upper bounds for the k-terminal cut problems and the k-pair cut problems. In each case the bounds show there are significantly fewer min weight solutions than problems. For example, he showed there exists a set of at most $\binom{n-1}{k-1}$(or $O(n^{k-1})$) k-cuts that contains a min weight solution for every k-terminal cut problem (of which there are $O(n^k)$).

Hassin [19] described a compact representation (of size $O(n)$) for the (general) 2-terminal cut problem and hence for the equivalent 1-pair cut problem. Cheng and Hu [3] described a different compact representation for this problem. Both representations derive from special tree structures in a graph.

We present in this paper compact representations for all values of k for both cut problems. For example, for the k-terminal cut problem we present a structure of size $O(n^{k-1})$, for fixed k, from which the value of a min weight solution to any of the $O(n^k)$ problems can be found. Furthermore, any such value can be found from this structure in polynomial time, for fixed k. We observe that for the 3-terminal cut and the 2-pair cut problems, the compact representations are closely related to the well-known notion of cycle bases in graphs.

Hassin [19] provided an algorithm for the (general) 2-terminal cut problem that finds all the min 2-cuts and a compact representation (as in result 3). However, it requires solving $O(n \log n)$ 2-terminal cut problems, which can be more than the best possible upper bound of $n-1$. Cheng and Hu [3] found a different algorithm for this problem that finds all the min 2-cuts and a different compact representation. It requires solving only $n-1$ 2-terminal cut problems. Hassin [16] provided a general algorithm of the type described in result 3 that can be applied to both of the more general cut problems studied in this paper. However, the algorithm requires solving

twice the number of problems in the best possible upper bound. For example, for the k-terminal cut problem Hassin's algorithm requires solving $2\binom{n-1}{k-1}$ k-terminal cut problems to find the set of at most $\binom{n-1}{k-1}$ min k-cuts. Also, the complexity of Hassin's algorithm, not including the work of solving the cut problems, is exponential (because the representation that it utilizes has exponential size).

We present a variant of Hassin's general algorithm in this paper. Our algorithm has the following properties: It finds a compact representation as well as all the min cuts for the k-terminal cut and k-pair cut problems (as well as a more general class of problems), it requires the solution of at most the best upper bound number of cut problems (hence the factor of 2 goes away), and the complexity is polynomial for both types of problems, for fixed k. For the special case of 2-cuts, the algorithm we present is considerably simpler both to describe and show valid than the 2-cut algorithm of Cheng and Hu. For the special case of 2-cuts in graphs, our algorithm is different from the algorithm of Gomory and Hu and appears to be new. Our algorithm is quite simple and requires no "shrinking."

The remainder of the paper is organized as follows. We end this section with a quick overview of the history of the cut problems we study. In section 2 we present our results for the k-terminal cut problem. In section 3 we review some useful techniques of Hassin, which we apply in our proofs. In section 4 we prove our results for the k-terminal cut problem. In section 5 we present and prove our results for the k-pair cut problem. In section 6 we consider the algorithmic question of how to efficiently find the compact representations for the two types of problems discussed above. Section 7 contains some open problems.

1.2. History of the k-terminal and k-pair cut problems. In the literature, a graphical k-terminal cut problem (where the cut weights derive from nonnegative edge weights) is sometimes called a *multiterminal cut problem.* Dahlhaus et al. [7] have proved the following. The graphical k-terminal cut problem is NP-hard for $k \geq 3$, even if k is fixed and all the edge weights are equal to 1 (see also [8]); if only planar graphs are considered, the problem is still NP-hard; however, if k is fixed and only planar graphs are considered, then there exists a polynomial time algorithm. The history of this problem and several applications are also briefly discussed in [7]. Additional work on the graphical problem using the techniques of polyhedral combinatorics appears in [4], [6], and [8]. Heuristics for this problem have been studied in [7] and [22]. Numerous applications of the 2-terminal cut problem appear in [2] and [27].

An algorithm for finding all the minimum weight 2-cuts, as in result 3, can be used to find the overall minimum weight 2-cut in a graph. This problem has found application in algorithms for solving the traveling salesman problem (see [5], [25], and [26]) and network reliability problems (see [13]). Padberg and Rinaldi [26] presented a variant of the algorithm of Gomory and Hu that can be implemented more easily and empirically runs faster. Gusfield [12] showed how the algorithm of Gomory and Hu can be significantly simplified by eliminating the need for "shrinking" in the construction of the flow equivalent tree. (The algorithm presented in this paper, when it is specialized to the graphical 2-terminal cut case, appears to be different from Gusfield's algorithm.)

We note that, surprisingly, there is an algorithm presented in [9] that finds the overall min k-cut in a graph in polynomial time for fixed k, starting from scratch. Related work appears in [14], [20], and [21]. Hao and Orlin [13] presented an algorithm for finding the overall min 2-cut that effectively requires the solution of only one max flow problem. Related work appears in [24] and [29].

Let us end this review of the k-terminal cut problem by mentioning some other related results. Granot and Hassin [11] proved variants of results 1–3 for the case of graphs with both edge and node weights. Hassin [17] proved variants of results 1–3 for the related "xcut problem." Hartvigsen [15] showed how the 2-terminal cut problem and the flow equivalent tree can be generalized to matroids. Finally, Trotter [30] showed how a different aspect of Gomory and Hu's work can be generalized to matroids and in so doing provided a generalization of the notion of cut tree.

In the literature, the k-pair cut problem (in graphs) is sometimes called the *multicommodity cut problem*. It is closely related to the *multicommodity flow problem*, which is commonly used to model communications, logistics, manufacturing, and transportation systems (see [2] for numerous examples and solution techniques). The relationship between these problems (in particular, when they have the same value for a given set of pairs of nodes) has been widely studied. For an excellent survey of these results, see [28]. A key result of this type is the 2-commodity flow theorem of Hu [23]. Note that a k-pair cut problem may have no feasible solution (since we only consider solutions that are 2-cuts; see section 5.1). A different problem arises, for example, if we consider solutions that are j-cuts for $j \leq k$.

2. The k-terminal cut problem and our results. Recall that a single *k-cut* is a partition of a set V into k nonempty subsets and that we give each such k-cut a real *weight*. A single *k-terminal cut problem* is the following: given $V' \subseteq V$, with $|V'| = k$, find a minimum weight k-cut of V such that each element of V' is in a different set of the partition. Let us emphasize that even for the case $k = 2$, this is a generalization of the well-known *min cut problem*, where V denotes the node set of a graph and the weights of the 2-cuts derive from nonnegative weights on the edges of the graph on V.

Observe that for a given set V, with $|V| = n$, there are $\binom{n}{k}$ different k-terminal cut problems. Hassin showed, however, that there are significantly fewer min weight solutions to these problems. Next we state his result. Note that $\binom{n}{k}$ is $O(n^k)$.

THEOREM 2.1 (see [19]). *There exists a set of at most $\binom{n-1}{k-1}$ (or $O(n^{k-1})$) k-cuts that contains a min weight solution for every k-terminal cut problem. (Furthermore, this result is the "best possible.")*

As noted in the introduction, Gomory and Hu [10] first proved the special case of Theorem 2.1 when $k = 2$ and the weights on the 2-cuts derive from nonnegative weights on the edges of a graph with node set V.

Hassin [19] proved another interesting result for the (general) case $k = 2$. The idea is that the solution *value* to any of the original $O(n^2)$ min-cut problems can be found by referring only to an object of size $O(n)$. Note that such a result is a bit surprising since it requires $O(n)$ space to write down any one 2-cut (e.g., by listing the corresponding node partition), and there can be $n - 1$ different min 2-cuts. After a quick definition, we can precisely state his result.

Let $H = (V, E)$ be a complete graph on V. Set the weight of each edge $uv \in E$ to be the weight of a min 2-cut for u and v (in the original problem). Let us call any maximum weight spanning tree of H a *compact representation tree* (or a *CR tree*). Observe that this tree requires $O(n)$ space. Hassin's result is the following.

THEOREM 2.2 (see [19]). *The value of a min 2-cut for any pair $u, v \in V$ can be obtained from a CR tree T as follows: add the edge uv to T and find the unique cycle that contains it; the minimum weight of an edge of T in this cycle is the value of a min 2-cut for u and v.*

As noted in the introduction, Gomory and Hu [10] proved the special case of this

theorem when the weights on the 2-cuts derive from nonnegative weights on the edges of a graph with node set V. They referred to the CR tree as a *flow equivalent tree*.

One objective of this paper is to generalize, to all k, Hassin's notion of a CR tree. That is, we want to create a compact structure of size $O(n^{k-1})$ from which we can obtain the solution value to any k-terminal cut problem. The trick is to use matrices instead of graphs, as follows.

Consider the k-terminal cut problems on a set V. Let M be a $\left(k \times \binom{n}{k-1}\right)$ matrix whose rows are indexed on the subsets of V of size k and whose columns are indexed on the subsets of V of size $k-1$. Let $M_{ij} = 1$ if j is a subset of i, and 0 otherwise. Thus each row of M corresponds to a k-terminal cut problem and each row contains k 1's. A *base* of M is a maximal independent (over GF(2)) set of rows of M.

Let the *weight* of each row of M be the solution value of the corresponding k-terminal cut problem, and let the *weight* of a base be the sum of the weights of the rows in the base. A matrix consisting of the rows in a maximum weight base of M is called a *compact representation matrix* (or a *CR matrix*) for the solutions to all the k-terminal cut problems.

To see how the notion of a CR matrix generalizes the notion of a CR tree, consider the following two examples.

Example 1. When $k = 2$, the matrix M is the edge-node incidence matrix for a complete graph constructed on V. It is well known that a set of rows of such a matrix is independent if and only if the corresponding edges in the complete graph are acyclic. Hence a CR matrix corresponds to Hassin's CR tree. This CR matrix has $n-1$ rows each with only two nonzero entries.

Example 2. When $k = 3$, the matrix M is the triangle-edge incidence matrix for a complete graph constructed on V. In this case the rows are incidence vectors for cycles in the complete graph and hence are vectors in the well-known cycle space of this graph. A CR matrix corresponds to a maximum weight base for the cycle space that consists of triangles (although the weights on the triangles do not derive from weights on the edges of these triangles). This CR matrix has $\binom{n-1}{2}$ (this is the dimension of the cycle space for a complete graph) rows each with only three nonzero entries.

Next we state our main results for k-cuts. The first result is that the solution value to any k-terminal cut problem can be found from the CR matrix. The second result is that the CR matrix is compact in the sense that it requires the same amount of space as the maximum number of different k-cut solutions, for fixed k. We prove these results in section 4.

THEOREM 2.3. *The value of a min k-cut for any set $V' \subseteq V$, where $|V'| = k$, can be obtained from a CR matrix as follows: add the incidence vector for V' (as it occurs in M) to the CR matrix and find the unique circuit (minimal dependent set of rows over $GF(2)$) that contains it; the minimum weight of a row of the CR matrix in this circuit is the value of a min k-cut for V'.*

THEOREM 2.4. *A CR matrix for the k-terminal cut problems on V can be stored in $O(n^{k-1})$ space, for fixed k.*

Theorem 2.1 is a special case of Theorem 2.3 by Example 1. Observe that the circuit in Theorem 2.3 can be found with Gaussian elimination. Also observe that, for fixed k, a CR matrix has polynomial size. Thus, for fixed k, the work to find a min k-cut value from a CR matrix is polynomial.

3. General results. We begin this section by reviewing some related work of Hassin that we use later. We finish this section by proving a simple, but useful, result

in linear algebra. All algebra done in this paper is over $GF(2)$.

Hassin [19] considered the following general setting. A finite set of *problems* is given together with a finite set of *solutions*. Each solution is given a distinct real *weight*. A matrix A is also given where there is a row of A corresponding to each problem, a column of A corresponding to each solution, entry $a_{ij} = 1$ if solution j is *feasible* for problem i, and $a_{ij} = 0$ otherwise. The *weight* of a problem equals the minimum weight of a solution that is feasible for that problem. The following is one of Hassin's key results.

THEOREM 3.1 (see Corollary 2.2 in [19]). *There exists a set of at most* $\text{rank}(A)$ *solutions that contains the minimum weight solution for every problem.*

Hassin used this result to prove Theorem 2.1 of this paper and a version of Theorem 2.1 for the k-pair cut problem (see Theorem 5.2 in this paper). That is, he calculated the ranks of the matrices A for these two problems and constructed examples to show that these bounds are the best possible.

Hassin also introduced the notion of a *maximum solution base for* A, which is defined to be a maximum weight base of the rows of A (where each row has the same weight as the associated problem). He showed the following.

THEOREM 3.2 (see Theorem 2.3 in [19]). *The value of a minimum weight solution to any problem can be obtained from a maximum solution base for A as follows: add the row from A for this problem to the maximum solution base for A and find the unique circuit (minimal dependent set of rows over* $GF(2)$*) that contains it; the minimum weight of a row of the solution base in this circuit is the value of a minimum weight solution to this problem.*

It is interesting to note that the matrices A for the two types of cut problems that we consider in this paper each have an exponential number of columns in n, although they have a polynomial rank in n, for fixed k. Hence the corresponding maximum solution bases of the matrices A are not "compact" representations of all the solution values. This leads us to make the following definition.

A matrix R is called a *representation matrix* for a matrix A if and only if

1. R and A have the same number of rows and
2. a set of rows of R is independent if and only if the corresponding set of rows of A is independent.

Clearly a representation matrix R for A can be substituted for A in the statement of Theorem 3.2. Hence one of our objectives is to find representation matrices for our two cut problems whose maximum weight bases require a "small" amount of space. In particular, we construct representation matrices whose bases have a size that is not only polynomial in n, for fixed k, but which can be stored in at most as much space as the maximum number of different min weight solutions. The way we accomplish this is to choose representation matrices that have a "small" number of 1's in each row (that is, a number of 1's that is bounded by a constant in k). We will use the following simple proposition to help us identify representation matrices.

PROPOSITION 3.3. *Let A, B, and C be three matrices such that* $BC = A$ *and* $\text{rank}(B) = \text{rank}(A)$*. Then B is a representation matrix for A.*

Proof. For a matrix M (over any field), let *colsp(M)* denote the vector space generated by the columns of M. Then $BC = A$ implies $colsp(A) \subseteq colsp(B)$. Since $\text{rank}(B) = \text{rank}(A)$, we can conclude $colsp(A) = colsp(B)$. Let B' be a subset of rows of B and let A' be the corresponding subset of rows of A. Then $colsp(B') = colsp(A')$, which implies that $\text{rank}(B') = \text{rank}(A')$. It follows that the rows of $B'$are independent

if and only if the rows of $A'$are independent. Thus B is a representation matrix for A. □

4. Proof of the main results for *k*-terminal cut problems. In this section we prove Theorems 2.3 and 2.4.

For the collection of k-terminal cut problems on a set V, consider the following: the associated matrix A (as defined in section 3) and the associated CR matrix M (as defined in section 2). Assume that corresponding rows of A and M refer to the same k-terminal cut problem. The main result of this section is the following.

THEOREM 4.1. *M is a representation matrix for A.*

This is our main result because Theorem 2.3 follows immediately.

COROLLARY 4.2. *Theorem* 2.3.

Proof of Corollary 4.2. The corollary follows immediately from Theorem 4.1 and Theorem 3.2. □

We use Proposition 3.3 from the previous section to prove Theorem 4.1. Thus we must produce a matrix, say, D, such that $MD = A$, and we must show that $\text{rank}(M) = \text{rank}(A)$. We make use of the following result of Hassin.

THEOREM 4.3 (see Theorem 3.2 in [19]). *For the k-terminal cut problem,* $\text{rank}(A) = \binom{n-1}{k-1}$.

We define the matrix D as follows.

Let D be a matrix whose rows are indexed on the subsets of V of size $k-1$ (hence D has $\binom{n}{k-1}$ rows) and whose columns are indexed on the k-cuts. Let $(V_1, \ldots, V_k)$ denote an arbitrary k-cut, and let $s(V_1, \ldots, V_k)$ denote the corresponding column in D. Then $s(V_1, \ldots, V_k)$ is defined to be the 0–1 incidence vector of the subsets $V'' \subset V$ for which the following two conditions hold.

1. $|V''| = k - 1$.
2. If k is odd: $|V'' \cap V_i| \leq 1$, $1 \leq i \leq k$;
   If k is even: $|V'' \cap V_i| = 1$, $1 \leq i \leq k-1$.

Hence $V'' \cap V_k = \emptyset$, when k is even. Let $\mathbf{S}$ denote the set of all k-cut solutions and, for $\mathbf{S}' \subseteq \mathbf{S}$, let $s(\mathbf{S}') = \{s(V_1, \ldots, V_k) : (V_1, \ldots, V_k) \in \mathbf{S}'\}$.

Let us extend this notation to the matrix M. Recall that the columns of M are indexed on the subsets of V of size $k-1$. Let us assume these subsets occur in the same order for M, from left to right, as they do for D, from top to bottom. Each row of M corresponds to a k-terminal cut problem, that is, a subset $V' \subseteq V$, such that $|V'| = k$. Let us denote each such row as $p(V')$. Thus $p(V')$ is a 0–1 incidence vector of the subsets of V' of size $k-1$. Let $\mathbf{P}$ denote the set of all k-terminal cut problems and, for $\mathbf{P}' \subseteq \mathbf{P}$, let $p(\mathbf{P}') = \{p(V') : V' \in \mathbf{P}'\}$.

PROPOSITION 4.4. *$MD = A$.*

Proof. Let $\mathbf{P}$ and $\mathbf{S}$ denote the collections of all k-terminal cut problems and k-cut solutions, respectively, on V. Then for arbitrary $V' \in \mathbf{P}$ and $(V_1, \ldots, V_k) \in \mathbf{S}$, it suffices to show that $(V_1, \ldots, V_k)$ is feasible for V' if and only if $p(V') \cdot s(V_1, \ldots, V_k) = 1$.

($\Rightarrow$) Assume $(V_1, \ldots, V_k)$ is feasible for V'.

Assume that $k = |V'|$ is odd. $p(V')$ has a 1 in precisely those entries corresponding to subsets of V' of size $k-1$. Because $(V_1, \ldots, V_k)$ is feasible for V', every entry for which $p(V')$ is 1 is also a 1 in $s(V_1, \ldots, V_k)$. The number of such entries is $|V'|$. Since $|V'|$ is odd, $p(V') \cdot s(V_1, \ldots, V_k) = 1$.

Assume that $k = |V'|$ is even. Observe that there is only one entry that is 1 in both $p(V')$ and $s(V_1, \ldots, V_k)$: the entry corresponding to the subset of V' of size $k-1$ that does not intersect V_k. Hence $p(V') \cdot s(V_1, \ldots, V_k) = 1$.

($\Leftarrow$) Assume $(V_1, \ldots, V_k)$ is not feasible for V'.

Assume that $k = |V'|$ is odd. There must exist at least one set in $V_1, \ldots, V_k$ that contains at least two members of V'. If there is exactly one set in $V_1, \ldots, V_k$ that contains exactly two members of V' and all others contain zero or one, then there are precisely two entries that are 1 in both $p(V')$ and $s(V_1, \ldots, V_k)$. Hence $p(V') \cdot s(V_1, \ldots, V_k) = 0$. In all other cases, there are no entries that are 1 in both $p(V')$ and $s(V_1, \ldots, V_k)$, yielding the same conclusion.

Assume that $k = |V'|$ is even. The argument is exactly the same as above except that if V_k contains exactly two members of V' and each of $V_1, \ldots, V_{k-1}$ contains zero or one, then there are no entries that are 1 in both $p(V')$ and $s(V_1, \ldots, V_k)$. □

PROPOSITION 4.5. $\text{rank}(M) = \binom{n-1}{k-1}$.

Proof. Let $\mathbf{P}$ and $\mathbf{S}$ denote the collections of all k-terminal cut problems and k-cut solutions, respectively, on a set V. Then we want to show that $\text{rank}(p(\mathbf{P})) = \binom{n-1}{k-1}$. Arbitrarily pick $v \in V$. Let $\mathbf{P}_v$ be the collection of k-terminal cut problems that contain v. Observe that $|\mathbf{P}_v| = \binom{n-1}{k-1}$. We claim that $p(\mathbf{P}_v)$ is a base for $p(\mathbf{P})$. First, observe that the vectors in $p(\mathbf{P}_v)$ are independent since $p(V')$, for each $V' \in \mathbf{P}_v$, has a 1 in the position indexed by $V' \setminus \{v\}$ and all other vectors in $p(\mathbf{P}_v)$ have a 0 in this position. Second, to show that $p(\mathbf{P}_v)$ spans $p(\mathbf{P})$, consider an arbitrary k-terminal cut problem $V' \in \mathbf{P}$ such that $v \notin V'$. Consider the subsets $\mathbf{P}'_v = \{V'' : V'' \in \mathbf{P}_v \text{ and } V'' \subset V' \cup \{v\}\}$. We claim that the sum of the vectors in $p(\mathbf{P}'_v)$ equals $p(V')$. Note that the vectors in $p(\mathbf{P}'_v)$ can contain a 1 only in positions indexed by subsets $V'' \subset V' \cup \{v\}$, where $|V''| = k - 1$. First, consider an arbitrary such position V'' such that $v \notin V''$. The only vector in $p(\mathbf{P}'_v)$ that is 1 in such a position is $p(V'' \cup \{v\})$. Hence the sum of the vectors in $p(\mathbf{P}'_v)$ is 1 in such a position. Second, consider an arbitrary such position V'' such that $v \in V''$. Let $\{v_1, v_2\} = V' \setminus V''$. The only vectors in $p(\mathbf{P}'_v)$ that are 1 in such a position are $p(V'' \cup \{v_1\})$ and $p(V'' \cup \{v_2\})$. Hence the sum of the vectors in $p(\mathbf{P}'_v)$ is 0 in such a position. Thus, $p(\mathbf{P}_v)$ is a base for $p(\mathbf{P})$. □

Proof of Theorem 4.1. The theorem follows from Theorem 4.3 and Propositions 3.3, 4.4, and 4.5. □

Proof of Theorem 2.4. By Proposition 4.5, a CR matrix has $\binom{n-1}{k-1}$ rows. By definition, each row has k 1's. Also by definition, a CR matrix has $\binom{n}{k-1}$ columns to which we can assign names. Thus a CR matrix can be stored in $k\binom{n-1}{k-1}$ space by replacing each row with a list of the names of the k columns that contain 1's. The result follows. □

5. The k-pair cut problem.

5.1. The problem and our results. As noted in the introduction, our second problem is related to the well-known multicommodity flow problem. Given a ground set V, a *k-pair cut problem* is a collection of (not necessarily distinct) pairs $\{s_i, t_i\}_{i=1}^k$ of V, where $s_i \neq t_i$. (In the multicommodity flow problem there are k commodities, each of which must be sent between the corresponding pair of nodes in a network. Some commodities may have the same corresponding pair of nodes. However, in our more general setting, we do not assume a network structure exists.) For this problem, we consider only 2-*cut solutions*; that is, partitions (V_1, V_2) of V into two nonempty sets. Each 2-cut solution is given a real *weight*. A 2-cut solution (V_1, V_2) is called *feasible* for a k-pair cut problem $\{s_i, t_i\}_{i=1}^k$ if and only if s_i and t_i are in different sets of the partition for every $i = 1, \ldots, k$. So the objective for a given k-pair cut problem is to find a min weight feasible 2-cut solution. Note that a 1-pair cut problem

is always equivalent to a 2-terminal cut problem. However, a k-pair cut problem for $k > 1$ is also equivalent to a 2-terminal cut problem if all k pairs are identical.

Let us address the question of how many different k-pair cut problems there are for a set V. There are $\binom{n}{2}$ different k-pair cut problems with one distinct pair among the $\{s_i, t_i\}_{i=1}^k$. There are $\binom{\binom{n}{2}}{2}$ different k-pair cut problems with two distinct pairs among the $\{s_i, t_i\}_{i=1}^k$. Since there cannot be more than $\binom{n}{2}$ distinct pairs, there are

$$\sum_{m=1}^{\min\left(k,\binom{n}{2}\right)} \binom{\binom{n}{2}}{m}$$

different k-pair cut problems. Hassin showed that there exists a significantly smaller set of min weight 2-cut solutions to these problems.

DEFINITION 5.1. *Let* $s = \sum_{m=1}^{\min(k,n-1)} \binom{n-1}{m}$.

THEOREM 5.2 (see [19]). *There exists a set of at most* s 2*-cut solutions that contains a min weight* 2*-cut solution for every* k*-pair cut problem. (Furthermore, this result is the "best possible.")*

Next we present a structure that compactly represents all the solutions to the k-pair cut problems. As for the k-terminal cut problem, the structure is a matrix. We call this matrix *compact* because (as we show) it can be stored in $O(s)$ space, for fixed k. The matrix for the 1-pair cut problem is identical to the matrix we constructed for the 2-terminal cut problem. (These two problems are identical.)

Before defining these matrices, it will be convenient to use the following notion. For a given k-pair cut problem consider the following graph (with no multiple edges): $G\,(V, \{\{s_i, t_i\} \text{ for i} = 1, \ldots, \text{k}\})$. Thus G has from 1 to k edges. Clearly this problem has a feasible solution if and only if G has no odd cycles, i.e., G is bipartite. Hence we may equivalently describe a feasible k-pair cut problem as a collection of pairs of nonempty sets $(A_j, B_j)_{j=1}^r$ such that all $2r$ of these sets are pairwise disjoint and each pair (A_j, B_j) is the node partition for one of the (nonsingleton) components of the bipartite graph G. Note that r can range from 1 to k. A 2-cut solution is feasible for a k-pair cut problem if and only if A_j and B_j are in different sets of the partition for every $j = 1, \ldots, r$.

Consider the k-pair cut problems on a set V. Let M' be a $q \times \sum_{m=1}^k \binom{n}{m}$ matrix whose rows are indexed on the feasible k-pair cut problems on V and whose columns are indexed on the subsets of V of size $\leq k$. (It is not necessary to explicitly calculate q.) For a feasible problem $\{s_i, t_i\}_{i=1}^k$ (or $(A_j, B_j)_{j=1}^r$) the corresponding row of M' is the incidence vector of all subsets $S \subseteq \{s_1, t_1, \ldots, s_k, t_k\}$ such that

$$A_j \subseteq S \text{ or } B_j \subseteq S, \text{but not both, for } j = 1, \ldots, r. \tag{5.1}$$

(Note that, for example, $A_j \subseteq S$ and $B_j \cap S \neq \emptyset$ is allowed. We show below that $|S|$ is always $\leq k$.)

Let the *weight* of each row of M' be the solution value of the corresponding k-pair cut problem, and let the *weight* of a base of M' be the sum of the weights of the rows in the base. A matrix consisting of the rows in a maximum weight base of M' is called a *CR matrix* for the solutions to all the k-pair cut problems.

Example 3. When $k = 1$, the matrix M' is the edge-node incidence matrix for a complete graph constructed on V. Hence (as in Example 1 for the 2-terminal cut problem) a CR matrix here corresponds to Hassin's CR tree and each row contains two nonzeros.

Example 4. For $k = 2$, consider again the complete graph constructed on V. The columns of M' can be partitioned into two sets that correspond to the nodes and edges of the complete graph. The rows of M' can also be partitioned into two sets that correspond to the individual edges and to the pairs of distinct edges of the complete graph. The submatrix of M' whose rows correspond to the edges and whose columns correspond to the nodes is the edge-node incidence matrix for the complete graph. The submatrix of M' whose rows correspond to the pairs of edges and whose columns correspond to the edges is the incidence matix of triangles and squares in the well-known cycle space. To see this consider the following: If a row corresponds to two adjacent edges of the form (s_1, t_1), (s_1, t_2), then it contains 1's in the columns that correspond to $\{s_1\}$, $\{s_1, t_1\}$, $\{s_1, t_2\}$, and $\{t_1, t_2\}$. If a row corresponds to two nonadjacent edges of the form (s_1, t_1), (s_2, t_2), then it contains 1's in the columns that correspond to $\{s_1, s_2\}$, $\{s_1, t_2\}$, $\{s_2, t_1\}$, and $\{t_1, t_2\}$. Hence, every row of M' contains two or four nonzeros.

Next we state our main results for k-pair cuts. The first result is that the solution value to any k-pair cut problem can be found from the CR matrix. The second result is that the CR matrix is compact in the sense that it requires the same amount of space as the maximum number of different k-cut solutions, for fixed k. We prove these results in the next section.

THEOREM 5.3. *The value of a min k-pair cut solution for any (feasible) k-pair cut problem can be obtained from a CR matrix as follows: add the incidence vector for the problem (as it occurs in M') to the CR matrix and find the unique circuit that contains it; the minimum weight of a row of the CR matrix in this circuit is the value of a min k-pair cut solution for the problem.*

THEOREM 5.4. *A CR matrix for the k-pair cut problems on V can be stored in $O(s)$ space, for fixed k.*

5.2. Proofs. In this section we prove Theorems 5.3 and 5.4. Let A be the appropriate matrix as defined in section 3, and let M' be the matrix defined in section 5.1. Assume the corresponding rows of A and M' refer to the same k-pair problem. The main result of this section is the following.

THEOREM 5.5. *M' is a representation matrix for A.*

This is our main result because Theorem 5.3 follows immediately.

COROLLARY 5.6. *Theorem* 5.3.

Proof of Corollary 5.6. The corollary follows immediately from Theorem 5.5 and Theorem 3.2. ∎

We use Proposition 3.3 to prove Theorem 5.5. Thus we must produce a matrix D' such that $M'D' = A$, and we must show that $\text{rank}(M') = \text{rank}(A)$. We make use of the following result of Hassin.

THEOREM 5.7 (see Theorem 3.4 in [19]). *For the k-pair cut problem,* $\text{rank}(A) = s$.

Next we show that the matrices M' from section 5.1 are well defined.

PROPOSITION 5.8. *Consider a feasible k-pair cut problem $\{s_i, t_i\}_{i=1}^k$ (or $(A_j, B_j)_{j=1}^r$) and the subsets $S \subseteq V$, defined in* (5.1). *Then, for each such S, $|S| \leq k$.*

Proof. Let G denote the graph defined in section 5.1 for $\{s_i, t_i\}_{i=1}^k$, and let $C_j = (V_j = A_j \cup B_j, E_j)$, for $j = 1, \ldots, r$, denote the (nonsingleton) components of G. Observe that, by definition,

$$|S| \leq \sum_{j=1}^{r} (|A_j| + |B_j| - 1) .$$

Also observe that, since each C_j is connected,

$$|V_j| - 1 \leq |E_j| \quad \text{for } j = 1, \ldots, r.$$

Thus we have

$$\sum_{j=1}^{r} (|V_j| - 1) \leq \sum_{j=1}^{r} |E_j|.$$

It follows that

$$|S| \leq \sum_{j=1}^{r} (|A_j| + |B_j| - 1) = \sum_{j=1}^{r} (|V_j| - 1) \leq \sum_{j=1}^{r} |E_j| \leq k. \qquad \square$$

We define the matrix D' as follows.

Let D' be a matrix whose rows are indexed on the subsets of V of size $\leq k$ (in the same order from top to bottom as the columns of M' from left to right). Let the columns of D' be indexed on the 2-cuts. In particular, for each 2-cut (V_1, V_2) let $s'(V_1, V_2)$ denote the corresponding column in D'. Then $s'(V_1, V_2)$ is defined to be the 0–1 incidence vector of the subsets $V'' \subset V$ such that $|V''| \leq k$ and $V'' \subseteq V_1$. For a problem $\{s_i, t_i\}_{i=1}^{k}$ (or $(A_j, B_j)_{j=1}^{r}$), let $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ (or $p'(A_j, B_j)_{j=1}^{r}$) denote the corresponding row of M'.

PROPOSITION 5.9. *$M'D' = A$.*

Proof. Let $\{s_i, t_i\}_{i=1}^{k}$ (or $(A_j, B_j)_{j=1}^{r}$) be an arbitrary k-pair cut problem, and let (V_1, V_2) be an arbitrary 2-cut. Then it suffices to show that (V_1, V_2) is feasible for $\{s_i, t_i\}_{i=1}^{k}$ if and only if $p'\left(\{s_i, t_i\}_{i=1}^{k}\right) \cdot s'(V_1, V_2) = 1$.

To begin, let us assume $r = 1$. Recall that $s'(V_1, V_2)$ is the incidence vector for all subsets of V_1 of size $\leq k$ and $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ is the incidence vector for all subsets of $A_1 \cup B_1$ of size $\leq k$ that either contain A_1 but not all of B_1 or contain B_1 but not all of A_1. We consider the following cases.

Case 1. Suppose (V_1, V_2) is feasible for (A_1, B_1). Let us assume, without loss of generality, that $A_1 \subseteq V_1$. Then there is only one common subset between those indexed by $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$, namely, the set A_1. The result follows.

Case 2. Suppose $A_1 \cup B_1$ is contained in V_1 or V_2. If $A_1 \cup B_1$ is contained in V_2, then $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ have no common subsets. Suppose $A_1 \cup B_1$ is contained in V_1. The number of subsets indexed by $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ that contain A_1 is equal to the number of proper subsets of B_1, which is $2^{|B_1|} - 1$. The number of subsets indexed by $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ that contain B_1 is equal to the number of proper subsets of A_1, which is $2^{|A_1|} - 1$. Hence the total number of subsets indexed by both $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ is $\left(2^{|A_1|} - 1\right) + \left(2^{|B_1|} - 1\right)$, which is even. The result follows.

Case 3. Suppose V_1 (hence, V_2) contains a proper, nonempty subset of nodes in both A_1 and B_1. Then $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ index no common subsets and the result follows.

Case 4. Suppose V_1 contains A_1 and a proper, nonempty subset of nodes in B_1. Let B' denote the subset of B_1 in V_1. Then $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^{k}\right)$ have

$2^{|B'|}$ common subsets. Conversely, if V_2 contains A_1 and a proper, nonempty subset of nodes in B_1, then $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^k\right)$ index no common subsets. Analogous arguments hold if V_1 contains B_1 and a proper, nonempty subset of nodes in A_1. The result follows.

Next we consider the case that $r > 1$. Let $p'(A_j, B_j)$ denote the incidence vector associated with the problem defined by (A_j, B_j). Let S_j denote the sets that are indexed by both the vectors $s'(V_1, V_2)$ and $p'(A_j, B_j)$ for $j = 1, \ldots, r$. Then the sets that are indexed by both the vectors $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^k\right)$ are precisely those sets that can each be obtained as follows: take one set from each of S_j for $j = 1, \ldots, r$ and then take the union of these sets. Hence the number of sets that are common to the vectors $s'(V_1, V_2)$ and $p'\left(\{s_i, t_i\}_{i=1}^k\right)$ is equal to $\prod_{j=1}^r |S_j|$. From the $r = 1$ part of this proof, we know that $|S_j|$ equals 1 if (V_1, V_2) separates (A_j, B_j) and is even otherwise. Hence $\prod_{j=1}^r |S_j|$ equals 1 if (V_1, V_2) is feasible for $\{s_i, t_i\}_{i=1}^k$ and is even otherwise. The result follows. □

As before, if $\mathbf{S}$ denotes the set of all 2-cut solutions, then for $\mathbf{S'} \subseteq \mathbf{S}$ let $s'(\mathbf{S'}) = \{s(V_1, V_2) : (V_1, V_2) \in \mathbf{S'}\}$. Similarly, if $\mathbf{P}$ denotes the set of all k-pair cut problems, then for $\mathbf{P'} \subseteq \mathbf{P}$ let $p'(\mathbf{P'}) = \left\{p'\left((A_j, B_j)_{j=1}^r\right) : (A_j, B_j)_{j=1}^r \in \mathbf{P'}\right\}$.

PROPOSITION 5.10. $\text{rank}(M') = s$.

Proof. Pick an arbitrary $v \in V$. Let P_v denote all the k-pair problems of the form $\{\{v, t\} : t \in T\}$ or, equivalently, (v, T), where $v \notin T$. Thus T ranges over all subsets of $V \backslash v$ of size $\leq k$. Observe that $|P_v| = \sum_{m=1}^{\min(k, n-1)} \binom{n-1}{m} = s$. We show that the vectors $p'(P_v)$ are independent, and then we show that they span all the vectors in $\mathbf{P}$.

Consider an arbitrary subset, say, B, of $V \backslash v$ of size $\leq k$. Observe that there is only one vector in $p'(P_v)$ that has a 1 in the entry indexed by B, namely, $p'(v, B)$. Thus $p'(P_v)$ contains an identity matrix of size s. Hence the vectors in $p'(P_v)$ are independent.

Consider an arbitrary k-pair problem $(A_j, B_j)_{j=1}^r$.

Case 1. To begin, let us assume $r = 1$ and $v \notin A_1 \cup B_1$.

Let P'_v denote the k-pair problems (v, T) in P_v such that $p'(A_1, B_1)$ has a 1 in position T. Let $sum(p'(P'_v))$ denote the sum of the vectors in $p'(P'_v)$. We show that $sum(p'(P'_v)) = p'(A_1, B_1)$. Let (v, T) denote an arbitrary problem in P'_v. As we noted above in constructing the identity matrix, $p'(v, T)$ is the only vector in $p'(P'_v)$ that has a 1 in position T. Thus $sum(p'(P'_v))$ has a 1 in every position in which $p'(A_1, B_1)$ has a 1. Also note that every vector in $p'(P'_v)$ has a 1 in position $\{v\}$. We showed in the proof of Proposition 5.9 (Case 2) that $p'(A_1, B_1)$ has an even number of 1's, hence $|p'(P'_v)|$ is even and $sum(p'(P'_v))$ has a 0 in position $\{v\}$. It remains to show that, for any problem (v, T) in P'_v, the positions in $p'(v, T)$ that are 1 and do not correspond to T or v occur in an even number of vectors in $p'(P'_v)$. Such positions correspond to subsets of the form $\{v\} \cup T'$, where T' is a proper subset of T. Such positions are equal to 1 for all vectors in $p'(P'_v)$ with problems of the form (v, T''), where $T' \subsetneq T'' \subsetneq A_1 \cup B_1$, and either

1. $A_1 \subseteq T''$ or
2. $B_1 \subseteq T''$, but not both.

The number of vectors of type 1 is zero (if $B_1 \subseteq T'$) or $2^{|B_1 \backslash T'|} - 2$ (otherwise). The number of vectors of type 2 is zero (if $A_1 \subseteq T'$) or $2^{|A_1 \backslash T'|} - 2$ (otherwise). The total number of such vectors is the sum of these two numbers, which is even.

Case 2. Now let us assume $r > 1$ and $v \notin \bigcup_{j=1}^r (A_j \cup B_j)$. Let P'_v denote the

k-pair problems (v,T) in P_v such that $p'(A_j,B_j)_{j=1}^r$ has a 1 in position T. We again must show that $sum(p'(P'_v)) = p'(A_j,B_j)_{j=1}^r$. As above, $sum(p'(P'_v))$ has a 1 in every position in which $p'(A_j,B_j)_{j=1}^r$ has a 1. Also, as above, $sum(p'(P'_v))$ has a 0 in position v. If we let (v,T) denote an arbitrary problem in P'_v, then it remains to show that the positions in $p'(v,T)$ that are 1 and do not correspond to T or v occur in an even number of vectors in $p'(P'_v)$. Such positions correspond to subsets of the form $\{v,T'\}$, where T' is a proper subset of T. Each set T' can be expressed as the union of one set for each j of the form $T''_j = T' \cap (A_j \cup B_j)$. Using the same reasoning as in Case 1, the number of sets of type T''_j can be expressed as $a_j + b_j$, where $a_j =$ zero or $2^{|B_j \backslash T'|} - 2$, and $b_j =$ zero or $2^{|A_j \backslash T'|} - 2$.

Hence, the number of vectors in $p'(P'_v)$ with a 1 in a position $\{v,T'\}$ is

$$\prod_{j=1}^{r} (a_j + b_j),$$

which is even.

Case 3. Next let us assume $r = 1$ and $v \in A_1 \cup B_1$. We may additionally assume that $v \in A_1$ and $|A_1| > 1$. Let P'_v denote the k-pair problems (v,T) in P_v such that $p'(A_1,B_1)$ has a 1 in position T. (Note that the sets T we consider here are different from the sets T considered in Case 1 in that there is one member of $A_1 \cup B_1$ they never contain, namely v.) Let $sum(p'(P'_v))$ denote the sum of the vectors in $p'(P'_v)$. We show that $sum(p'(P'_v)) = p'(A_1,B_1)$. Let (v,T) denote an arbitrary problem in P'_v. As we noted above in constructing the identity matrix, $p'(v,T)$ is the only vector in $p'(P'_v)$ that has a 1 in position T. Thus $sum(p'(P'_v))$ has a 1 in every position in which $p'(A_1,B_1)$ has a 1 with an index T that does not contain v, that is, for all indices T such that $B_1 \subseteq T$ and $v \notin T$.

Let us consider the remaining indices of $p'(A_1,B_1)$ that have value 1. These indices T' must satisfy $B_1 \subseteq T'$ and $v \in T'$, or they must satisfy $A_1 \subseteq T'$. The number of problems (v,T) that have a 1 in an index that satisfies the first condition is $2^{|A_1 \backslash v|} - 1$, and the number that satisfies the second condition is 1 (just $(v, A_1 \cup B_1 \backslash v)$). Both of these numbers are odd, as required.

Finally, we must show that the positions in $p'(v,T)$ that are 1 and do not correspond to a 1 in $p'(A_1,B_1)$ occur in an even number of vectors in $p'(P'_v)$. Such positions correspond to subsets of the form $\{v,T'\}$, where $T' \subsetneq T$ and $B_1 \nsubseteq T'$. Each such position has value 1 in $2^{|A_1 \backslash v \backslash T|}$ vectors in $p'(P'_v)$, which is even.

Case 4. Next let us assume $r > 1$ and $v \in A_1 \cup B_1$. We may additionally assume that $v \in A_1$. The argument for this case is similar to the argument for Case 2. We leave it to the reader. □

Proof of Theorem 5.5. The theorem follows from Theorem 5.7 and Propositions 3.3, 5.9, and 5.10. □

Proof of Theorem 5.4. By Proposition 5.10, a CR matrix has s rows. From the definition, it is not difficult to see that each row contains at most 2^k 1's. (The maximum occurs when the k edges that define the problem are pairwise nonadjacent.) Also, by definition, a CR matrix has $\sum_{m=1}^{k} \binom{n}{m}$ columns to which we can assign names. Thus a CR matrix can be stored in $s2^k$ space by replacing each row with a list of the names of the at most 2^k columns that contain 1's. □

6. Construction of compact representations. Our objective in this section is to present an algorithm for constructing the compact representation matrices we

have described for the two cut problems. This algorithm generalizes Gomory and Hu's result 3 (given in section 1.1).

Our algorithm takes as input a representation matrix. The weights of the rows are initially unknown but can be found by solving the corresponding cut problem. Hence an important component of the complexity of the algorithm is how many problems have to be solved. Our algorithm is closely based on the one presented by Hassin [16], which constructs maximum solution bases (defined in section 2). However, our algorithm achieves a better complexity. In particular, our algorithm requires solving a smaller number of problems: the number of rows in a maximum solution base (or a CR matrix). Hassin's algorithm requires solving twice this number of problems. As we have seen, this number is polynomial for our examples, for fixed k.

With Hassin's algorithm, the complexity of the work in addition to solving the cut problems is exponential (since the algorithm utilizes the matrix A; see section 2). With our algorithm this additional work is polynomial (for fixed k) for our cut problems. This could be significant for an implementation of this algorithm. In particular, if a cut problem can be solved in polynomial time, then the entire compact representation can be constructed in polynomial time with our algorithm. An interesting example is the k-terminal cut problem on planar graphs for which there exists a polynomial time algorithm for fixed k (see [7]).

Finally, Hassin's algorithm requires that all solutions have different weights. Our algorithm does not require this assumption.

As in section 2, let us assume we have a finite set of *problems*, say, P, and a finite set of *solutions*, say, S. Each solution has a distinct real *weight* and we have a problem-solution incidence matrix A. Let M be a representation matrix for A. The *weight* of a problem equals the minimum weight of a solution that is feasible for that problem; hence each row of A and M has a weight corresponding to the problem. (As we noted, however, these row weights are unknown at the start of the algorithm.)

A matrix consisting of the rows in a maximum weight base of M is called a *CR matrix* for the solutions S. (Note that, in general, this matrix can have any number of columns and hence may not actually be "compact." It is compact, however, for the two cut problems studied in this paper.)

Next we present our algorithm. This algorithm is based on Algorithms 3.3 and 3.4 in Hassin [16]. One difference between our algorithm and Hassin's is that we do not explicitly construct a *minimal cover* (i.e., a set of solutions that contains a feasible solution for every problem), which partly accounts for the extra factor of 2 in the complexity of Hassin's algorithm. Another difference between the algorithms is that we input a representation matrix for A, instead of A itself. This can improve the complexity when the representation matrix is smaller than A. We also express the complexity of our algorithm in terms of elementary operations rather than solely in terms of the number of problems that are solved (as Hassin did).

Algorithm. Compact representation.

Input. A representation matrix M for problems P and solutions S; and weights w for the solutions.

Output. A set of rows $P' \subseteq P$ of M that is a CR matrix and the set $S' \subseteq S$ of min weight solutions.

Step 0. Set $S' := \emptyset$; $P' := \emptyset$.

Step 1. Set $P^* := \{p \in P : \text{rows } P' \cup \{p\} \text{ of } M \text{ are independent}\}$. If $P^* = \emptyset$, output P' and S'; done.

Step 2. For each $p \in P^*$: If there exists a feasible solution in S' for p, then set

$$w(p) = \min\{w(s) : s \in S' \text{and } s \text{ is feasible for } p\};$$

otherwise, set $w(p) := +\infty$.

Step 3. Let $p^* \in P^*$ achieve $\max\{w(p) : p \in P^*\}$. Set $P' := P' \cup \{p^*\}$.

Step 4. Find a min weight solution s^* for p^*. If $w(s^*) < w(p^*)$, set $S' := S' \cup \{s^*\}$. Go to Step 1.

End.

In order to prove the validity of the algorithm, we make use of the following procedure and two propositions. It seems most natural to prove these results by referring to some elementary results in matroid theory. For background the reader is referred to [1] and [31].

PROCEDURE. GENERALIZED GREEDY.

Input. A matrix M with row weights; an independent set B' of rows of M.

Step 1. Set $X := \emptyset$ and put the set of rows of $M\backslash B'$ into nonincreasing order by weight (break ties arbitrarily).

Step 2. Consider, in order, each row x of the rows of $M\backslash B'$ and do the following: If $B' \cup X \cup \{x\}$ is independent, set $X := X \cup \{x\}$.

End.

PROPOSITION 6.1. *Let B' be contained in a maximum weight base of the rows of a matrix M with row weights, and let X be the output of the generalized greedy procedure. Then $B' \cup X$ is a maximum weight base of the rows of M.*

Proof. Let N denote the matroid on the rows of M whose independent sets are the independent sets of rows of M. Consider the following collection of sets: $\{Y \subseteq \{\text{the rows of } M\backslash B'\} : B' \cup Y \text{ is independent}\}$. This collection of sets is known to form a matroid, say, N', on $M\backslash B'$ (i.e., the matroid obtained from N by "contracting" the elements of B'). Then the generalized greedy procedure is simply the well-known greedy algorithm on N' and hence finds a maximum weight base X of the matroid N'. The proposition claims that $B' \cup X$ is a maximum weight base for N. Clearly $B' \cup X$ is a base for N. To see that it has maximum weight, let B be a maximum weight base for N that contains B'. Suppose the weight of B exceeds the weight of $B' \cup X$. Then the weight of $B\backslash B'$ must exceed the weight of X. But $B\backslash B'$ is an independent set in N' and this contradicts our choice of X. □

PROPOSITION 6.2. *Let M be a representation matrix for a collection of problems and solutions. Let the rows of M have weights that derive from the minimum weight solutions to the corresponding problems. Consider the equivalence classes of rows defined as follows: two rows are in the same class if and only if they have the same weight. Let B' be contained in a maximum weight base of the rows of M and suppose that B' contains no row in some class, say, C. Then, for every $c \in C$, $B' \cup \{c\}$ is contained in a maximum weight base.*

Proof. Let B be a maximum weight base that contains B' and suppose $c \in C$, but $c \notin B$. Let D be the unique circuit formed by adding c to B. By Theorem 3.2 there exists an element of $D\backslash c$ that has the same weight as c, say, d. From basic matroid theory, $B\backslash c \cup d$ is also a base and, by our choice of d, it has the same weight as B. The result follows. □

THEOREM 6.3. *The compact representation algorithm works.*

Proof. Let us begin with an observation: If we change or perturb some of the weights of the solutions S by a very small amount and then find a maximum weight base of M, then this base will also have maximum weight under the original weights. Our first objective is to perturb these weights in a special way.

Let us partition the solutions S into equivalence classes so that two solutions are in the same class if and only if they have the same weight. Apply the algorithm. Whenever a solution s^* is added to S' in Step 4, do the following: Add a very small number ε to the weight of all solutions, except s^*, in the equivalence class that contains s^*. Thus this class is split into two classes, one of which contains only s^*.

Next consider a second application of the algorithm, but using the new solution weights. Note that in this application we can choose the same sequence of problems p^* in Step 3 and, due to our choice of weights, the solutions s^* that we find in Step 4 are unique; so let us assume that we do this. Now we show that the algorithm constructs a maximum weight base under the new weights, and hence, by our observation above, a maximum weight base under the original weights.

Let us assume, inductively, that we are entering Step 1 with a set of rows P' that is contained in a maximum weight base of M. It suffices to show that in Step 3, $P' \cup p^*$ is contained in a maximum weight base of M. To see this, observe that two things can happen in Step 4: either $w(s^*) < w(p^*)$ or $w(s^*) = w(p^*)$. If we find that $w(s^*) < w(p^*)$, then $s^* \notin S'$; furthermore, since this min weight solution is unique, P' contains no problem in the equivalence class of p^* and Proposition 6.2 tells us that $P' \cup p^*$ is contained in a maximum weight base of M. We then add a new solution to the set S' in Step 4. Suppose we find that $w(s^*) = w(p^*)$. Observe that in all subsequent passes through Step 2, the value of w for any problem cannot increase. Thus p^* must be a maximum weight row in M that can be added to P' to form an independent set. Proposition 6.1 tells us that $P' \cup p^*$ is contained in a maximum weight base of M. □

In order to analyze the complexity of the algorithm, let us assume we have an oracle C that tells us if a particular solution is feasible for a particular problem. Let us say its worst case complexity is C^*. Let us also assume we have an oracle R that produces the minimum weight solution for any problem. Let R^* denote the worst case complexity of R. Let d denote the number of columns of M.

THEOREM 6.4. *The worst case time complexity of the compact representation algorithm is* $O\left(\operatorname{rank}(M)\left\{|P|\operatorname{rank}(M)d + |P|C^* + R^*\right\}\right)$.

Proof. The algorithm repeats Steps 1–4 a number of times equal to the final size of $|P'|$, which is $\operatorname{rank}(M)$. Consider a pass through Step 1. Assume we have performed Gauss–Jordan elimination on the rows P'. Then checking if the rows $P' \cup \{p\}$ are independent takes $O\left(\operatorname{rank}(M)d\right)$ elementary operations. We must make $O\left(|P|\right)$ such checks; hence this step requires $O\left(|P|\operatorname{rank}(M)d\right)$ time (including the amount of time to put the rows of P' back into Gauss–Jordan form in Step 3). Because S' has at most one solution added to it in each iteration, Step 2 requires only checking to see if the newest addition to S' is feasible for at most $|P|$ problems. Hence this step requires $O(|P|C^*)$ time. Step 3 requires $O(|P|)$ time, which is dominated by other steps. Step 4 requires R^* time. The result follows. □

Finally, we can see how the complexity of the compact representation algorithm specializes to the two cut problems we have studied in this paper.

For the k-terminal cut problem we have the following values:

- $\operatorname{rank}(M) = \binom{n-1}{k-1}$,
- $|P| = \binom{n}{k}$,
- $d = \binom{n}{k-1}$,
- $C^* = O(k)$ (if the problems and solutions are stored as subsets of V).

For the k-pair cut problem we have the following values:

- $\operatorname{rank}(M) = \sum_{m=1}^{\min(k,n-1)} \binom{n-1}{m}$,

- $|P| = O\left(\sum_{m=1}^{\min\left(k,\binom{n}{2}\right)} \binom{\binom{n}{2}}{m}\right)$ (note that not all problems are feasible),
- $d = \sum_{m=1}^{k} \binom{n}{m}$,
- $C^* = O(k)$ (if the problems and solutions are stored as subsets of V).

Hence for both problems we solve $\text{rank}(M)$ cut problems and the additional work is polynomial for fixed k.

7. Open problems. In this section we present the following two open problems for the k-terminal cut and k-pair cut problems considered in this paper.

1. What is the best complexity of actually finding a min-cut value from a compact representation?
2. Can Gomory and Hu's result on cut trees be generalized; that is, does there exist a compact structure from which not only a min-cut value but also the actual min-cut can be obtained in polynomial time?

Acknowledgment. The author would like to thank Refael Hassin for his helpful comments on this paper.

REFERENCES

[1] R.E. Bixby, *Matroids and operations research*, in Advanced Techniques in the Practice of Operations Research, H.S. Greenberg, F.H. Murphy, and S.H. Shaw, eds., North-Holland, New York, 1982, pp. 333–458.
[2] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows*, Prentice-Hall, Englewood Cliffs, NJ, 1993.
[3] C.K. Cheng and T.C. Hu, *Maximum concurrent flows and minimum cuts*, Algorithmica, 8 (1992), pp. 233–249.
[4] S. Chopra and M.R. Rao, *On the multiway cut polyhedra*, Networks, 21 (1991), pp. 51–89.
[5] H.P. Crowder and M.W. Padberg, *Solving large-scale symmetric travelling salesman problems to optimality*, Management Sci., 26 (1980), pp. 495–509.
[6] W.H. Cunningham, *The optimal multiterminal cut problem*, DIMACS Ser. Discrete Math. Theoret. Comput. Sci., 5 (1991), pp. 105–120.
[7] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis, *The complexity of multiterminal cuts*, SIAM J. Comput., 23 (1994), pp. 864–894.
[8] M. Fiala, *The minimum 3-cut problem: An application of polyhedral combinatorics*, Honors Project, Carleton University, 1986.
[9] O. Goldschmidt and D.S. Hochbaum, *A polynomial algorithm for the k-cut problem for fixed k*, Math. Oper. Res., 19 (1994), pp. 24–37.
[10] R.E. Gomory and T.C. Hu, *Multi-terminal network flows*, J. Soc. Indust. Appl. Math., 9 (1961), pp. 551–570.
[11] F. Granot and R. Hassin, *Multi-terminal maximum flows in node-capacitated networks*, Discrete Appl. Math., 13 (1986), pp. 157–163.
[12] D. Gusfield, *Very simple methods for all pairs network flow analysis*, SIAM J. Comput., 19 (1990), pp. 143–155.
[13] J. Hao and J.B. Orlin, *A faster algorithm for finding a minimum cut in a graph*, J. Algorithms, 17 (1994), pp. 424–446.
[14] D. Hartvigsen, *Minimum path bases*, J. Algorithms, 15 (1993), pp. 125–142.
[15] D. Hartvigsen, *Generalizing the all-pairs min cut problem*, Discrete Math., 147 (1995), pp. 151–169.
[16] R. Hassin, *An algorithm for computing maximum solution bases*, Oper. Res. Lett., 9 (1990), pp. 315–318.
[17] R. Hassin, *Multiterminal xcut problems*, Ann. Oper. Res., 33 (1989), pp. 215–225.
[18] R. Hassin, *Simultaneous solution of families of problems*, in Algorithms, Lecture Notes in Comput. Sci. 450, T. Asano, T. Ibarake, H. Imai, and T. Nishizeki, eds., Springer-Verlag, Tokyo, 1990, pp. 288–299.
[19] R. Hassin, *Solution bases of multiterminal cut problems*, Math. Oper. Res., 13 (1988), pp. 535–542.

[20] X. He, *An improved algorithm for the planar* 3*-cut problem*, J. Algorithms, 12 (1991), pp. 23–37.
[21] D.S. Hochbaum and D.B. Shmoys, *An* $O(|V|^2)$ *algorithm for the planar* 3*-cut problem*, SIAM J. Algebraic Discrete Methods, 6 (1985), pp. 707–712.
[22] D.S. Hochbaum and L. Tsai, *A greedy algorithm for the* 3*-cut problem and its worst-case bound*, Tech. Rep. IP-318, University of California, Berkeley, 1983.
[23] T.C. Hu, *Multicommodity network flows*, Oper. Res., 11 (1963), pp. 344–360.
[24] H. Nagamochi and T. Ibaraki, *Computing edge-connectivity in multigraphs and capacitated graphs*, SIAM J. Discrete Math., 5 (1992), pp. 54–66.
[25] M.W. Padberg and M. Grötschel, *Polyhedral computations in the traveling salesman problem*, in The Traveling Salesman Problem, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, eds., John Wiley, Chichester, 1985, pp. 307–360.
[26] M.W. Padberg and G. Rinaldi, *Optimization of a* 532*-city symmetric traveling salesman problem*, Oper. Res. Lett., 6 (1987), pp. 1–7.
[27] J.C. Picard and M. Queyranne, *Selected applications of minimum cuts in networks*, INFOR, 20 (1982), pp. 394–422.
[28] A. Schrijver, *Min-max results in combinatorial optimization*, in Mathematical Programming, The State of the Art, A. Bachem, M. Grotschel, and B. Korte, eds., Springer-Verlag, Berlin, 1983, pp. 439–500.
[29] M. Stoer and F. Wagner, *A simple min cut algorithm*, J. ACM, 44 (1997), pp. 585–591.
[30] L. E. Trotter, *On the generality of multi-terminal flow theory*, Ann. Discrete Math., 1 (1977), pp. 517–525.
[31] D.J.A. Welsh, *Matroid Theory*, Academic Press, London, 1976.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 67–92

BETTER APPROXIMATION GUARANTEES FOR JOB-SHOP SCHEDULING*

LESLIE ANN GOLDBERG†, MIKE PATERSON†, ARAVIND SRINIVASAN‡, AND ELIZABETH SWEEDYK§

Abstract. Job-shop scheduling is a classical NP-hard problem. Shmoys, Stein, and Wein presented the first polynomial-time approximation algorithm for this problem that has a good (polylogarithmic) approximation guarantee. We improve the approximation guarantee of their work and present further improvements for some important NP-hard special cases of this problem (e.g., in the *preemptive* case where machines can suspend work on operations and later resume). We also present NC algorithms with improved approximation guarantees for some NP-hard special cases.

Key words. approximation, guarantees, job-shop, scheduling

AMS subject classifications. 68Q25, 68R05

PII. S0895480199326104

1. Introduction.

Job-shop scheduling is a classical NP-hard minimization problem [10]. We improve the approximation guarantees for this problem and for some of its important special cases, both in the sequential and parallel algorithmic domains; the improvements are over the current best algorithms of Leighton, Maggs, and Rao [11] and Shmoys, Stein, and Wein [21]. In job-shop scheduling, we have n jobs and m machines. A job consists of a sequence of operations, each of which is to be processed on a specific machine for a specified integral amount of time; a job can have more than one operation on a given machine. The operations of a job must be processed in the given sequence, and a machine can process at most one operation at any given

*Received by the editors March 22, 1999; accepted for publication (in revised form) October 19, 2000; published electronically January 16, 2001. A preliminary version of this work appears in *Proceedings of the Eighth Annual ACM–SIAM Symposium on Discrete Algorithms*, New Orleans, LA, 1997, ACM, New York, pp. 599–608.

http://www.siam.org/journals/sidma/14-1/32610.html

†Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK (leslie@dcs.warwick.ac.uk, msp@dcs.warwick.ac.uk). Part of the first author's work was performed at Sandia National Laboratories and was supported by the U.S. Department of Energy under contract DE-AC04-76AL85000. The work of the first author was also supported by ESPRIT Project RAND-II(21726) and by EPSRC grant GR/L60982. The work of the first and second authors was supported by ESPRIT Project ALCOM-IT(20244) and by EU Fifth Framework Project ALCOM–FT(IST-1999-14186).

‡Bell Laboratories, Lucent Technologies, 600-700 Mountain Avenue, Murray Hill, NJ 07974-0636 (srin@research.bell-labs.com). This work was done while this author was at (i) the National University of Singapore, supported in part by National University of Singapore Research Grants RP950662 and RP960620; (ii) Cornell University, Ithaca, NY, supported by an IBM Graduate Fellowship; (iii) the School of Mathematics, Institute for Advanced Study, Princeton, NJ, supported by grant 93-6-6 of the Alfred P. Sloan Foundation to the Institute for Advanced Study; (iv) DIMACS (NSF Center for Discrete Mathematics and Theoretical Computer Science), supported by NSF grant NSF-STC91-19999 to DIMACS and by support to DIMACS from the New Jersey Commission on Science and Technology; (v) the Sandia National Laboratories, New Mexico; (vi) the Department of Computer Science, University of Warwick, Coventry, UK; and (vii) the Department of Computer Science, University of Melbourne, Victoria, Australia, sponsored by a "Travel Grants for Young Asian Scholars" scheme of the University of Melbourne; this part of the work was done while this author was on study leave.

§Department of Computer Science, Harvey Mudd College, Olin Science Center, 301 E. Twelfth Street, Claremont, CA 91711-5980 (z@cs.hmc.edu). The research of this author was supported by an NSF Research Training grant. Part of this research was done while this author was visiting Sandia National Laboratories.

time. The problem is to schedule the jobs so that the *makespan*, the time when all jobs have been completed, is minimized. An important special case of this problem is *preemptive* scheduling, wherein machines can suspend work on operations, switch to other operations, and later resume the suspended operations (if this is not allowed, we have the *nonpreemptive* scenario, which we take as the default); in such a case, all operation lengths may be taken to be one. Even this special case with $n = m = 3$ is NP-hard, as long as the input is encoded concisely [16, 22]. We present further improved approximation factors for preemptive scheduling and related special cases of job-shop scheduling.

Formally, a job-shop scheduling instance consists of jobs $J_1, J_2, \ldots, J_n$, machines $M_1, M_2, \ldots, M_m$, and for each J_j, a sequence of μ_j *operations* $(M_{j,1}, t_{j,1})$, $(M_{j,2}, t_{j,2})$, $\ldots$, $(M_{j,\mu_j}, t_{j,\mu_j})$. Each operation is a (machine, processing time) pair: each $M_{j,k}$ represents some machine M_i, and the pair $(M_{j,i}, t_{j,i})$ signifies that the corresponding operation of job J_j must be processed on machine $M_{j,i}$ for an *uninterrupted* integral amount of time $t_{j,i}$. No machine can process more than one operation at a time; the operations of each given job must be scheduled in the given order. (For each job J_j, the waiting time from the completion of an operation $(M_{j,i}, t_{j,i})$ until the scheduling of $(M_{j,i+1}, t_{j,i+1})$ is allowed to be any nonnegative amount.) The problem that we focus on throughout this paper is to come up with a schedule that has a small makespan for general job-shop scheduling and for some of its important special cases.

1.1. Earlier work. As described earlier, even very restricted special cases of job-shop scheduling are NP-hard. Furthermore, the problem seems quite intractable in practice, even for relatively small instances. Call a job-shop instance *acyclic* if no job has more than one operation that needs to run on any given machine. A single instance of acyclic job-shop scheduling consisting of 10 jobs, 10 machines, and 100 operations resisted attempts at exact solution for 22 years, until its resolution by Carlier and Pinson [6]. More such exact solutions for certain instances (with no more than 20 jobs or machines) were computationally provided by Applegate and Cook, who also left open the exact solution of certain acyclic problems, e.g., some with 15 jobs, 15 machines, and 225 operations [3]. The reader is referred to Martin and Shmoys for a recent approach to computing optimal schedules for such problems [14].

Thus, efficient exact solution of all instances with, say, 30 jobs, 30 machines, and 900 operations seems quite beyond our reach at this point; an obvious next question is to look at efficient approximability. Define a ρ-approximation algorithm as a polynomial-time algorithm that always outputs a feasible schedule with a makespan of at most ρ times optimal; ρ is called the approximation guarantee. A negative result is known: if there is a ρ-approximation algorithm for job-shop scheduling with $\rho < 5/4$, then P = NP [23].

There are two simple lower bounds on the makespan of any feasible schedule: $P_{\max}$, the maximum total processing time needed for any job, and $\Pi_{\max}$, the maximum total amount of time for which any machine has to process operations. Recall the definition of acyclic job-shop scheduling given at the beginning of this subsection. For the NP-hard special case of acyclic job-shop scheduling wherein all operations have unit length, a breakthrough was achieved by Leighton, Maggs, and Rao in [11], showing that a schedule of makespan $O(P_{\max} + \Pi_{\max})$ always exists! (See sections 6.1 and 6.2 of Scheideler [17] for a shorter proof of this result.) Such a schedule can also be computed in polynomial time [12]. Feige and Scheideler have presented many new advances in acyclic job-shop scheduling [8].

What about upper bounds for general job-shop scheduling? It is not hard to

see that a simple greedy algorithm, which always schedules available operations on machines, delivers a schedule of makespan at most $P_{\max}\Pi_{\max}$; one would, however, like to aim for much better. Let $\mu = \max_j \mu_j$ denote the maximum number of operations per job, and let $p_{\max}$ be the maximum processing time of any operation. By invoking ideas from [11, 19, 20] and by introducing some new techniques, good approximation algorithms were developed in [21]. Their deterministic approximation bounds were slightly improved in [18] to yield the following proposition. (To avoid problems with small positive numbers, henceforth let $\log x$ denote $\log_2 x$ if $x \geq 2$ and 1 if $x < 2$; similarly, let $\log\log x$ denote $\log_2 \log_2 x$ if $x \geq 4$ and 1 if $x < 4$.)

PROPOSITION 1.1 (see [21, 18]). *There is a deterministic polynomial-time algorithm that delivers a schedule of makespan*

$$\mathrm{O}\left((P_{\max} + \Pi_{\max}) \cdot \frac{\log(m\mu)}{\log\log(m\mu)} \cdot \log(\min\{m\mu, p_{\max}\})\right)$$

for general job-shop scheduling. If we replace m by n in this bound, then such a schedule can also be computed in RNC.

This is a ρ-approximation algorithm with

$$\rho = \mathrm{O}(\log(m\mu)\log(\min\{m\mu, p_{\max}\})/\log\log(m\mu)).$$

See [21, 9] for further results on approximating some special cases of shop scheduling that are not discussed here. See [15] for definitions of the complexity classes NC and RNC.

1.2. Our results. Our first result improves Proposition 1.1 by a doubly logarithmic factor and provides further improvements for important special cases.

THEOREM 1.2. *There are the following deterministic algorithms for general job-shop scheduling, delivering schedules of makespan* $\mathrm{O}((P_{\max} + \Pi_{\max}) \cdot \rho)$:

(a) *a polynomial-time algorithm, with*

$$\rho = \frac{\log(m\mu)}{\log\log(m\mu)} \cdot \left\lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log(m\mu)} \right\rceil,$$

and if we replace m by n in this bound, then such a schedule can also be computed in NC;

(b) *a polynomial-time algorithm with*

$$\rho = \frac{\log m}{\log\log m} \cdot \log(\min\{m\mu, p_{\max}\}); \text{ and}$$

(c) *an NC algorithm with*

$$\rho = \frac{\log m}{\log\log m} \cdot \log(\min\{n\mu, p_{\max}\}).$$

Thus, part (a) improves on the previous approximation bound by a doubly logarithmic factor. The impact of parts (b) and (c) is best seen for preemptive scheduling, wherein $p_{\max} = 1$, and for the related situations where $p_{\max}$ is "small". Our motivation for focusing on these cases is twofold. First, preemptability is known to be a powerful primitive in various scheduling models; see, e.g., [4]. Second, the result of Leighton, Maggs, and Rao shows that preemptability is powerful for acyclic job-shops

(in the sense that there is a schedule of makespan $O(P_{\max} + \Pi_{\max})$ in the preemptive case). Recall that job-shop scheduling is NP-hard even when $n = m = 3$ and $p_{\max} = 1$. Parts (b) and (c) of Theorem 1.2 show that, as long as the number of machines is small or fixed, we get very good approximations. (It is trivial to get an approximation factor of m: our approximation ratio is $O(\log m / \log\log m)$ if $p_{\max}$ is fixed.) Note that for the case in which $p_{\max}$ is small, part (c) is both a derandomization and an improvement of the previous best parallel algorithm for job-shop scheduling (see Proposition 1.1).

We further explore the issue of when good approximations are possible, once again with a view to generalizing the result of Leighton, Maggs, and Rao [11]; this is done by the somewhat technical Theorem 1.3. In the statement of the theorem, "with high probability" means "with probability at least $1 - \epsilon$, for a positive constant ϵ." The failure probability ϵ can be made arbitrarily small (exponentially small in the size of the problem instance) by repeating the algorithm many times. Theorem 1.3 shows that (a) if no job requires too much of any given machine for processing, or (b) if repeated uses of the same machine by a given job are well-separated in time, then good approximations are possible. Say that a job-shop instance is *w-separated* if every distinct pair $((M_{j,\ell}, t_{j,\ell}), (M_{j,r}, t_{j,r}))$ of operations of the same job with the same machine (i.e., every pair such that $M_{j,\ell} = M_{j,r}$) has $|\ell - r| \geq w$.

THEOREM 1.3. *There is a randomized polynomial-time algorithm for job-shop scheduling that, with high probability, delivers a schedule of makespan* $O((P_{\max} + \Pi_{\max}) \cdot \rho)$, *where*

(a) *if every job needs at most u time units on each machine, then*

$$\rho = \frac{\log u}{\log\log u} \cdot \left\lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log u} \right\rceil;$$

(b) *if the job-shop instance is w-separated and* $p_{\max} = 1$, *then*

$$\rho = 1 \quad \text{if } w \geq \log(P_{\max} + \Pi_{\max})/2;$$

$$\rho = \frac{\log(P_{\max} + \Pi_{\max})}{w \log(\log(P_{\max} + \Pi_{\max})/w)} \quad \text{otherwise.}$$

Part (a) of Theorem 1.3 shows quantitatively the advantages of having multiple copies of each machine; in such a case, we can try to spread out the operations of a job somewhat equitably to the various copies. Part (b) of Theorem 1.3 shows that if we have some (limited) flexibility in rearranging the operation sequence of a job, then it may pay to spread out multiple usages of the same machine.

1.3. Main contributions. Most of our results rely on probabilistic ideas; in particular, we exploit a "random delays" technique due to [11]. We make four contributions, which we first sketch in general terms. The rough idea behind the "random delays" technique is as follows. We give each job a delay chosen randomly from a suitable range and independently of the other jobs, and we imagine each job waiting out this delay and then running without interruption; next we argue that, with high probability, not too many jobs contend for any given machine at the same time [11, 21]. We then resolve contentions by "expanding" the above "schedule"; the "low contention" property is invoked to argue that a small amount of such expansion suffices. The approach of [21] to this "expansion" problem is as follows. First, they present an upper bound on the maximum amount of contention on any machine at any step, which is shown to hold with high probability. Suppose we are given such a schedule, in which

at most s operations contend for any machine at any time. If all operations are of the same length, this can be converted into a valid schedule by an s-fold expansion of each time step. However, the operation lengths may be disparate. But we may round all operation lengths up to the nearest power of 2; thus, there will be only $O(\log p_{\max})$ operation lengths. The approach of [21] is then to carefully decompose the schedule into certain intervals such that within each interval, all operation lengths are the same. These, along with some other ideas, constitute the "expansion" approach of [21].

Our first contribution is a better combinatorial solution to the above expansion problem, which leads to a smaller expansion than that of [21]. In particular, we do not handle different operation lengths separately, but we show a way of combining them. The second contribution shows that a relaxed notion of "low contention" suffices: we do not require that the contention on machines be low at each time step. The first contribution helps to prove Theorem 1.2(a); parts (b) and (c) of Theorem 1.2 make use of the second contribution. We derandomize the sequential formulations using a technique of [2] and then parallelize. A simple but crucial ingredient of Theorem 1.2 is a new way to structure the operations of jobs in an initial (infeasible) schedule; we call this *well-structuredness* and present it in section 2. This notion is our third contribution. Finally, Theorem 1.3 comes about by introducing random delays and by using the Lovász local lemma (LLL) [7]. Although this is also done in [11], our improvements arise from a study of the correlations involved and by using Theorem 1.2(a). This study of correlations is our fourth contribution. The rest of this paper is organized as follows. Section 2 sets up some preliminary notions, section 3 presents the proof of Theorem 1.2, and Theorem 1.3 is proved in section 4.

2. Preliminaries. For any nonnegative integer k, we let $[k]$ denote the set $\{1, 2, \ldots, k\}$. The base of the natural logarithm is denoted by e as usual and, for convenience, we may use $\exp(x)$ to denote e^x.

As in [21], we assume throughout that all operation lengths are powers of 2. This can be achieved by multiplying each operation length by at most 2. This assumption on operation lengths will only affect our approximation factor and running time by a constant factor. Thus, $P_{\max}$, $\Pi_{\max}$, and $p_{\max}$ should be replaced by some $P'_{\max} \leq 2P_{\max}$, $\Pi'_{\max} \leq 2\Pi_{\max}$, and $p'_{\max} \leq 2p_{\max}$, respectively, in what follows. To retain simplicity we have avoided using such new notation.

2.1. Reductions. It is shown in [21] that, in deterministic polynomial time, we can reduce the general shop-scheduling problem to the case (i) where $p_{\max} \leq n\mu$, and (ii) where $n \leq \text{poly}(m, \mu)$, while incurring an *additive* $O(P_{\max} + \Pi_{\max})$ term in the makespan of the schedule produced. The reduction (i) also works in NC. (Of the two reductions, (ii) is more involved; it uses, e.g., an algorithm due to [20].)

Thus, for our sequential algorithms we assume that $n \leq \text{poly}(m, \mu)$ and that $p_{\max} \leq \text{poly}(m, \mu)$; while for our NC algorithms we assume only that $p_{\max} \leq n\mu$.

2.2. Bounds. We use the following Chernoff–Hoeffding bounds on the expectation and tails of distributions (see [15]).

FACT 2.1. *Let $X_1, X_2, \ldots, X_\ell \in [0, 1]$ be* independent *random variables with $X \doteq \sum_i X_i$. Then for any $\delta > 0$, $\mathrm{E}[(1+\delta)^X] \leq e^{\delta \mathrm{E}[X]}$.*

We define $G(\mu, \delta) \doteq (e^\delta/(1+\delta)^{1+\delta})^\mu$. Using Markov's inequality and Fact 2.1, we obtain the following bound on the tail of the binomial distribution.

FACT 2.2. *Let $X_1, X_2, \ldots, X_\ell \in [0, 1]$ be* independent *random variables with $X \doteq \sum_i X_i$ and $\mathrm{E}[X] = \mu$. Then for any $\delta > 0$, $\Pr[X \geq \mu(1+\delta)] \leq G(\mu, \delta)$.*

2.3. Random delays. Our algorithms use *random initial delays* which were developed in [11] and used in [21]. A *B-delayed schedule* of a job-shop instance is constructed as follows. Each job J_j is assigned a delay d_j in $\{0, 1, \ldots, B-1\}$. In the resulting B-delayed schedule, the operations of J_j are scheduled consecutively, starting at time d_j. A *random B-delayed schedule* is a B-delayed schedule in which the delays have been chosen independently and uniformly at random from $\{0, 1, \ldots, B-1\}$. Our algorithms schedule a job-shop instance by choosing a random B-delayed schedule for some suitable B and then expanding this schedule to resolve conflicts between operations that use the same machine at the same time.

For a B-delayed schedule $\mathcal{S}$, the *contention*, $C(M_i, t)$, is the number of operations scheduled on machine M_i in the time interval $[t, t+1)$. (Recall that operation lengths are integral.) For any job J_j, define the random variable $X_{i,j,t}$ to be 1 if some operation of J_j is scheduled on M_i in the time interval $[t, t+1)$ by $\mathcal{S}$, and 0 otherwise. Since no two operations of J_j contend for M_i simultaneously, $C(M_i, t) = \sum_j X_{i,j,t}$. If the delays are chosen uniformly at random and $B \geq \Pi_{\max}$, then $\mathrm{E}[X_{i,j,t}]$ is at most the total processing time of J_j on M_i divided by $\Pi_{\max}$. Thus, $\mathrm{E}[C(M_i, t)] = \sum_j \mathrm{E}[X_{i,j,t}] \leq \Pi_{\max}/\Pi_{\max} = 1$. We also note that the random variables $\{X_{i,j,t} \mid j \in [n]\}$ are mutually independent for any given i and t. We record all of this as follows.

FACT 2.3. *If $B \geq \Pi_{\max}$ and $\mathcal{S}$ is a random B-delayed schedule, then for any machine M_i and any time t, $C(M_i, t) = \sum_j X_{i,j,t}$, where the* 0-1 *random variables $\{X_{i,j,t} \mid j \in [n]\}$ are mutually independent. Also, $\mathrm{E}[C(M_i, t)] \leq 1$.*

2.4. Well-structuredness. Recall that all operation lengths are assumed to be powers of 2. We say that a delayed schedule $\mathcal{S}$ is *well-structured* if for each k, all operations with length 2^k begin in $\mathcal{S}$ at a time instant that is an integral multiple of 2^k. We shall use the following simple way of constructing such schedules from randomly delayed schedules. First, create a new job-shop instance by replacing each operation $(M_{j,\ell}, t_{j,\ell})$ by the operation $(M_{j,\ell}, 2 \cdot t_{j,\ell})$. Suppose $\mathcal{S}$ is a random B-delayed schedule for this modified instance for some B; we will call $\mathcal{S}$ a *padded random B-delayed schedule*. From $\mathcal{S}$, we can construct a well-structured delayed schedule, $\mathcal{S}'$, for the original job-shop instance: simply insert $(M_{j,l}, t_{j,l})$ with the correct boundary in the slot assigned to $(M_{j,l}, 2 \cdot t_{j,l})$ by $\mathcal{S}$. S' will be called a *well-structured random B-delayed schedule* for the original job-shop instance.

3. Proof of Theorem 1.2. In this section we prove Theorem 1.2. In section 3.1 we give a randomized polynomial-time algorithm that proves part (b) of the theorem. In section 3.2 we improve the algorithm to prove part (a). Finally we discuss the derandomization and parallelization of these algorithms in section 3.3. Throughout, we shall assume upper bounds on n and $p_{\max}$ as described in section 2.1; this explains terms such as $\log(\min\{m\mu, p_{\max}\})$ in the bounds of Theorem 1.2. Given a delayed schedule $\mathcal{S}$, define $C(t) \doteq \max_i \ C(M_i, t)$.

LEMMA 3.1. *There is a randomized polynomial-time algorithm that takes a job-shop instance and produces a* well-structured *delayed schedule which has a makespan $L \leq 2(P_{\max} + \Pi_{\max})$. With high probability, this schedule satisfies*

(a) $\forall i \in [m]\ \forall t \in \{0, 1, \ldots, L-1\},\ C(M_i, t) \leq \alpha$, *and*

(b) $\sum_{t=0}^{L-1} C(t) \leq \beta(P_{\max} + \Pi_{\max})$,

where $\alpha = c_1 \log(m\mu)/\log\log(m\mu)$ and $\beta = c_2 \log m/\log\log m$, for sufficiently large constants $c_1, c_2 > 0$.

Proof. Recall that all operation lengths are assumed to be powers of 2. Let $B = 2\Pi_{\max}$, and let $\mathcal{S}$ be a *padded* random B-delayed schedule of the new instance,

as described in section 2.4. $\mathcal{S}$ has a makespan of at most $2(P_{\max} + \Pi_{\max})$. Let $\mathcal{S}'$ be the well-structured random B-delayed schedule for the original instance that can be constructed from $\mathcal{S}$, as described in section 2.4. The contention on any machine at any time under $\mathcal{S}'$ is clearly no more than under $\mathcal{S}$. Thus in order to show that $\mathcal{S}'$ satisfies (a) and (b) with high probability, it suffices to show that $\mathcal{S}$ has this property. We will prove this now.

Part (a). The following proof is based on that of [21]. Fix any positive integer k and any M_i. For any set $U = \{u_1, u_2, \ldots, u_k\}$ of k units of processing that need to be done on M_i, let Collide(U) be the event that in S all these k units get scheduled at the same unit of time on M_i. It is not hard to see that $\Pr[\text{Collide}(U)] \leq (1/B)^{k-1}$. (If $u_1, \ldots, u_k$ are from different jobs, then $\Pr[\text{Collide}(U)] \leq (1/B)^{k-1}$. Otherwise, $\Pr[\text{Collide}(U)] = 0$.) Recall that $B = 2\Pi_{\max}$. Since there are at most $\binom{2\Pi_{\max}}{k}$ ways of choosing U, we get

$$\Pr[\exists t : \ C(M_i, t) \geq k] = \Pr[\exists U : \ \text{Collide}(U)] \leq \binom{2\Pi_{\max}}{k} (1/(2\Pi_{\max}))^{k-1},$$

and so $\Pr[\exists t : \ C(M_i, t) \geq k] \leq 2\Pi_{\max}/k!$. Thus,

$$\Pr[\exists t \ \exists i : \ C(M_i, t) \geq k] \leq 2m\Pi_{\max}/k!.$$

But $\Pi_{\max} \leq n\mu p_{\max}$, which by our assumptions in section 2.1 is poly(m, μ). Since $\lceil \alpha \rceil ! > (m\mu)^{c_1/2}$ for sufficiently large m or μ, we can satisfy (a) with high probability if we choose c_1 sufficiently large.

Part (b). Let $\gamma = \beta\epsilon/2$, where ϵ is the desired constant in the probability bound. Let the constant c_2 in the definition of β be sufficiently large so that $\gamma > 1$. Fix any M_i and t, and let $\lambda = \mathrm{E}[C(M_i, t)]$. (By Fact 2.3, $\lambda \leq 1$.) By Fact 2.1, with $1 + \delta = \gamma$,

$$\mathrm{E}[\gamma^{C(M_i,t)}] \leq e^{(\gamma-1)\lambda} \leq e^{(\gamma-1)}.$$

Hence, for any given t,

$$\begin{aligned} \mathrm{E}[\gamma^{C(t)}] = E[\gamma^{\max_{i\in[m]} C(M_i,t)}] &\leq E\left[\sum_{i\in[m]} \gamma^{C(M_i,t)}\right] = \sum_{i\in[m]} \mathrm{E}[\gamma^{C(M_i,t)}] \\ &\leq me^{\gamma-1}. \end{aligned} \tag{3.1}$$

Since the function $x \mapsto \gamma^x$ is convex, by Jensen's inequality we get that $\mathrm{E}[\gamma^{C(t)}] \geq \gamma^{\mathrm{E}[C(t)]}$. If we choose c_2 sufficiently large, then $\gamma^\gamma \geq me^{\gamma-1}$. Combining these observations with bound (3.1), we get $\mathrm{E}[C(t)] \leq \gamma$. By linearity of expectation, $\mathrm{E}[\sum_t C(t)] \leq 2\gamma(P_{\max} + \Pi_{\max})$ and finally, by Markov's inequality, we have

$$\Pr\left[\sum_t C(t) > \beta(P_{\max} + \Pi_{\max})\right] \leq 2\gamma/\beta = \epsilon. \qquad \square$$

3.1. Proof of Theorem 1.2(b). Recall that our goal is a polynomial-time algorithm which delivers a schedule with makespan $\mathrm{O}((P_{\max} + \Pi_{\max}) \cdot \frac{\log m}{\log\log m} \cdot \log(\min\{m\mu, p_{\max}\}))$. Assume $\mathcal{S}$ is a delayed schedule satisfying the conditions of Lemma 3.1 with makespan $L = \mathrm{O}(P_{\max} + \Pi_{\max})$. We begin by partitioning the schedule into *frames*, i.e., time intervals $\{[ip_{\max}, (i+1)p_{\max}), i = 0, 1, \ldots, \lceil L/p_{\max} \rceil - 1\}$.

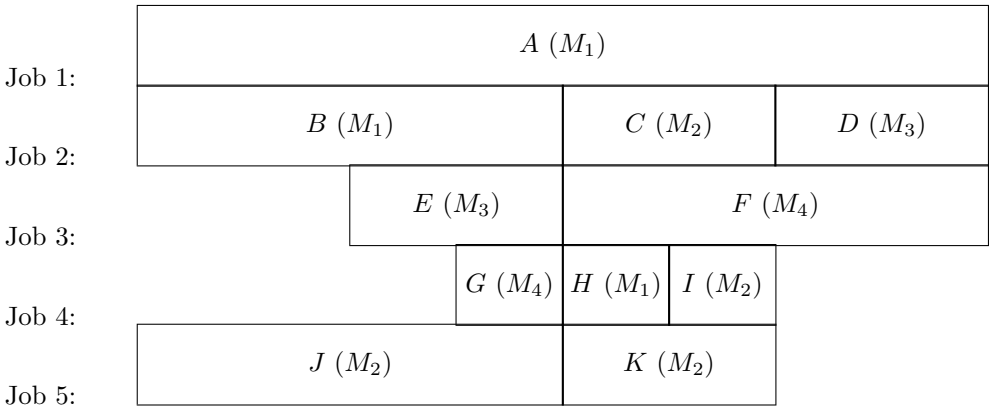


FIG. 1. *One frame of S, where $p_{\max} = 8$, A–K are the labels of operations, and M_1–M_4 are the machines.*

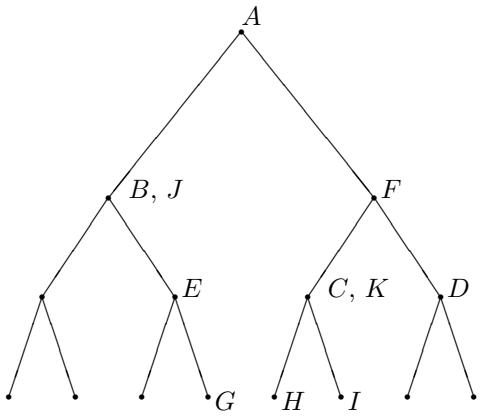


FIG. 2. *Assigning operations to nodes of T. For example, if u denotes the leftmost node on the second-highest level, then $S_1(u) = \{B\}$, $S_5(u) = \{J\}$, and $S_\ell(u) = \emptyset$ for every other ℓ.*

By the definition of $p_{\max}$ and the fact that $\mathcal{S}$ is well structured, no operation straddles a frame. For example, see Figure 1.

We construct a feasible schedule for the operations performed under schedule $\mathcal{S}$ for each frame. Concatenating these schedules yields a feasible schedule for the original problem. We give the frame-scheduling algorithm where, without loss of generality, we assume that its input is the first frame.

Let T be a rooted complete binary tree with $p_{\max}$ leaves. For every node u of T, let $l(u)$ and $r(u)$ be the labels, respectively, of the leftmost and rightmost leaves of the subtree rooted at u. We shall associate the operations scheduled during the frame with the nodes of T in a natural way. For $i = 1, \ldots, m$ we define $S_i(u)$ to be those operations that are scheduled on M_i by $\mathcal{S}$ for precisely the time interval $[l(u), r(u) + 1)$; each operation scheduled by $\mathcal{S}$ in the first frame is in exactly one $S_i(u)$. For example, see Figure 2. Let $p(u) = (r(u) - l(u) + 1) \cdot \max_i ||S_i(u)||$, where $||S_i(u)||$ denotes the cardinality of $S_i(u)$. $p(u)$ is the amount of time needed to perform the operations associated with u. For example, see Figure 3. Let the nodes of T be numbered as $u_1, u_2, \ldots$ in the *preorder* traversal of T. Define $f(u_1) = 0$ and for $j \geq 2$, let $f(u_j) = \sum_{k<j} p(u_k)$. For example, see Figure 4. The algorithm simply

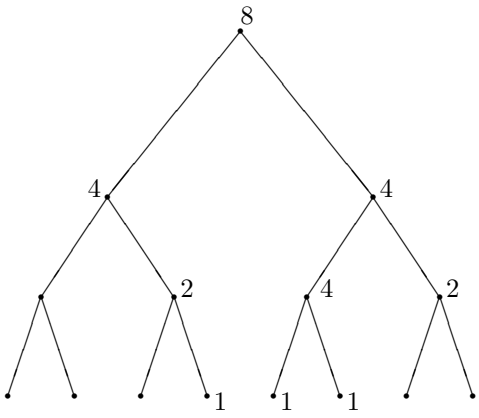


FIG. 3. *Calculating p for each node.*

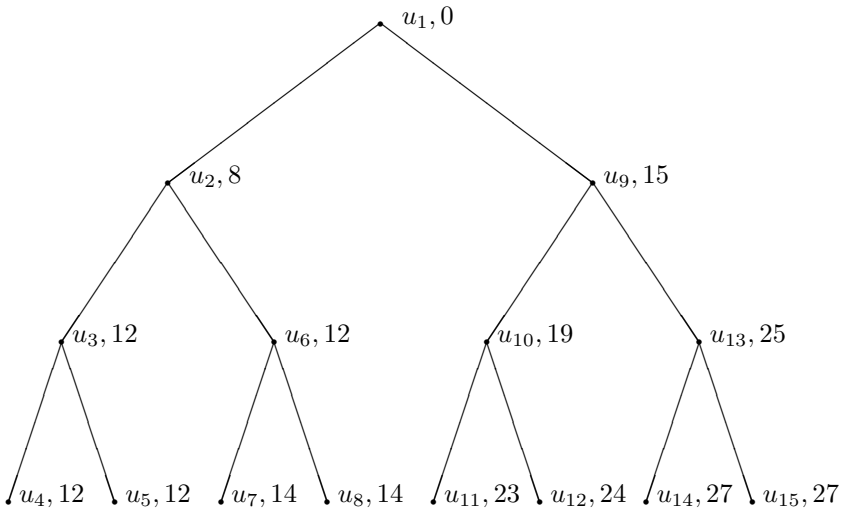


FIG. 4. *Calculating f for each node.*

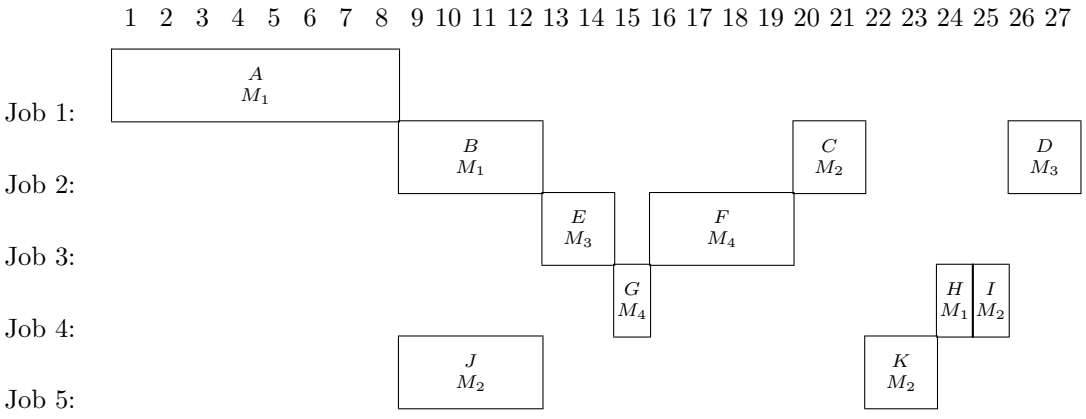


FIG. 5. *The schedule S'.*

schedules the operations in $S_i(u)$ on machine M_i consecutively beginning at time $f(u)+1$ and concluding by the end of timestep $f(u)+p(u)$. Let $\mathcal{S}'$ be the resulting schedule. For example, see Figure 5. Note that our algorithm does not necessarily give the same schedule as the algorithm of Shmoys, Stein, and Wein. For instance, our algorithm produces a different schedule than the one that their algorithm produces on the example given in [21]. Part (b) of Theorem 1.2 follows from both Lemma 3.1 and the following lemma.

LEMMA 3.2. *$\mathcal{S}'$ is feasible and has makespan at most $\sum_{u\in T} p(u)$, which is at most $(1+\log_2 p_{\max})\cdot\sum_{j=0}^{p_{\max}-1} C(j)$, where $C(t)$ is the maximum contention at time t under schedule S.*

Proof. By construction, no machine performs more than one operation at a time. Suppose O_1 and O_2 are distinct operations of job J scheduled in the first frame. Assume $O_1 \in S_i(u)$ and $O_2 \in S_j(v)$, where possibly $i=j$. Assume O_1 concludes before O_2 begins under $\mathcal{S}$; thus u and v are roots of disjoint subtrees of T and u precedes v in the preorder traversal of T. Thus, O_1 concludes before O_2 begins in $\mathcal{S}'$ and the new schedule is feasible.

Clearly the makespan of $\mathcal{S}'$ is at most $\sum_{u\in T} p(u)$. Fix a node u at some height k in T. (We take leaves to have height 0.) Then $p(u) = 2^k \max_i ||S_i(u)||$. Since the maximum number of jobs scheduled at any time t on any machine under $\mathcal{S}$ is $C(t)$, we get that $\forall t \in [l(u),\ldots,r(u)]$, $\max_i ||S_i(u)|| \le C(t)$. Thus,

$$p(u) \le 2^k \max_i ||S_i(u)|| \le \sum_{t\in[l(u),\ldots,r(u)]} C(t).$$

Since each leaf of T has $(1+\log_2 p_{\max})$ ancestors, the makespan of $\mathcal{S}'$ is at most

$$\sum_{u\in T} p(u) \le \sum_{u\in T}\sum_{t\in[l(u),\ldots,r(u)]} C(t) = (1+\log_2 p_{\max})\cdot\sum_{t=0}^{p_{\max}-1} C(t). \qquad \square$$

3.2. Proof of Theorem 1.2(a). Recall that our goal is a polynomial-time algorithm which delivers a schedule with makespan $O((P_{\max}+\Pi_{\max})\cdot\frac{\log(m\mu)}{\log\log(m\mu)}\cdot\lceil\frac{\log(\min\{m\mu,p_{\max}\})}{\log\log(m\mu)}\rceil)$. We give a slightly different frame-scheduling algorithm and show that the feasible schedule for each frame has makespan $O(p_{\max}\alpha\lceil\log(p_{\max})/\log\alpha\rceil)$, where $\alpha = c_1\log(m\mu)/\log\log(m\mu)$ as in Lemma 3.1. Without loss of generality, we assume that α is a power of 2 (by increasing it if necessary). Thus, under the assumptions from section 2.1, the final schedule satisfies the bounds of Theorem 1.2(a).

The difficulty with the algorithm given in section 3.1 is that the operations may be badly distributed to the nodes of T by $\mathcal{S}$, which would make $\mathcal{S}'$ inefficient. To clarify, consider the example given in Figures 1–5. In this case, node u_{10} is assigned operations C and K and node u_{11} is assigned operation H. The algorithm schedules operations C and K before operation H. However, since H is on a different machine from C and K, it could have been scheduled to overlap C or K. In this section, we show how to overcome this problem by "pushing down" operations C and K to nodes u_{11} and u_{12}.

The algorithm that we describe here starts with the allocation of operations to nodes of T that is defined in section 3.1. That is, $S_i(u)$ is taken to be the set of operations that are scheduled on M_i by S for time interval $[l(u), r(u)+1)$. The algorithm then chops T into disjoint subtrees in a manner described below. For each subtree, it redistributes the operations that are allocated to the nodes of the subtree

by "pushing" some operations from parents to children (in a manner which will be described shortly). After the redistribution, $R_i(u)$ is the set of machine-i operations that are allocated to node u. $p(u)$ is then taken to be the maximum over all i of the sum of the lengths of the operations in $R_i(u)$. The algorithm then finishes the algorithm of section 3.1: the p-values computed for each node are used to compute $f(v)$ (for every node v). Then the operations in $R_i(v)$ are scheduled beginning at time $f(v)+1$ and concluding by the end of timestep $f(v)+p(v)$.

The partitioning of T is done by removing all edges from parents with height equal to 0 modulo $\log \alpha$. (Thus, every resulting subtree T' has height at most $\log \alpha$.)

Let lg denote the logarithm to the base 2. (In some places below, we will not be able to use $\log x$ since, as defined by us, $\log x$ does not always equal the logarithm of x to the base 2. So we need lg.)

The redistribution of operations for subtree T' proceeds in a top-down manner, independently for each machine M_i. We will illustrate the process with the job-shop instance in Figure 6, where we assume (for descriptive purposes) that T has only one subtree T'. Start at the root, u_1, of T'. Suppose that u_1 has h operations allocated to it. (In this case, $h=3$.) Let $h' = 2^{\lceil \lg h \rceil}$ (in this case, $h'=4$) and allocate $h'-h$ dummy operations $\emptyset$ to T' as in Figure 7. (The reason for adding the dummy operations is to make the number of operations at the root equal to a power of 2.) If the height of the subtree rooted at u_1 (in this case, 2) is at least $\lg(h')$ (which is also 2 in this case), then the h' operations originally allocated to u_1 are reallocated to the h' nodes that are at distance $\lg(h')$ below u as in Figure 8. Next, the operations are further reallocated recursively in the subtrees below u_1. (In this case, the operations are recursively reallocated in the subtrees rooted at u_2 and u_5.) If, in one of these recursive calls, the height, k, of the subtree being considered is less than $\lg(h')$ (where h' is the number of *originally allocated* operations at the root, counting dummy operations), then $h'/2^k$ of the operations originally allocated to the root are reallocated to each of the leaves. For example, in the recursive call on the subtree rooted at u_2 in Figure 8, $h'=4$ (because a dummy operation is added to u_2 to make the number of operations a power of 2) and the height, k, of the subtree below u_2 is 1. Thus, $h'/2^1$ operations are pushed from u_2 to each of its children as in Figure 9. The recursive call at u_5 and the recursive calls at the leaves do not further redistribute operations.

A more formal description of the pushdown algorithm is as follows. As above, we assume that $||S_i(v)||$ is a power of 2 for all i and v; furthermore, although we will push some operations down the tree, $S_i(v)$ will refer throughout this paper to the *original* set of operations scheduled on M_i for the time interval $[l(v), r(v)+1)$. First, partition the tree T into disjoint subtrees, by removing all edges from parents with height equal to 0 modulo $\log \alpha$. We then proceed independently for each subtree T' that is produced from the partition and for each machine M_i, by calling a recursive procedure pushdown(T', i), which we describe now. Given a binary tree T'' with root u and a machine index i, pushdown(T'', i) is as follows. If T'' is a leaf, the procedure does nothing. Otherwise, suppose $||S_i(u)|| = h'$, with h' being a power of 2. If the height k of T'' is at least $\lg(h')$, then the h' operations of $S_i(u)$ are reallocated to the h' nodes that are at distance $\lg(h')$ below u; else if $k < \lg(h')$, then $h'/2^k$ of the operations in $S_i(u)$ are reallocated to each of the leaves of T''. Finally, we recursively call the procedure on the left and right subtrees of T''.

Note that if the new algorithm is applied to the problem instance from Figures 1–5, then the makespan is reduced by one, because operations C and K are pushed

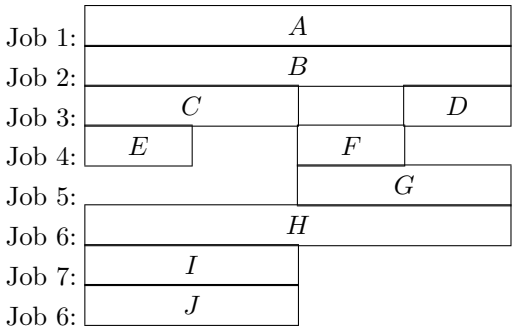


FIG. 6. *One frame of S, focusing* only *on operations for a single machine.*

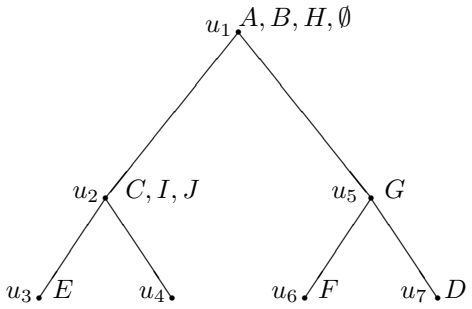


FIG. 7. *Assigning a dummy operation $\emptyset$ to the root of T'.*

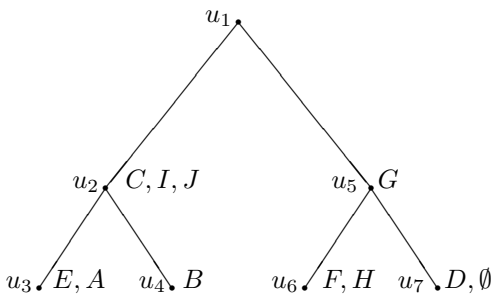


FIG. 8. *Redistributing the operations originally allocated to u_1.*

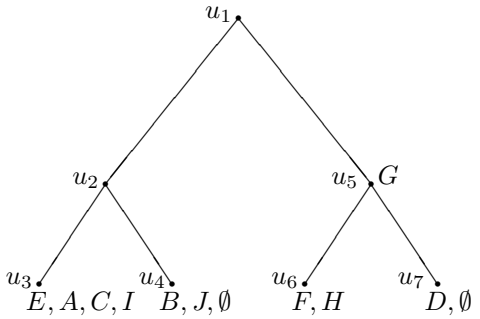


FIG. 9. *Redistributing the operations originally allocated to u_2.*

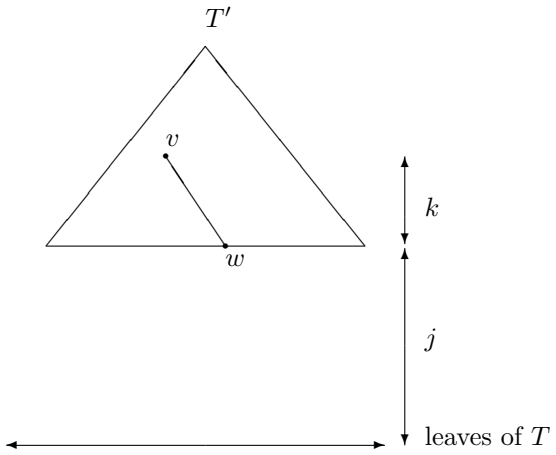


FIG. 10. *The case in which w is a leaf.*

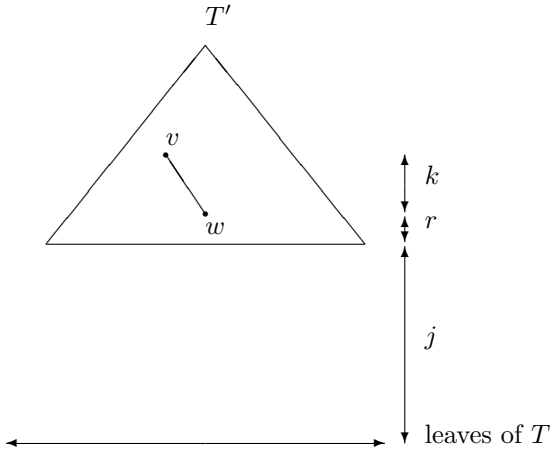


FIG. 11. *The case in which w is not a leaf.*

down to the leaves so operation H is scheduled to overlap operation C.

Let $\mathcal{S}'$ denote the schedule produced (from S) by the new algorithm.

LEMMA 3.3. *$\mathcal{S}'$ is a feasible schedule with makespan* $\mathrm{O}(p_{\max}\alpha\lceil\log p_{\max}/\log\alpha\rceil)$.

Proof. The proof that $\mathcal{S}'$ is feasible follows exactly as before. The makespan of $\mathcal{S}'$ is no more than $\sum_{u\in T} p(u)$.

Consider a subtree T' of the partition. Assume the leaves of T' are at height j in T. Let w be a node in T', and let V be the subset of nodes of T' consisting of w and its ancestors in T'.

First, suppose w is a leaf. Let v be a node in V and assume that v has height k in T' with $||S_i(v)|| = h$. (See Figure 10.)

Then v contributes at most $2^{\lceil\lg h\rceil}/2^k$ operations to $R_i(w)$ and each has length 2^{j+k}. The time needed to perform these operations is $2^{\lceil\lg h\rceil-k}\cdot 2^{j+k} = 2^{\lceil\lg h\rceil+j}$. By Lemma 3.1, part (a), $\sum_{v\in V}||S_i(v)|| \le 2\alpha$. (The factor of 2 arises from the (possible) padding of $S_i(v)$ with dummy operations.) Thus $p(w) \le 2^{j+1}\alpha$.

Now suppose w is at height $r > 0$ in T'. (See Figure 11.) A node $v \in V$ at height $r+k$ in T' contributes at most one operation to $R_i(w)$, and its length is 2^{j+k+r}. Thus

$p(w) \leq \sum_{k=0}^{\log \alpha - r} 2^{j+k+r} \leq 2^{j+1}\alpha$.

Thus, if node w is at height $r+j$ in T and is in the layer of the partition containing T', then $p(w) \leq 2^{j+1}\alpha$; also, there are $p_{\max}/2^{r+j}$ nodes at this height in T. The sum of these $p(w)$'s is thus at most $2\alpha p_{\max}/2^r$. Therefore each layer contributes at most $4\alpha p_{\max}$, and there are $\lceil (\log p_{\max})/(\log \alpha) \rceil$ layers. Thus, $\sum_{v \in T} p(v)$ satisfies the bound of the lemma. □

3.3. Derandomization and parallelization. Note that all portions of our algorithm are deterministic (and can be implemented in NC), except for the setting of the initial random delays, which we now show how to derandomize. The method of conditional probabilities could be applied to give the sequential derandomization; however, that result will follow from the NC algorithm that we present. We begin with a technical lemma.

LEMMA 3.4. *Let $x_1, x_2, \ldots, x_\ell$ be nonnegative integers such that $\sum_i x_i = \ell a$ for some $a \geq 1$. Let $k \leq a$ be any positive integer. Then, $\sum_{i=1}^{\ell} \binom{x_i}{k} \geq \ell \cdot \binom{a}{k}$.*

Proof. For real x, we define, as usual, $\binom{x}{k} \doteq (x(x-1)\cdots(x-k+1))/k!$. We first verify that the function $f(x) = \binom{x}{k}$ is nondecreasing and convex for $x \geq k$, by a simple check that the first and second derivatives of f are nonnegative for $x \geq k$. Think of minimizing $\sum_i \binom{x_i}{k}$ subject to the given constraints. If $x_i \leq (k-1)$ for some i, then there should be an index j such that $x_j \geq (k+1)$, since $\sum_i x_i \geq \ell k$. Thus, we can lessen the objective function by simultaneously setting $x_i := x_i + 1$ and $x_j := x_j - 1$. Hence we may assume that all the integers x_i are at least k. By the convexity of f for $x \geq k$, we see that the objective function is at least $\sum_{i=1}^{\ell} \binom{a}{k}$. □

Define, for $z = (z_1, z_2, \ldots, z_n) \in \Re^n$, a family of symmetric polynomials $S_j(z), j = 0, 1, \ldots, n$, where $S_0(z) \equiv 1$, and for $1 \leq j \leq n$, $S_j(z) \doteq \sum_{1 \leq i_1 < i_2 \cdots < i_j \leq n} z_{i_1} z_{i_2} \cdots z_{i_j}$. We recall one of the main results of [2]. (This is not explicitly presented in [2] but is an obvious corollary of the results of section 4 in [2].) In the statement of Proposition 3.5 below, the function G refers to the one introduced in section 2.2 of this paper. Namely, $G(\mu, \delta) = (e^\delta/(1+\delta)^{1+\delta})^\mu$.

PROPOSITION 3.5 (see [2]). *Suppose we are given m* independent *random variables $y_1, \ldots, y_m$, each of which takes values* uniformly *in $R = \{0, 1, \ldots, 2^b - 1\}$, where $b = \mathrm{O}(\log N)$; here N is a parameter that roughly stands for "input length," and $m = N^{\mathrm{O}(1)}$. Suppose we are also given, for each $j \in [m]$, a finite set of binary random variables $\{z_{jt} : t = 1, 2, \ldots\}$, where z_{jt} is 1 if and only if y_j lies in some fixed subset R_{jt} of R. Also given are r random variables*

$$U_i = \sum_{j=1}^{m} z_{j,f(i,j)}, \qquad i \in [r],$$

where f is some arbitrary given function. If $\mathrm{E}[U_i] < 1$ for each i, then given any positive integer k such that $k = \mathrm{O}(\log N)$, we can find, deterministically using $N^{\mathrm{O}(1)}$ processors and $\mathrm{O}(\log^{\mathrm{O}(1)} N)$ time on the EREW PRAM, a setting $y_1 := w_1, \ldots, y_m := w_m$ such that

$$\sum_{i \in [r]} S_k(z_{1,f(i,1)}, \ldots, z_{m,f(i,m)}) \leq rG(1, k-1)(1 + N^{-c})$$

for any desired constant $c > 0$. See [15] for a definition of the EREW PRAM.

In our setting, the random variables y_i are the initial random delays of the jobs. It is easy to verify that each random variable $C(M_i, t)$ is of the form of some U_j in

the notation of Proposition 3.5. By giving the initial random delays in the range $\{0, 1, \ldots, 2\Pi_{\max}\}$ instead of from $\{0, 1, \ldots, 2\Pi_{\max} - 1\}$, we can ensure the condition $\mathrm{E}[U_j] < 1$ of Proposition 3.5 ($\mathrm{E}[C(M_i, t)] \leq 2\Pi_{\max}/(2\Pi_{\max} + 1)$ now). Let α and β be as in Lemma 3.1, and note that both are logarithmically bounded in the length of the input, as required for the parameter k in Proposition 3.5. Let the random variables $X_{i,j,t}$ be as in Fact 2.3. From the proof of part (a) of Lemma 3.1, we see that $\sum_{i,t} G(1, \alpha - 1)$ is smaller than 1; thus, by Proposition 3.5, we can find a setting $\vec{w}$ for the initial delays in NC such that

$$\sum_{i,t} S_\alpha(X_{i,1,t}, X_{i,2,t}, \ldots, X_{i,n,t}) < 1. \tag{3.2}$$

If the congestion of some machine M_i at some t were at least α due to the above setting of the initial delays to $\vec{w}$, then the left-hand side of (3.2) would be at least 1, contradicting (3.2). Thus, we have an NC derandomization of Theorem 1.2(a).

As for Theorem 1.2(b), we can similarly find an NC assignment of initial delays $\vec{w}$ such that

$$\begin{aligned} \sum_{i,t} S_\beta(X_{i,1,t}, X_{i,2,t}, \ldots, X_{i,n,t}) &= \mathrm{O}((P_{\max} + \Pi_{\max}) m G(1, \beta - 1)) \\ &= \mathrm{O}((P_{\max} + \Pi_{\max})). \end{aligned} \tag{3.3}$$

Let $C(t)$ be the (deterministic) maximum contention at time t, due to this setting. Note that

$$\binom{C(t)}{\beta} \leq \sum_i S_\beta(X_{i,1,t}, X_{i,2,t}, \ldots, X_{i,n,t}).$$

Thus, by (3.3), we see that

$$\sum_t \binom{C(t)}{\beta} = \mathrm{O}((P_{\max} + \Pi_{\max})).$$

We invoke Lemma 3.4 to conclude that $\sum_t C(t) = \mathrm{O}((P_{\max} + \Pi_{\max})\beta)$; thus, we have an NC derandomization of Theorem 1.2(b).

We remark that the work of Mahajan, Ramos, and Subrahmanyam [13] could also be used to obtain an NC derandomization.

4. Proof of Theorem 1.3. We now set about to prove Theorem 1.3; we are very much motivated here by the framework of [5, 1, 12] and of section 6.1 of [17]. The new ideas we need are due to the two basic ways in which job-shop scheduling generalizes packet routing: both acyclicity and the "$p_{\max} = 1$" condition can be violated. Theorem 4.3 is used first (in the next subsection) and proved later; this is to help the reader get to some of the new ideas quickly. The algorithms are shown in sections 4.2 and 4.3.

4.1. Preliminary results. We start with a standard fact about the function G of section 2.2, where $G(\mu, \delta) = (e^\delta/(1 + \delta)^{1+\delta})^\mu$.

FACT 4.1. (a) *If $\delta \in [0, 1]$, then $e^\delta/(1 + \delta)^{(1+\delta)} \leq e^{-\delta^2/3}$.* (b) *If $0 < \mu_1 \leq \mu_2$, then for any $\delta \geq 0$, $G(\mu_1, \mu_2\delta/\mu_1) \leq G(\mu_2, \delta)$.*

Proof. (a) The proof follows from observing that the function $\delta \mapsto \ln(e^{-\delta^2/3}(1 + \delta)^{(1+\delta)}e^{-\delta})$ is 0 when $\delta = 0$, and that its derivative is $\ln(1 + \delta) - 2\delta/3$, which is nonnegative for $\delta \in [0, 1]$.

(b) We need to show that

$$(1+\delta)^{(1+\delta)\mu_2} \leq \left(1+\frac{\mu_2\delta}{\mu_1}\right)^{(1+\frac{\mu_2\delta}{\mu_1})\mu_1},$$

i.e., that $\Phi(v) \doteq (1+v\delta)\ln(1+v\delta) - v(1+\delta)\ln(1+\delta) \geq 0 \forall v \geq 1$. We have $\Phi(1) = 0$; $\Phi'(v) = \delta + \delta\ln(1+v\delta) - (1+\delta)\ln(1+\delta)$. For $v \geq 1$, $\Phi'(v) \geq \delta - \ln(1+\delta) \geq 0$. □

The next lemma follows from [18].

LEMMA 4.2 (see [18]). *Let $X_1, \ldots, X_\ell \in \{0,1\}$ be random variables such that, for any set $T \subseteq \{1,2,\ldots,\ell\}$, $\Pr[\bigwedge_{i\in T}(X_i = 1)] \leq \prod_{i\in T}\Pr[X_i = 1]$; informally, the X_i are "negatively correlated." Then if $X = \sum_i X_i$ with $\mathrm{E}[X] \leq \mu$, we have, for any $\delta \geq 0$, $\Pr[X \geq \mu(1+\delta)] \leq G(\mu,\delta)$.*

Suppose we are given a job-shop instance I. A *delayed schedule* $\mathcal{S}$ for I is any "schedule" in which each job J_j waits for some arbitrary nonnegative integral amount of time d_j and then gets processed continuously. (Thus, $\mathcal{S}$ is a delayed schedule if and only if there exists some nonnegative integer B such that $\mathcal{S}$ is a B-delayed schedule.) Suppose, for some nonnegative integer B', we choose integers $d'_1, d'_2, \ldots, d'_j$ uniformly at random and independently, from $\{0,1,\ldots,B'-1\}$. The (random) schedule obtained by giving an initial delay of d'_j (*in addition to* the d_j above) to each job J_j will be called a *random $(B',\mathcal{S})$-delayed schedule*. Note that this also will be a delayed schedule.

We require a few more definitions related to $\mathcal{S}$ as above. Suppose L denotes the makespan of $\mathcal{S}$. Then, given an integer ℓ, an *ℓ-interval* is any time interval of the form $[t, t+\ell)$, where t is an integer such that $0 \leq t \leq L-1$. We denote the interval $[t,t+\ell)$ by F_t. The *contention of machine M_i in interval F_k in the schedule $\mathcal{S}$*, denoted $C_{\mathcal{S},\ell}(i,k)$, is the total processing time on M_i within F_k, in the schedule $\mathcal{S}$. (Suppose, for instance, an operation O of length $\ell+2$ uses M_i and is scheduled to run on M_i in the interval $[t,t+\ell+2)$, in $\mathcal{S}$. Then, for example, O contributes a value of ℓ to $C_{\mathcal{S},\ell}(i,t)$ and a value of three to $C_{\mathcal{S},\ell}(i,t+\ell-1)$.)

Given any integers j_1, j_2 such that $1 \leq j_1 \leq j_2 \leq n$, we let $C'_{\mathcal{S},\ell}(i,k,j_1,j_2)$ denote the total processing time on machine M_i in the interval F_k in the schedule S that is imposed by jobs $J_{j_1}, J_{j_1+1}, \ldots, J_{j_2}$. (In particular, $C'_{\mathcal{S},\ell}(i,k,1,n) = C_{\mathcal{S},\ell}(i,k)$.)

Given a *delayed* schedule $\mathcal{S}$, we call $\mathcal{S}$ an (L,ℓ,C)-schedule if and only if

- the makespan of $\mathcal{S}$ is at most L, and
- for all machines M_i and all ℓ-intervals F_k, $C_{\mathcal{S},\ell}(i,k) \leq C$.

We emphasize that this notation will be employed only for delayed schedules.

We start with Theorem 4.3, which will be of much help in proving Theorem 1.3. Given an (L,ℓ,C_0)-schedule for a job-shop instance, Theorem 4.3 shows a sufficient condition under which we can efficiently construct an $(L+B,\ell',C_1)$-schedule for appropriate values of B,ℓ' and C_1. In most of our applications of the theorem, we will have (i) $\ell' \ll \ell$, (ii) $B \ll L$, and (iii) C_1 sufficiently small so that the new "relative congestion" C_1/ℓ' is not much more than the original relative congestion C_0/ℓ. Thus, by slightly increasing the makespan of the delayed schedule, we are able to bound the relative congestion in intervals of much smaller length. (Note from (i) that $\ell' \ll \ell$.) Appropriate repetitions of this idea, along with some other tools, will help us prove Theorem 1.3.

For convenience, we define $x^+ = \max(x,0)$.

THEOREM 4.3. *There is a sufficiently large constant $c_3 > 0$ such that the following holds. Suppose $\mathcal{S}$ is an (L,ℓ,C)-schedule for a given job-shop instance I for some*

L, ℓ, C. Let nonnegative integers $\ell' \le \ell$ and $B \le \ell - \ell' + 1$ be arbitrary; let $\mathcal{S}'$ be a random $(B, \mathcal{S})$-delayed schedule.

Suppose $\delta > 0$ is such that for all integers $i \in [m]$, $0 \le k \le L + B - 1$ and $1 \le j_1 \le j_2 \le n$,

$$\Pr\left[C'_{\mathcal{S}',\ell'}(i,k,j_1,j_2) \ge \frac{\ell'}{B} \cdot (C'_{\mathcal{S},\ell}(i,(k-B+1)^+,j_1,j_2) + C\delta)\right] \le (\max\{L,B,C\})^{-c_3}.$$

Then there is a Las Vegas algorithm to construct an $(L+B, \ell', \frac{C\ell'}{B} \cdot (1+3\delta))$-schedule for I; the expected running time of the algorithm is poly(m, L, ℓ, C). The proof of Theorem 4.3 will be presented in section 4.5. Using ideas from our earlier proofs, we obtain the following corollary.

COROLLARY 4.4. *For general job-shop scheduling, there is a polynomial-time Las Vegas algorithm to construct a schedule of makespan*

$$\mathrm{O}\left((P_{\max} + \Pi_{\max}) \cdot \frac{\log(P_{\max} + \Pi_{\max})}{\log\log(P_{\max} + \Pi_{\max})} \cdot \left\lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log(P_{\max} + \Pi_{\max})} \right\rceil\right).$$

Proof. Let I be a job-shop scheduling instance with associated values $P_{\max}$ and $\Pi_{\max}$; define $L = 2P_{\max}$ and $B = 2\Pi_{\max}$. Let I' be the modified instance formed by replacing each operation $(M_{j,k}, t_{j,k})$ by the operation $(M_{j,k}, 2 \cdot t_{j,k})$. We trivially have an (L, B, B)-schedule $\mathcal{S}$ for I'. Choose $\lambda = c' \log(P_{\max} + \Pi_{\max}) / \log\log(P_{\max} + \Pi_{\max})$; c' is a suitably large constant as specified below.

Let $\mathcal{S}'$ denote the random $(B, \mathcal{S})$-delayed schedule. From the proof of Lemma 3.1(a), we find that $\Pr[C'_{\mathcal{S}',1}(i,k,j_1,j_2) \ge \lambda] \le (\max\{L,B,C\})^{-c_3}$ will hold for all i, k, j_1, j_2, by making c' large. Thus, by setting $\ell' = 1$ in Theorem 4.3, we can efficiently find an $(L + B, 1, C_0)$-schedule for I', where $C_0 = \mathrm{O}(\lambda)$. So we can efficiently construct a well-structured schedule for I with makespan $\mathrm{O}(P_{\max} + \Pi_{\max})$ in which, for all machines M_i and time steps t, the number of operations scheduled on machine M_i in the time interval $[t, t+1)$ is at most $\mathrm{O}(\lambda)$. The corollary now follows from the proof of Theorem 1.2(a) by using this fact in place of Lemma 3.1. ∎

We now present Lemma 4.5, which shows a way of using Theorem 4.3. The notion of "w-separated" in part (b) of the lemma is as defined in section 1.2. Namely, every distinct pair of operations of the same job with the same machine has at least $w - 1$ operations between them.

LEMMA 4.5. (a) *Consider any job-shop instance I in which any job needs at most u units of processing on any machine. Suppose $\mathcal{S}$ is some (L, ℓ, C)-schedule for I. For nonnegative integers $\ell' \le \ell$ and $B \le \ell - \ell' + 1$, suppose $\mathcal{S}'$ denotes the random $(B, \mathcal{S})$-delayed schedule. Then, for any $\delta > 0$ and all i, k, j_1, j_2,*

$$\Pr\left[C'_{\mathcal{S}',\ell'}(i,k,j_1,j_2) \ge \frac{\ell'}{B} \cdot (C'_{\mathcal{S},\ell}(i,(k-B+1)^+,j_1,j_2) + C\delta)\right] \le G(C\ell'/(Bu), \delta).$$

(b) *Suppose I is a w-separated job-shop instance with $p_{\max} = 1$ and that $\mathcal{S}$ denotes the (unique) 0-delayed schedule for I. Let $\mathcal{S}'$ denote the random $\Pi_{\max}$-delayed schedule for I. Then, for any $\delta > 0$ and $\forall i, k, j_1, j_2$,*

$$\Pr\left[C'_{\mathcal{S}',w}(i,k,j_1,j_2) \ge \frac{w}{\Pi_{\max}} \cdot (C'_{\mathcal{S},\Pi_{\max}+w-1}(i,(k-\Pi_{\max}+1)^+,j_1,j_2) + \Pi_{\max}\delta)\right] \le G(w,\delta).$$

Proof. To have some common notation for parts (a) and (b), we define the following quantities for part (b). First, in part (b), S is a $(P_{\max}, \ell, C)$-schedule, where, e.g., $\ell = \Pi_{\max} + w - 1$ and $C = \Pi_{\max}$. Also, in part (b), $\mathcal{S}'$ is the random $(B, \mathcal{S})$-delayed schedule, where $B = \Pi_{\max}$; we also set $\ell' = w$ in part (b). Note that the conditions $\ell' \le \ell$ and $B \le \ell - \ell' + 1$ now hold for part (b) also.

We now make some observations common to parts (a) and (b). Fix i, k, j_1, j_2. Since all new delays introduced by $\mathcal{S}'$ lie in $\{0, 1, \ldots, B-1\}$, the only units of processing that can get scheduled on M_i in the interval $[k, k+\ell')$ in $\mathcal{S}'$ are those that were scheduled on M_i in the interval $\mathcal{I} \doteq [(k - B + 1)^+, k + \ell')$ in $\mathcal{S}$. Note that the length of $\mathcal{I}$ is at most $\ell' + B - 1 \le \ell$. For each job J_j, number its single units of processing scheduled on M_i in $\mathcal{I}$ in $\mathcal{S}$, as $U_{j,1}, U_{j,2}, \ldots$. Since the length of $\mathcal{I}$ is at most ℓ, the definition of C' shows that the number of such units for each job J_j is at most $C'_{\mathcal{S},\ell}(i, (k - B + 1)^+, j, j)$.

Let $X_{j,t}$ be the indicator random variable for $U_{j,t}$ getting scheduled in the interval $[k, k + \ell')$ in $\mathcal{S}'$. We have

$$C'_{\mathcal{S}',\ell'}(i, k, j_1, j_2) \le \sum_{j=j_1}^{j_2} \sum_{t} X_{j,t}. \tag{4.1}$$

Since $\mathrm{E}[X_{j,t}] \le \ell'/B$ for each j, t, we have

$$\begin{aligned} \nu \doteq \mathrm{E}[C'_{\mathcal{S}',\ell'}(i, k, j_1, j_2)] &\le \sum_{j=j_1}^{j_2} (C'_{\mathcal{S},\ell}(i, (k - B + 1)^+, j, j)\ell'/B) \\ &= C'_{\mathcal{S},\ell}(i, (k - B + 1)^+, j_1, j_2)\ell'/B. \end{aligned} \tag{4.2}$$

We now handle part (a). By (4.1), $C'' \doteq C'_{\mathcal{S}',\ell'}(i, k, j_1, j_2)/u$ is at most $\sum_{j=j_1}^{j_2} Y_j$, where $Y_j \doteq u^{-1} \sum_t X_{j,t}$. By the definition of u, $Y_j \le 1$ for each j. So the random variables $\{Y_j : j \in [j_1, j_2]\}$ lie in $[0, 1]$ and are independent. A Chernoff–Hoeffding bound shows for any $\delta' \ge 0$ that

$$\Pr[C'_{\mathcal{S}',\ell'}(i, k, j_1, j_2) \ge \nu(1 + \delta')] = \Pr[C'' \ge (\nu/u) \cdot (1 + \delta')] \le G(\nu/u, \delta'). \tag{4.3}$$

Next, $C'_{\mathcal{S},\ell}(i, (k - B + 1)^+, j_1, j_2) \le C$, since $\mathcal{S}$ is an (L, ℓ, C)-schedule. So, applying (4.2) and Fact 4.1(b) to (4.3) completes the proof for part (a).

For part (b), consider any job J_j. Since $\ell' = w$ here, the definition of w-separated shows that we cannot have $X_{j,t} = X_{j,t'} = 1$, if $t \ne t'$. This easily leads us to see that the random variables $\{X_{j,t} : j, t\}$ are negatively correlated, in the sense of Lemma 4.2. So, an application of Lemma 4.2 and Fact 4.1(b) to (4.1) and (4.2) completes the proof for part (b). □

Next we will use these results to prove Theorem 1.3 in sections 4.2 and 4.3.

4.2. Proof of Theorem 1.3(b). Recall that we are considering any w-separated job-shop instance I with $p_{\max} = 1$ now. Let $\mathcal{S}$ be the 0-delayed schedule for I. Thus, $\mathcal{S}$ is a $(P_{\max}, \ell, C)$-schedule, where, e.g., $\ell = \Pi_{\max} + w - 1$ and $C = \Pi_{\max}$. Also let $\mathcal{S}'$ be the random $(B, \mathcal{S})$-delayed schedule, where $B = \Pi_{\max}$; i.e., $\mathcal{S}'$ is the random B-delayed schedule for I. Define $\ell' = w$.

We can ensure that $G(w, \delta) \le (P_{\max} + \Pi_{\max})^{-c_3}$, by choosing (i) $\delta = c''$ if $w \ge \log(P_{\max} + \Pi_{\max})/2$, and (ii) $\delta = c'' \log(P_{\max} + \Pi_{\max})/(w \log(\log(P_{\max} + \Pi_{\max})/w))$ if $w < \log(P_{\max} + \Pi_{\max})/2$ for some suitably large constant c''. By Lemma 4.5(b) and

Theorem 4.3, we can then efficiently construct a $(P_{\max} + \Pi_{\max}, w, w(1+3\delta))$-schedule $\mathcal{S}''$ for I. We partition $\mathcal{S}''$ into $\lceil (P_{\max} + \Pi_{\max})/w \rceil$ intervals each of length w; crucially, each of these intervals (subproblems) is an *acyclic* job-shop instance. Also, in each of these subproblems, $p_{\max} = 1$, and any machine has at most $w(1 + 3\delta)$ operations to be scheduled on it. Via the result of [12], each subproblem can be efficiently scheduled with makespan $O(w + w(1 + \delta)) = O(w(1 + \delta))$. We then concatenate all these schedules, leading to a final makespan of $O((P_{\max} + \Pi_{\max})(1 + \delta))$.

4.3. Proof of Theorem 1.3(a). Recall that our goal is to show the existence of a schedule with makespan $O((P_{\max}+\Pi_{\max}) \cdot \frac{\log u}{\log\log u} \cdot \lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log u} \rceil)$, assuming that every job needs at most u time units on each machine. We assume that $u \geq 2$. Indeed, if $u = 1$, then we have an acyclic job-shop instance with $p_{\max} = 1$; so we will be able to efficiently construct a schedule of length $O(P_{\max} + \Pi_{\max})$ [11, 12]. The algorithm is presented in section 4.3.2; we start with a useful tool.

4.3.1. L'-splitting. Suppose we are given an (L, ℓ, C)-schedule $\mathcal{S}$ for a job-shop instance I and want to split it into subproblems each of makespan at most L', where $p_{\max} < L' < L$. If $p_{\max} = 1$, this is easy, as seen in section 4.2. Consider the case where $p_{\max}$ is arbitrary. We now show a simple way of partitioning the operations of $\mathcal{S}$ into at most $\lceil L/(L' - p_{\max}) \rceil$ subproblems $\mathcal{P}_1, \mathcal{P}_2, \ldots$. We will also output an (L', ℓ, C)-schedule $\mathcal{S}_i$ for each $\mathcal{P}_i$. These subproblems will be such that they can be solved independently and the resulting schedules can be concatenated to give a feasible schedule for I. This "L'-splitting" process is as follows.

We consider all operations that are completely finished by time L' in $\mathcal{S}$; scheduling this set of operations becomes our first subproblem $\mathcal{P}_1$. $\mathcal{S}$ provides a natural (L', ℓ, C)-schedule $\mathcal{S}_1$ for $\mathcal{P}_1$. If we have covered all operations by this process, we stop; if not, we define the next subproblem $\mathcal{P}_2$ as follows. Define $t_1 = 0$. Let t_2 be the smallest integer such that (i) $t_2 \leq L'$ and (ii) there is some operation O starting at time t_2 in $\mathcal{S}$ such that O is not completely finished by time L'. (Note that $L' - p_{\max} < t_2 \leq L'$.) Our second subproblem $\mathcal{P}_2$ consists of all operations (a) finishing by time $t_2 + L'$ in $\mathcal{S}$ and (b) not covered by $\mathcal{P}_1$. The time interval $[t_2, t_2 + L')$ in $\mathcal{S}$ provides an (L', ℓ, C)-schedule $\mathcal{S}_2$ for $\mathcal{P}_2$ in the obvious way. Once again, if we have not covered all operations, we define t_3 to be the smallest integer such that (i) $t_3 \leq t_2 + L'$ and (ii) there is some operation O starting at time t_3 in $\mathcal{S}$ such that O is not completely finished by time $t_2 + L'$. We have $t_3 > t_2 + L' - p_{\max}$; thus $t_3 > 2(L' - p_{\max})$. $\mathcal{P}_3$ consists of all operations finishing by time $t_3 + L'$ that were not covered by $\mathcal{P}_1$ and $\mathcal{P}_2$. We iterate this until all operations are covered.

In general, we have $t_{i+1} \geq i(L' - p_{\max})$; so the total number of subproblems created is at most $\lceil L/(L' - p_{\max}) \rceil$. It is also easy to see that we have an (L', ℓ, C)-schedule $\mathcal{S}_i$ for each $\mathcal{P}_i$. Also, the subproblems can be solved independently and the resulting schedules can be concatenated to give a feasible schedule for I.

4.3.2. Algorithm and analysis. We choose a sufficiently large positive constant b_0. Define $L_0 = P_{\max} + \Pi_{\max}$, and $L_i = \log L_{i-1}$ for $i \geq 1$. We repeat this iteration until we arrive at a t for which either $L_{t+1} \geq L_t$ or $L_{t+1} \leq 36b_0^2$. (Thus, the iteration proceeds for $O(\log^*(P_{\max} + \Pi_{\max}))$ steps.) Also, for $1 \leq i \leq t$, define

$$C_i \doteq L_i^3 \left(1 + \frac{b_0}{\sqrt{L_1}}\right) \prod_{j=1}^{i-1} \left(\left(1 + \frac{b_0}{\sqrt{L_{j+1}}}\right) \cdot \frac{1}{1 - (L_{j+1}/L_j)^3} \right). \tag{4.4}$$

Recall that $L_i \geq 36b_0^2$ for $1 \leq i \leq t$. If b_0 is large enough, we have

$$C_i \leq L_i^3 \exp\left(\left(\sum_{j=1}^{i} \frac{b_0}{\sqrt{L_j}}\right) + \mathrm{O}\left(\sum_{j=1}^{i-1}\left(\frac{L_{j+1}}{L_j}\right)^3\right)\right) \leq L_i^3 \exp(3b_0/\sqrt{L_i}) \leq 2L_i^3. \tag{4.5}$$

The second inequality follows from the fact that the terms L_j^{-1} increase exponentially with $L_j^{-1} \leq (36b_0^2)^{-1}$ and b_0 sufficiently large.

The algorithm is as follows. First, if $u^2 > P_{\max} + \Pi_{\max}$, then Corollary 4.4 shows that we can construct a schedule of makespan as claimed by Theorem 1.3(a). So suppose $u^2 \leq P_{\max} + \Pi_{\max}$. The algorithm consists of a preprocessing step and a general (recursive) step, motivated by the approach of section 6.1 of [17].

Preprocessing step. We start with the obvious $(P_{\max}, \ell, \Pi_{\max})$-schedule $\mathcal{S}$, where ℓ can be taken arbitrarily large.

First, we handle the case where $u \geq b_0 L_1$. We call this the "simple case." Define $\ell' = u^2$, $B = \Pi_{\max}$, and $\delta = 1$. If b_0 is large enough, then $G(u, \delta) \leq (P_{\max} + \Pi_{\max})^{-c_3}$. Thus, by Lemma 4.5(a) and Theorem 4.3, we can efficiently construct a $(P_{\max} + \Pi_{\max}, u^2, 4u^2)$-schedule $\mathcal{S}'$. We apply u^2-splitting to $\mathcal{S}'$, as defined in section 4.3.1. Since $u \geq 2$ and $p_{\max} \leq u$, the total number of subproblems is at most $\lceil (P_{\max} + \Pi_{\max})/(u^2 - p_{\max}) \rceil \leq \mathrm{O}((P_{\max} + \Pi_{\max})/u^2)$. Also, each of the subproblems has "$P_{\max}$" at most u^2 and "$\Pi_{\max}$" at most $4u^2$. So, by Corollary 4.4, each of these subproblems can be efficiently given a valid schedule of makespan $\mathrm{O}(u^2 \cdot \frac{\log u}{\log\log u} \cdot \lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log u} \rceil)$. As seen above, the number of subproblems is $\mathrm{O}((P_{\max} + \Pi_{\max})/u^2)$, so the concatenation of these schedules yields a final schedule of makespan as claimed by Theorem 1.3(a).

We now move on to the more interesting case where $u < b_0 L_1$. We define $\ell' = L_1^3$, $B = \Pi_{\max}$, and $\delta = b_0/(3\sqrt{L_1})$. By Fact 4.1(a) and since $u < b_0 L_1$, we have $G(\ell'/u, \delta) \leq \exp(-b_0 L_1/27)$, which can be made at most $(P_{\max} + \Pi_{\max})^{-c_3}$ if b_0 is chosen sufficiently large. By Lemma 4.5(a) and Theorem 4.3, we can efficiently construct a $(P_{\max} + \Pi_{\max}, L_1^3, C_1)$-schedule $\mathcal{S}'$. (See (4.4) for the definition of the C_i.) We apply L_1^4-splitting to $\mathcal{S}'$ to obtain some subproblems, each of which also comes with an (L_1^4, L_1^3, C_1)-schedule. The number of subproblems is at most

$$\begin{aligned} \lceil L_0/(L_1^4 - p_{\max}) \rceil &\leq \lceil L_0/(L_1^4 - b_0 L_1) \rceil \\ &\leq L_0/(L_1^4 - b_0 L_1) + 1 \\ &\leq \frac{L_0}{L_1^4} \cdot (1 + \mathrm{O}(1/L_1^3) + \mathrm{O}(L_1^4/L_0)) \\ &\leq \frac{L_0}{L_1^4} \cdot (1 + \mathrm{O}(1/L_1^3)). \end{aligned} \tag{4.6}$$

Next, we show a recursive scheme to handle each of these subproblems.

General step. Suppose, in general, we have a subproblem which comes with an (L_i^4, L_i^3, C_i)-schedule, $1 \leq i \leq t$. First we dispose of some easy cases. If $i = t$, then $L_i = \mathrm{O}(1)$; by (4.5), $C_i = \mathrm{O}(1)$ also. Thus, we can efficiently find a schedule of length $\mathrm{O}(1)$. So we assume $i \leq t - 1$. Next, suppose $u^2 \geq L_i^3/2$. Note that the "$P_{\max}$" and "$\Pi_{\max}$" values of the given subproblem are, respectively, at most L_i^4 and $C_i \cdot (L_i^4/L_i^3) = \mathrm{O}(L_i^4)$. Thus, if $u^2 \geq L_i^3/2$, then Corollary 4.4 shows that we can construct a schedule of makespan

$$\mathrm{O}\left(L_i^4 \cdot \frac{\log u}{\log\log u} \cdot \left\lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log u} \right\rceil\right). \tag{4.7}$$

So we assume that $u^2 < L_i^3/2$.

We now show a scheme that will construct a feasible schedule for the problem if $u \geq b_0 L_{i+1}$; if $u < b_0 L_{i+1}$, we will show how to reduce this problem to a number of subproblems, each of which comes with an $(L_{i+1}^4, L_{i+1}^3, C_{i+1})$-schedule.

First, suppose $u \geq b_0 L_{i+1}$. We follow our approach for the simple case of the preprocessing step. Define $B = L_i^3/2$, $\ell = L_i^3$, $\ell' = u^2$, and $\delta = 1$. Since $u^2 < L_i^3/2$, we have $B + \ell' \leq \ell$ as required by Theorem 4.3. So, if b_0 is sufficiently large, we will have

$$G(C_i \ell'/(Bu), \delta) \leq (L_i^4 + C_i)^{-c_3}, \tag{4.8}$$

since $L_i^3 < C_i \leq 2L_i^3$ by (4.4) and (4.5). As in the "simple case," we can get an $(L_i^4 + B, \ell', \mathrm{O}(\ell'))$-schedule, apply ℓ'-splitting to it, and solve the resulting subproblems using Corollary 4.4. The final schedule will have makespan as in (4.7).

Finally, suppose $u < b_0 L_{i+1}$. We follow the general idea of the "interesting case" of the preprocessing step. Define $B = L_i^3 - L_{i+1}^3$, $\ell = L_i^3$, $\ell' = L_{i+1}^3$, and $\delta = b_0/(3\sqrt{L_{i+1}})$. Once again, since $u < b_0 L_{i+1}$, we will have (4.8). Thus, as in the "interesting case," we construct an $(L_i^4 + L_i^3, L_{i+1}^3, C_{i+1})$-schedule and apply L_{i+1}^4-splitting to it. As a result, we get some number of subproblems, each of which is equipped with an $(L_{i+1}^4, L_{i+1}^3, C_{i+1})$-schedule; we recurse on these independently. As in the derivation of (4.6), the number of subproblems is at most

$$\lceil (L_i^4 + L_i^3)/(L_{i+1}^4 - b_0 L_{i+1}) \rceil \leq \frac{L_i^4}{L_{i+1}^4} \cdot (1 + \mathrm{O}(1/L_{i+1}^3)). \tag{4.9}$$

Let the final set of subproblems we solve be those that come with an (L_p^4, L_p^3, C_p)-schedule for some p. The product of the terms in (4.6) and (4.9) as i runs from 1 to $p-1$ is $\mathrm{O}(L_0/L_p^4)$. Thus, by (4.7), the final makespan is

$$\begin{aligned} &\mathrm{O}\left((L_0/L_p^4) L_p^4 \cdot \frac{\log u}{\log\log u} \cdot \left\lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log u} \right\rceil \right) \\ &\qquad = \mathrm{O}\left(L_0 \cdot \frac{\log u}{\log\log u} \cdot \left\lceil \frac{\log(\min\{m\mu, p_{\max}\})}{\log\log u} \right\rceil \right), \end{aligned}$$

as claimed by Theorem 1.3(a).

4.4. Basic ideas from earlier constructivizations of the LLL. This section is based on the work of [5, 1, 12]. The main result here is Theorem 4.7, which will be used in section 4.5 to prove Theorem 4.3.

Given an undirected graph $G = (V, E)$, recall that a set $C \subseteq V$ is a *dominating set* of G if and only if all vertices in $V - C$ have some neighbor in C. For any positive integer ℓ, we define G^ℓ to be the graph on the same vertex set V, with two vertices adjacent if and only if they are distinct *and* there is a path of length at most ℓ that connects them in G. We let $\Delta(G)$ denote the maximum degree of the vertices in G. Also, suppose $\mathcal{R}$ is some random process and that each vertex in V represents some event related to $\mathcal{R}$. We say that G is a *dependency graph for* $\mathcal{R}$ if and only if for each $v \in V$ and any set of vertices S such that no element of S is adjacent to v in G, we have that the event corresponding to v is independent of any Boolean combination of the events corresponding to the elements of S.

In Lemma 4.6 and following, the phrase "connected component" means "*maximal* connected subgraph," as usual.

LEMMA 4.6. *Given an undirected graph $G_1 = (V, E)$ with a dominating set C, let G_2 be the subgraph of G_1^3 that is induced by C. Pick an arbitrary maximal independent set I in G_2, and let G_3 be the subgraph of G_2^3 induced by I. Suppose G_1 has a connected component with N vertices. Then G_3 has a connected component with at least $N/((\Delta(G_1)+1)(\Delta(G_1))^3)$ vertices.*

Proof. Let $C_1 = (U, E')$ be a connected component of G_1 with N vertices. Then, the vertices in $C \cap U$ are connected in G_2, which is seen as follows. Suppose $v_1, u_1, u_2, \ldots, u_t, v_t$ is a path in C_1, where v_1 and v_t are in $C \cap U$, and $u_1, \ldots, u_t$ are all in $U - (C \cap U)$. Then, since $C \cap U$ is a dominating set in C_1, for $1 < i < t$, u_i must have some neighbor $v_i \in C \cap U$. Hence, there are paths $v_i, u_i, u_{i+1}, v_{i+1}$ for $1 \leq i < t$ so v_1 and v_t are connected in G_2. Thus, all of the vertices in $C \cap U$ are connected in G_2. Since $C \cap U$ is a dominating set in C_1, it is also easy to check that $|C \cap U| \geq N/(\Delta(C_1)+1) \geq N/(\Delta(G_1)+1)$. Thus, $C \cap U$ yields a connected component C_2 in G_2 that has at least $N/(\Delta(G_1)+1)$ vertices.

Since $\Delta(C_2) \leq (\Delta(G_1))^3 - 1$, one can similarly show that $I \cap (C \cap U)$ yields a connected component C_3 in G_3 that has at least

$$\frac{|C \cap U|}{\Delta(C_2)+1} \geq \frac{|C \cap U|}{(\Delta(G_1))^3} \geq \frac{N}{(\Delta(G_1)+1)(\Delta(G_1))^3}$$

vertices. □

We present a key ingredient of [5, 1, 12] in the following theorem.

THEOREM 4.7. *Let a graph $G = (V, E)$ be a dependency graph for a random process $\mathcal{R}$, with the probability of occurrence of the event represented by any vertex of G being at most r. Run the process $\mathcal{R}$, and let $C \subseteq V$ be the vertices of G that represent the events (among the elements of V) that occurred during the run. (Thus, C is a random subset of V with some distribution.) Let G_1 be the subgraph of G induced by $C \cup C'$, where C' is the set of vertices of G that have at least one neighbor in C. Then, for any $x \geq 1$, the probability of G_1 having a connected component with at least $x(\Delta(G)+1)(\Delta(G))^3$ vertices is at most $|V|\Delta(G)^{-18}\sum_{y \geq x}(\Delta(G)^{18}r)^y$.*

Proof. Observe that, by construction, C is a dominating set for G_1. Construct G_2, I, and G_3 as in the statement of Lemma 4.6. Note that $\Delta(G_1) \leq \Delta(G)$. Thus, by Lemma 4.6, we need only to bound the probability of G_3 having a connected component with x or more vertices.

Suppose that a size-y set S of vertices of G forms a connected component in G_3. Then there is a subtree T of G_3 which spans the vertices in S. T can be represented by a list L which lists all of the vertices that are visited in a depth-first traversal of T. Each vertex in T (except the root) is visited both before its children and after each child (the root is visited only after each child), so each vertex appears on L once for each edge adjacent to it in T. Thus, the length of L is $2(y-1)$. If two vertices are adjacent on L, then they are adjacent in G_3, which implies that the distance between them in G is at most 9. Thus, given G, the number of possible sets S is at most the number of possible lists L, which is at most $|V|$ (the number of choices for the first vertex on L) times $(\Delta(G)^9)^{2(y-1)}$ (the number of choices for the rest of L). Thus, the number of sets S which could possibly correspond to size-y connected components in G_3 is at most $|V|\Delta(G)^{-18}\Delta(G)^{18y}$.

The definition of I implies that the vertices in G_3 form an independent set in G. Furthermore, given any independent set S of size y in G, Bayes's theorem and the definition of dependency graphs show that the probability that all elements of S are

in G_3 is at most r^y. Thus, the probability that G_3 has a connected component of size y is at most $|V|\Delta(G)^{-18}(\Delta(G)^{18}r)^y$. □

4.5. Proof of Theorem 4.3. We now assume the notation of Theorem 4.3 and prove the theorem. Define the following "bad" events:

$$\mathcal{E}(i,k,j_1,j_2) \equiv \left(C'_{\mathcal{S}',\ell'}(i,k,j_1,j_2) \geq \frac{\ell'}{B}\cdot(C'_{\mathcal{S},\ell}(i,(k-B+1)^+,j_1,j_2)+C\delta)\right);$$
$$\mathcal{E}'(i,k,j_1) \equiv (\exists j_2 \geq j_1 :\ \mathcal{E}(i,k,j_1,j_2)).$$

By the assumption of Theorem 4.3, $\Pr[\mathcal{E}(i,k,j_1,j_2)] \leq (\max\{L,B,C\})^{-c_3}\forall(i,k,j_1,j_2)$. Now, for the given instance I, $P_{\max} \leq L$ and $\Pi_{\max} \leq C\cdot\lceil L/\ell\rceil \leq CL$. Thus, in particular, at most CL jobs use any given machine M_i. So, we have $\forall(i,k,j_1)$ that

$$\Pr[\mathcal{E}'(i,k,j_1)] \leq p \doteq CL(\max\{L,B,C\})^{-c_3}. \tag{4.10}$$

The algorithm processes the jobs in the order $J_1, J_2, \ldots$. When it is job J_j's turn, we give it a random delay from $\{0,1,\ldots,B-1\}$ and check if this makes, for any pair (i,k), the event $\mathcal{E}(i,k,1,j)$ true. If so, we temporarily set aside J_j and all *yet-unprocessed* jobs that use machine M_i. Let $\mathcal{J}_1$ denote the set of jobs which *do* get assigned a delay by this process. We shall basically show that, with high probability, the problem of assigning delays to the jobs not in $\mathcal{J}_1$ gets decomposed into a set of much smaller subproblems. To this end, we first set up some notation in order to apply Theorem 4.7.

Construct an undirected graph G with the events $\mathcal{E}'(i,k,1)$ as nodes, with an edge between two distinct nodes $\mathcal{E}'(i,k,1)$ and $\mathcal{E}'(i',k',1)$ if and only if either (P1) $i=i'$, or (P2) there is some job that uses both the machines M_i and $M_{i'}$. It is easy to check that G is a valid dependency graph for the events $\mathcal{E}'(i,k,1)$. The number of vertices in G is at most $m(L+B)$. Recall that at most CL jobs use any given machine and that each such job uses at most $L-1$ other machines. Thus, each node can have at most $L+B$ neighbors of type (P1) and at most $CL(L-1)(L+B)$ neighbors of type (P2). So $\Delta(G)$ is at most

$$L+B+CL(L-1)(L+B) \leq CL^2(L+B)-1.$$

Run the above random process of randomly scheduling and setting aside (if necessary) some of the jobs. Let $\mathcal{C}$ be the set of events $\mathcal{E}'(i,k,1)$ that actually happened. Let $\mathcal{C}'$ be the set of nodes of G that have at least one neighbor in $\mathcal{C}$, and let G_1 be the subgraph of G that is induced by $\mathcal{C}\cup\mathcal{C}'$. Thus, by applying Theorem 4.7 with $|V| \leq m(L+B)$, $\Delta(G) \leq CL^2(L+B)-1$, $x=\log m$, and $r=p$, we see from (4.10) that

$$\Pr[G_1 \text{ has a connected component with at least } (CL^2(L+B))^4\log m \text{ nodes}] \leq 1/2 \tag{4.11}$$

if c_3 is appropriately large.

We repeat the above process until all connected components of G_1 have at most $(CL^2(L+B))^4\log m$ nodes. By (4.11), we expect to run the above process at most twice.

What have we achieved? First, let us give all the jobs in $\mathcal{J}_1$ their assigned delays and remove them from consideration. The key observation is as follows. Fix any

remaining job J_j. Then, for *no two* machines M_i and $M_{i'}$ that are both used by J_j can we have two nodes $\mathcal{E}'(i,k,1)$ and $\mathcal{E}'(i',k',1)$ in *different* connected components of G_1. This is because $\mathcal{E}'(i,k,1)$ and $\mathcal{E}'(i',k',1)$ are neighbors in G. Thus, the problem in each connected component of G_1 can be solved completely *independently* of the other connected components.

So all connected components of G_1 have at most $(CL^2(L+B))^4 \log m$ nodes. To further reduce this component size, we repeat the above process on each connected component CC_t of G_1 separately, as follows. Fix any such CC_t. Define $f_1(i)$ to be the least index j such that $J_j \notin \mathcal{J}_1$ and such that J_j uses M_i. (If all jobs that use M_i are in $\mathcal{J}_1$, we define $f_1(i) = n+1$ for convenience.) Note that all jobs J_j that use M_i and have $j \geq f_1(i)$ lie outside the set $\mathcal{J}_1$. We process the jobs lying outside $\mathcal{J}_1$ in order as before. When it is job J_j's turn, we give it a random delay from $\{0,1,\ldots,B-1\}$ and check if this makes, for any pair (i,k), the event $\mathcal{E}(i,k,f_1(i),j)$ true. (This is mostly the same as before, except that now we have "$f_1(i)$" in place of "1".) If so, we temporarily set aside J_j and all yet-unprocessed jobs lying outside $\mathcal{J}_1$ that use M_i.

We proceed similarly as above. Let $\mathcal{J}_2$ denote the set of jobs which get assigned a delay by this process. We now show that the problem of assigning delays to the jobs not in $\mathcal{J}_1 \cup \mathcal{J}_2$ gets decomposed into even smaller subproblems with high probability. In place of the bad events $\{\mathcal{E}'(i,k,1)\}$, the bad events now are $\{\mathcal{E}'(i,k,f_1(i))\}$. We can once again invoke Theorem 4.7; we take $|V| \leq (CL^2(L+B))^4 \log m$, $\Delta(G) \leq CL^2(L+B)-1$, $x = \log\log m$, and $r = p$. As before, if c_3 is large enough, we expect to repeat this process at most twice before ensuring that all resulting "connected components" have at most $(CL^2(L+B))^4 \log\log m$ nodes.

We now consider any connected component CC'_t remaining after the above two passes. (Once again, all these components can be handled independently.) Define $f_2(i)$ to be the least index j such that $J_j \notin (\mathcal{J}_1 \cup \mathcal{J}_2)$ and such that J_j uses M_i. We now show how to give delays to all jobs lying outside $(\mathcal{J}_1 \cup \mathcal{J}_2)$ in a manner that avoids all the events $\mathcal{E}'(i,k,f_2(i))$. There are two cases.

Case I. $\log\log m \leq L+B+C$. In this case, the number of "nodes" (events $\mathcal{E}'(i,k,f_2(i))$) in CC'_t is $\mathrm{poly}(L,B,C)$. Thus, if we start with a random B-delayed schedule for the jobs associated with CC'_t, the probability that at least one "bad" event associated with CC'_t (i.e., at least one node of CC'_t) happens is at most $1/2$, if c_3 is large enough. So we expect to run this process on CC'_t at most twice.

Case II. $\log\log m > L+B+C$. The number of nodes in CC'_t is $\mathrm{O}(\mathrm{poly}(\log\log m))$ in this case. So the number of machines associated with CC'_t is also $\mathrm{O}(\mathrm{poly}(\log\log m))$, and hence the number of jobs associated with CC'_t is at most $\mathrm{O}(L \cdot \mathrm{poly}(\log\log m))$, i.e., $\mathrm{O}(\mathrm{poly}(\log\log m))$.

We recall the LLL.

LEMMA 4.8 (see [7]). *Let $E_1, E_2, \ldots, E_\ell$ be any events with $\Pr[E_i] \leq q \forall i$. If each E_i is mutually independent of all but at most d of the other events E_j and if $eq(d+1) \leq 1$, then $\Pr[\bigwedge_{i=1}^{\ell} \overline{E_i}] > 0$.*

As seen above, any event $\mathcal{E}'(i,k,f_2(i))$ depends on at most $CL^2(L+B)-1$ other such events. Also, $\Pr[\mathcal{E}'(i,k,f_2(i))] \leq p \forall i,k$. Thus, if c_3 is sufficiently large, the LLL shows that *there exists* a way of giving a delay in $\{0,1,\ldots,B-1\}$ to each job associated with CC'_t in order to avoid all the events $\mathcal{E}'(i,k,f_2(i))$ associated with CC'_t. But here, there are at most $\mathrm{O}(\mathrm{poly}(\log\log m))$ jobs, and each has only $B \leq \log\log m$ possible initial delays! Thus, *exhaustive search* can be applied to find a "good" B-delayed schedule that we know to exist: the time needed for CC'_t is at most

$$(\log\log m)^{\mathrm{O}(\mathrm{poly}(\log\log m))} = m^{o(1)}.$$

Let $\mathcal{S}''$ be the final delayed schedule produced. Consider any interval $(k, k+\ell')$. We have

$$\begin{aligned} C_{\mathcal{S}'',\ell'}(i,k) &\le \frac{\ell'}{B} \cdot (C'_{\mathcal{S},\ell}(i,(k-B+1)^+,1,f_1(i)-1)+C\delta) \\ &+ \frac{\ell'}{B} \cdot (C'_{\mathcal{S},\ell}(i,(k-B+1)^+,f_1(i),f_2(i)-1)+C\delta) \\ &+ \frac{\ell'}{B} \cdot (C'_{\mathcal{S},\ell}(i,(k-B+1)^+,f_2(i),n)+C\delta) \\ &= \frac{\ell'}{B} \cdot (C'_{\mathcal{S},\ell}(i,(k-B+1)^+,1,n)+3C\delta) \\ &\le \frac{\ell'}{B} \cdot (C+3C\delta) \end{aligned}$$

as required.

It is also easy to check via linearity of expectation that the expected running time of the algorithm is poly(m, L, ℓ, C). This concludes the proof of Theorem 4.3.

Acknowledgments. Aravind Srinivasan thanks David Shmoys for introducing him to this area, for sharing many of his insights, and for his several suggestions. He also thanks Cliff Stein and Joel Wein for their suggestions and many helpful discussions. The authors thank Uri Feige for bringing the work of [16] and [22] to their attention, and Bruce Maggs and Andréa Richa for sending them an updated version of [12]. The authors also thank the referees for their many helpful suggestions.

REFERENCES

[1] N. Alon, *A parallel algorithmic version of the Local Lemma*, Random Structures and Algorithms, 2 (1991), pp. 367–378.

[2] N. Alon and A. Srinivasan, *Improved parallel approximation of a class of integer programming problems*, Algorithmica, 17 (1997), pp. 449–462.

[3] D. Applegate and W. Cook, *A computational study of the job-shop scheduling problem*, ORSA J. Comput., 3 (1991), pp. 149–156.

[4] A. Bar-Noy, R. Canetti, S. Kutten, Y. Mansour, and B. Schieber, *Bandwidth allocation with preemption*, in Proceedings of the 27th Annual ACM Symposium on Theory of Computing, Las Vegas, NV, 1995, ACM, New York, 1995, pp. 616–625.

[5] J. Beck, *An algorithmic approach to the Lovász Local Lemma*, Random Structures and Algorithms, 2 (1991), pp. 343–365.

[6] J. Carlier and E. Pinson, *An algorithm for solving the job-shop problem*, Management Sci., 35 (1989), pp. 164–176.

[7] P. Erdös and L. Lovász, *Problems and results on* 3*-chromatic hypergraphs and some related questions*, in Infinite and Finite Sets, A. Hajnal, R. Rado, and V. T. Sós, eds., Colloq. Math. Soc. Janos Bolyai 11, North Holland, Amsterdam, 1975, pp. 609–627.

[8] U. Feige and C. Scheideler, *Improved bounds for acyclic job shop scheduling*, in Proceedings of the 30th Annual ACM Symposium on Theory of Computing, Dallas, TX, 1998, ACM, New York, 1998, pp. 624–633.

[9] L. A. Hall, *Approximability of flow shop scheduling*, in Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science, Milwaukee, WI, 1995, pp. 82–91.

[10] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, *Sequencing and scheduling: algorithms and complexity*, in Logistics of Production and Inventory, Handbooks Oper. Res. Management Sci., 4, S. C. Graves et al., eds., Elsevier, New York, 1993, pp. 445–522.

[11] F. T. Leighton, B. Maggs, and S. Rao, *Packet routing and jobshop scheduling in* O(*congestion + dilation*) *steps*, Combinatorica, 14 (1994), pp. 167–186.

[12] F. T. Leighton, B. Maggs, and A. Richa, *Fast algorithms for finding* O(*congestion + dilation*) *packet routing schedules*, Combinatorica, 19 (1999), pp. 375–401.

[13] S. Mahajan, E. A. Ramos, and K. V. Subrahmanyam, *Solving some discrepancy problems in NC*, in Proceedings of the Annual Conference on Foundations of Software Technology and

Theoretical Computer Science, Lecture Notes in Comput. Sci. 1346, Springer, New York, 1997, pp. 22–36.

[14] P. Martin and D. B. Shmoys, *A new approach to computing optimal schedules for the job-shop scheduling problem*, in Proceedings of the MPS Conference on Integer Programming and Combinatorial Optimization, Vancouver, B.C., Canada, 1996, Lecture Notes in Comput. Sci. 1084, Springer, Berlin, 1996, pp. 389–403.

[15] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, Cambridge, UK, 1995.

[16] G. Rayzman, *Approximation Techniques for Job-Shop Scheduling Problems,* M.Sc. Thesis, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot, Israel, 1996.

[17] C. Scheideler, *Universal Routing Strategies for Interconnection Networks*, Lecture Notes in Comput. Sci. 1390, Springer, New York, 1998.

[18] J. P. Schmidt, A. Siegel, and A. Srinivasan, *Chernoff–Hoeffding bounds for applications with limited independence*, SIAM J. Discrete Math., 8 (1995), pp. 223–250.

[19] S. V. Sevast'yanov, *Efficient construction of schedules close to optimal for the cases of arbitrary and alternative routes of parts*, Soviet Math. Dokl., 29 (1984), pp. 447–450.

[20] S. V. Sevast'yanov, *Bounding algorithm for the routing problem with arbitrary paths and alternative servers*, Kibernetika, 22 (1986), pp. 74–79, 134 (in Russian).

[21] D. B. Shmoys, C. Stein, and J. Wein, *Improved approximation algorithms for shop scheduling problems*, SIAM J. Comput., 23 (1994), pp. 617–632.

[22] Yu. N. Sotskov and N. V. Shaklevich, NP-*hardness of shopt-scheduling problems with three jobs,* Discrete Appl. Math., 59 (1995), pp. 237–266.

[23] D. P. Williamson, L. A. Hall, J. A. Hoogeveen, C. A. J. Hurkens, J. K. Lenstra, S. V. Sevast'yanov, and D. B. Shmoys, *Short shop schedules*, Oper. Res., 45 (1997), pp. 288–294.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 93–115

ENUMERATION OF EQUICOLORABLE TREES*

NICHOLAS PIPPENGER†

Abstract. A tree, being a connected acyclic graph, can be bicolored in two ways, which differ from each other by exchange of the colors. We shall say that a tree is *equicolorable* if these bicolorings assign the two colors to equal numbers of vertices. Labelled equicolored trees have been enumerated several times in the literature, and from this result it is easy to enumerate labelled equicolorable trees. The result is that the probability that a randomly chosen n-vertex labelled tree is equicolorable is asymptotically just twice the probability that its vertices would be equicolored if they were assigned colors by independent unbiased coin flips. Our goal in this paper is the enumeration of unlabelled equicolorable trees (that is, trees up to isomorphism), both exactly (in terms of generating functions) and asymptotically. We treat both the rooted and unrooted versions of this problem and conclude that in either case the probability that a randomly chosen n-vertex unlabelled tree is equicolorable is asymptotically 1.40499... times as large as the probability that it would be equicolored if its vertices were assigned colors by independent unbiased coin flips.

Key words. generating functions, asymptotic enumeration

AMS subject classifications. 05A15, 05A16, 05C05

PII. S0895480100368463

1. Introduction.

Our goal in this paper is the enumeration, exact and asymptotic, of certain kinds of trees. A tree can have its vertices bicolored (so that adjacent vertices are oppositely colored) in exactly two ways, which differ by exchange of the colors. We shall be particularly interested in those trees for which equal numbers of vertices are assigned the two colors; we call such trees *equicolorable*. (It is tempting to call them "balanced," but the term "balanced trees" is already in use for several kinds of objects different from those treated here.)

Our solution to this problem also yields the enumeration of *equicolored* trees, that is, equicolorable trees that have been assigned one of their two equitable bicolorings. For trees that are labelled or rooted, the distinction between enumerating "equicolorable" and "equicolored" trees is trivial, for there are exactly two equicolored trees for each equicolorable one. However, when we enumerate unlabelled and unrooted trees, the distinction is significant, for to enumerate equicolored trees we must count each equicolorable tree once or twice depending on whether or not there is a color-exchanging automorphism of the tree.

Our approach to these problems can be sketched as follows. For the sake of example we consider unlabelled but rooted trees. We consider bicolorings of these trees (not necessarily equicolorings) in which the vertices are colored red and blue, and in which the root is colored red. Specifying the color of the root fixes the colors of all other vertices. Let $r_{l,m}$ denote the number of such red-rooted trees with l red and m blue vertices. Let

$$r(x,y) = \sum_{l \geq 1, m \geq 0} r_{l,m}\, x^l y^m$$

*Received by the editors February 28, 2000; accepted for publication (in revised form) October 17, 2000; published electronically January 16, 2001. This research was supported by an NSERC research grant.

http://www.siam.org/journals/sidma/14-1/36846.html

†Department of Computer Science, University of British Columbia, Vancouver, BC, V6T 1Z4, Canada (nicholas@cs.ubc.ca).

be the generating function for red-rooted trees in which the coefficient of $x^l y^m$ is the number of trees with l red vertices and m blue vertices. Then $r(y, x)$ is the generating function for blue-rooted trees.

By standard combinatorial methods, we obtain a functional equation determining $r(x, y)$. Then we make the substitutions $x = z \exp(i\vartheta)$ and $y = z \exp(-i\vartheta)$ and thus define

$$\begin{aligned} r_\vartheta(z) &= r\big(z \exp(i\vartheta), z \exp(-i\vartheta)\big) \\ &= \sum_{\substack{n \geq 1 \\ k \equiv n \pmod 2}} r_{(n+k)/2,(n-k)/2}\ z^n \exp(ik\vartheta), \end{aligned}$$

which is a trigonometric series in which the coefficient of $z^n \exp(ik\vartheta)$ is the number of red-rooted trees with n vertices and k more red vertices than blue vertices. In fact, it will be technically more convenient to work with the related series

$$\begin{aligned} c_\vartheta(z) &= \frac{r_\vartheta(z) + r_{-\vartheta}(z)}{2}, \\ &= \sum_{\substack{n \geq 1 \\ k \equiv n \pmod 2}} r_{(n+k)/2,(n-k)/2}\ z^n \cos(k\vartheta) \end{aligned}$$

in which red-rooted and blue-rooted trees are each counted with weight one-half. In deriving the functional equation for this series, the conjugate series

$$\begin{aligned} s_\vartheta(z) &= \frac{r_\vartheta(z) - r_{-\vartheta}(z)}{2i}, \\ &= \sum_{\substack{n \geq 1 \\ k \equiv n \pmod 2}} r_{(n+k)/2,(n-k)/2}\ z^n \sin(k\vartheta) \end{aligned}$$

will play an auxiliary role. Next we use the formula

$$\frac{1}{2\pi} \int_0^{2\pi} \cos(k\vartheta)\, d\vartheta = \begin{cases} 1 & \text{if } k = 0, \\ 0 & \text{if } k \neq 0 \end{cases}$$

to segregate the terms corresponding to equicolored trees from the others. Specifically, the generating function

$$r^*(z) = \frac{1}{2\pi} \int_0^{2\pi} c_\vartheta(z)\, d\vartheta$$

enumerates equicolorable rooted trees, since we have counted both the red- and blue-rooted versions of the tree, each with weight one-half. This method of extracting the diagonal terms from a bivariate power series appears to be new in the combinatorial literature. Hautus and Klarner [H] give a method based on contour integration. Our method is, of course, equivalent but more convenient in the case at hand because of the role of the conjugate trigonometric series indicated above.

Finally, by standard analytic methods, we determine the asymptotic behavior of the coefficients in $r^*(z)$. To do this we determine, for each ϑ in the interval $[0, 2\pi)$ of integration, the behavior of the coefficients in $c_\vartheta(z)$; then we estimate the integral of the resulting expression. Since $c_\vartheta(z)$ is periodic in ϑ with period 2π, the integral

can be taken over any interval of length 2π. In fact, since the greatest contributions to the integral come when ϑ is near one of two points, 0 and π, it will be technically convenient to take the interval of integration to be $[-\pi/2, 3\pi/2)$, which has 0 and π as interior points.

The enumeration of equicolorable labelled trees (either rooted or unrooted) follows easily from the enumeration of equicolored labelled trees, which has been dealt with already in the literature. Nevertheless, in section 2 we shall solve this problem again using methods that will extend later to the unlabelled case. This will provide a testing ground for our methods in a setting where the outcome is known in advance. The result is that the probability that a randomly chosen n-vertex labelled tree is equicolorable is asymptotically just twice the probability that its vertices would be equicolored if they were assigned colors by independent unbiased coin flips (which is $\binom{n}{n/2}/2^n \sim (2/\pi n)^{1/2}$ for n even and 0 for n odd.) In section 3, we shall enumerate equicolorable rooted unlabelled trees. Finally, in section 4, we shall enumerate equicolorable unrooted unlabelled trees. For both rooted and unrooted trees, we conclude that the probability that a randomly chosen n-vertex unlabelled tree is equicolorable is asymptotically $1.40499\ldots$ times as large as the probability that it would be equicolored if its vertices were assigned colors by independent unbiased coin flips.

2. Labelled trees. It was Cayley [C5] who in 1889 first stated that the number of labelled trees on n vertices is n^{n-2}, although this result is implicit in earlier related work by Sylvester [S2] in 1857 and Borchardt [B] in 1860. Many proofs of this result are now known; see Moon [M1, M2]. The one most relevant here, due to Pólya [P1, P2], is as follows.

A labelled tree on n vertices can be rooted in exactly n different ways, so it will suffice to show that the number R_n of rooted labelled trees is n^{n-1}. Let

$$R(z) = \sum_{n\geq 1} \frac{R_n}{n!}$$

be the exponential generating function for rooted labelled trees. Pólya's *component principle* states that if $F(z)$ is the exponential generating function for labelled "components," then

$$G(z) = \exp F(z) \tag{2.1}$$

is the exponential generating function for labelled structures comprising zero or more disjoint components. Since a rooted tree comprises a root (enumerated by z), together with zero or more disjoint rooted trees (the subtrees adjacent to the root, enumerated by $\exp R(z)$), $R(z)$ satisfies the functional equation

$$R(z) = z \exp R(z). \tag{2.2}$$

From this equation, Lagrange's inversion formula gives $n^{n-1}/n!$ as the coefficient $[z^n]R(z)$ of z^n in $R(z)$. Thus we conclude that $R_n = n^{n-1}$, as claimed.

Since we shall work later with functional equations to which Lagrange's inversion formula cannot be applied, it will be instructive to see how even without it the asymptotic behavior of the coefficients of $R(z)$ can be extracted from (2.2).

The key idea will be the use of Darboux's lemma to deduce the asymptotic behavior of the coefficients from the nature of the singularities of $R(z)$. To find the singularities of $R(z)$ as a function of z, we write (2.2) as $\Phi\big(z, R(z)\big) = 0$, where

$$\Phi(z, w) = z \exp w - w.$$

To locate the singularities, we calculate

$$\frac{\partial}{\partial w}\Phi(z,w) = \Phi(z,w) + w - 1.$$

The singularities occur when this derivative and $\Phi(z,w)$ vanish simultaneously for $w = R(z)$. This happens only for $w = W_0 = R(Z_0) = 1$ and $z = Z_0 = 1/e$. To expand $R(z)$ in the neighborhood of $z = Z_0$, we calculate

$$\frac{\partial^2}{\partial w^2}\Phi(z,w) = \frac{\partial}{\partial w}\Phi(z,w) + 1 = \Phi(z,w) + w$$

and

$$\frac{\partial}{\partial z}\Phi(z,w) = \big(\Phi(z,w) + w\big)/z.$$

Then we have

$$\frac{\partial^2}{\partial w^2}\Phi(z,w)\big|_{w=W_0,z=Z_0} = 1$$

and

$$\frac{\partial}{\partial z}\Phi(z,w)\big|_{w=W_0,z=Z_0} = 1/Z_0,$$

so that

$$\begin{aligned}\Phi(z,w) = \frac{1}{2}(w-W_0)^2 + O\big((w-W_0)^3\big)\\ - (1-z/Z_0) + O\big((w-W_0)(1-z/Z_0)\big) + O\big((1-z/Z_0)^2\big).\end{aligned}$$

Thus at $z = Z_0$, $R(z)$ has a branch point of order 2 and an expansion of the form

$$R(z) = A(z) + B(z)(1-z/Z_0)^{1/2},$$

where $A(z) = 1+O(z)$ and $B(z) = -2^{1/2}+O(z)$ are analytic functions of z. Applying Darboux's lemma [D] (see also Knuth and Wilf [K]), we conclude that $[z^n]R(z)$ is asymptotic to $e^n/n^{3/2}(2\pi)^{1/2}$ and thus by Stirling's formula that R_n is asymptotic to n^{n-1}.

The problem of enumerating equicolored labelled trees will be reduced to the problem of enumerating certain "rooted spanning trees." Let $K_{l,m} = (V,W,E)$ denote the complete bipartite graph with l red vertices $V = \{v_1,\dots,v_l\}$, m blue vertices $W = \{w_1,\dots,w_m\}$, and lm edges E. (Each edge is an unordered pair comprising one vertex from V and one from W.) Let $R_{l,m}$ denote the number of red-rooted spanning trees in $K_{l,m}$, that is, the number of spanning trees in which one of the red vertices has been distinguished as the root. Since each unrooted spanning tree in $K_{l,m}$ can be assigned a red root in exactly l different ways, the number of unrooted spanning trees in $K_{l,m}$ is $R_{l,m}/l$. In particular, there are $R_{m,m}/m$ unrooted spanning trees in $K_{m,m}$. Each equicolored labelled tree on $n = 2m$ vertices gives rise to $\binom{2m}{m}$ unrooted spanning trees in $K_{m,m}$, since the $2m$ vertices $U = \{u_1,\dots,u_{2m}\}$ can be partitioned into m red vertices V and m blue vertices W in exactly $\binom{2m}{m}$ different ways. Thus there are $\binom{2m}{m}R_{m,m}/m$ equicolored labelled trees. Since each equicolorable labelled tree can be equicolored in exactly two different ways, there are $\binom{2m}{m}R_{m,m}/2m$ equicolorable

labelled trees on $n = 2m$ vertices. (There are, of course, no equicolorable trees with an odd number of vertices.)

The problem of enumerating equicolorable labelled trees reduces to the problem of enumerating red-rooted spanning trees in $K_{l,m}$. This latter problem has been solved by Austin [A] (see also Scoins [S1] and Glicksman [G]), who showed that $R_{l,m} = l^m m^{l-1}$. The proofs of Austin and Scoins are based on the following idea.

Let

$$R(x,y) = \sum_{l\geq 1, m\geq 0} \frac{R_{l,m}}{l!\, m!}$$

be the bivariate exponential generating function for red-rooted spanning trees in $K_{l,m}$. The component principle analogous to (2.1) for bivariate exponential generating functions is

$$G(x,y) = \exp F(x,y),$$

where $F(x,y)$ is the generating function for components and $G(x,y)$ is the generating function for structures comprising zero or more disjoint components. Since a rooted spanning tree with a red root comprises a red root (enumerated by x), together with zero or more disjoint rooted trees (which have blue roots, and are thus enumerated by $R(y,x)$), $R(x,y)$ satisfies the functional equation

$$R(x,y) = x \exp R(y,x). \tag{2.3}$$

Austin and Scoins use this equation, together with Lagrange's inversion formula, to show that the coefficient $[x^l y^m]R(x,y)$ of $x^l y^m$ in $R(x,y)$ is $l^m m^{l-1}/l!\, m!$. Thus $R_{l,m} = l^m m^{l-1}$, as claimed. In particular, $R_{m,m} = m^{2m-1}$. As before, we shall derive this asymptotic behavior without using Lagrange's inversion formula.

As indicated in the introduction, we begin by making the substitutions $x = z\exp(i\vartheta)$ and $y = z\exp(-i\vartheta)$ and thus defining

$$R_\vartheta(z) = R\big(z\exp(i\vartheta), z\exp(-i\vartheta)\big). \tag{2.4}$$

From (2.3) and (2.4) we obtain

$$R_\vartheta(z) = z\exp\big(i\vartheta + R_{-\vartheta}(z)\big) \tag{2.5}$$

as the functional equation satisfied by $R_\vartheta(z)$.

We shall be interested in real values of ϑ, and it will turn out that the singularities of $R_\vartheta(z)$ occur at real values of z. It will be convenient therefore to work with relatives of $R_\vartheta(z)$ that are real when ϑ and z are real. Thus we define

$$C_\vartheta(z) = \frac{R_\vartheta(z) + R_{-\vartheta}(z)}{2} \tag{2.6}$$

and

$$S_\vartheta(z) = \frac{R_\vartheta(z) - R_{-\vartheta}(z)}{2i}. \tag{2.7}$$

We can find the functional equations satisfied by $C_\vartheta(z)$ and $S_\vartheta(z)$ by substituting (2.5) into (2.6) and (2.7), then substituting $R_\vartheta(z) = C_\vartheta(z) + iS_\vartheta(z)$ and $R_{-\vartheta(z)} = C_\vartheta(z) - iS_\vartheta(z)$ into the result to obtain

$$C_\vartheta(z) = z\exp C_\vartheta(z)\, \cos\big(\vartheta - S_\vartheta(z)\big) \tag{2.8}$$

and

$$S_\vartheta(z) = z \exp C_\vartheta(z) \, \sin\bigl(\vartheta - S_\vartheta(z)\bigr). \tag{2.9}$$

We shall need to determine the singularities of $C_\vartheta(z)$ as a function of z with ϑ fixed. To find them, we square (2.8) and (2.9) and add them to obtain

$$C_\vartheta(z)^2 + S_\vartheta(z)^2 = z^2 \exp\bigl(2C_\vartheta(z)\bigr).$$

We then use this result to eliminate $S_\vartheta(z)$ from (2.8), obtaining

$$C_\vartheta(z) = z \exp C_\vartheta(z) \, \cos\Bigl(\vartheta - \bigl(z^2 \exp\bigl(2C_\vartheta(z)\bigr) - C_\vartheta(z)^2\bigr)^{1/2}\Bigr).$$

This equation can be written as $\Phi_\vartheta\bigl(z, C_\vartheta(z)\bigr) = 0$, where

$$\Phi_\vartheta(z,w) = z \exp w \, \cos\Bigl(\vartheta - \bigl(z^2 \exp(2w) - w^2\bigr)^{1/2}\Bigr) - w.$$

To locate the singularities of $C_\vartheta(z)$, we calculate

$$\frac{\partial}{\partial w} \Phi_\vartheta(z,w) = \Phi_\vartheta(z,w) - 1 + z^2 \exp(2w).$$

The singularities occur when this derivative and $\Phi(z,w)$ vanish simultaneously for $w = C_\vartheta(z)$, so that we have

$$Z_\vartheta^\pm = \pm \exp -C_\vartheta(Z_\vartheta^\pm). \tag{2.10}$$

Substituting this relation into (2.8) and (2.9) yields

$$C_\vartheta(Z_\vartheta^\pm) = \pm \cos\bigl(\vartheta - S_\vartheta(Z_\vartheta^\pm)\bigr), \tag{2.11}$$

and

$$S_\vartheta(Z_\vartheta^\pm) = \pm \sin\bigl(\vartheta - S_\vartheta(Z_\vartheta^\pm)\bigr), \tag{2.12}$$

where, of course, we must take the same sign throughout all three equations.

The solution to (2.11) and (2.12), and similar pairs of equations, can be expressed in terms of the coordinates of the cycloid curve, defined parametrically by

$$\begin{aligned} X(t) &= t + \sin t, \\ Y(t) &= \cos t. \end{aligned}$$

This curve is the locus of a marked point on a hoop of radius 1 that rolls without slipping on the line $Y = -1$. We define the cycloid function by

$$\operatorname{cyc} \vartheta = Y\bigl(X^{-1}(\vartheta)\bigr).$$

This function is periodic with period 2π. It has a crest $\operatorname{cyc} \vartheta = 1 - \vartheta^2/8 + O(\vartheta^4)$ in the neighborhood of $\vartheta = 0$, and a cusp $\operatorname{cyc} \vartheta = -1 + (9/2)^{1/3}(\vartheta - \pi)^{2/3}$ in the neighborhood of $\vartheta = \pi$. (Some discussions of the cycloid assume that the hoop rolls on the line $Y = 0$, or that $\vartheta = 0$ corresponds to a cusp rather than a crest, or both.) We shall also need the cocycloid curve, defined by

$$\begin{aligned} X(t) &= t + \sin t, \\ Z(t) &= \sin t, \end{aligned}$$

and the cocycloid function, defined by

$$\operatorname{cocyc}\vartheta = Z\big(X^{-1}(\vartheta)\big).$$

This function represents the lag of the center of the hoop behind the marked point. It is also periodic with period 2π, and it has inflections at $\vartheta = 0$ and $\vartheta = \pi$ with expansions $\operatorname{cocyc}\vartheta = \vartheta/2 + O(\vartheta^3)$ and $\operatorname{cocyc}\vartheta = -6^{1/3}(\vartheta-\pi)^{1/3} + O(\vartheta-\pi)$ in the neighborhoods of these points, respectively. Finally, we have the identity $\operatorname{cyc}^2\vartheta + \operatorname{cocyc}^2\vartheta = 1$.

These definitions allow us to solve (2.11) and (2.12). We have $W_\vartheta^+ = C_\vartheta(Z_\vartheta^+) = \operatorname{cyc}\vartheta$, so that

$$Z_\vartheta^+ = \exp(-\operatorname{cyc}\vartheta). \tag{2.13}$$

Taking the minus sign in (2.11) and (2.12) is equivalent to shifting ϑ by π, so we have

$$Z_\vartheta^- = -\exp\big(-\operatorname{cyc}(\vartheta-\pi)\big). \tag{2.14}$$

It may appear paradoxical that we have found two singularities for $C_\vartheta(z)$, whereas there was only one for $R(z) = C_0(z)$. The resolution of this paradox will appear shortly.

To expand $C_\vartheta(z)$ in the neighborhood of $z = Z_\vartheta^+$, we calculate

$$\frac{\partial^2}{\partial w^2}\Phi_\vartheta(z,w) = \frac{\partial}{\partial w}\Phi_\vartheta(z,w) + 2z^2\exp(2w) = \Phi_\vartheta(z,w) - 1 + 3z^2\exp(2w)$$

and

$$\frac{\partial}{\partial z}\Phi_\vartheta(z,w) = \big(\Phi_\vartheta(z,w) + w + z^2\exp(2w)\big)/z.$$

We then have

$$\frac{\partial^2}{\partial w^2}\Phi_\vartheta(z,w)\big|_{w=W_\vartheta^+,z=Z_\vartheta^+} = 2$$

and

$$\frac{\partial}{\partial z}\Phi_\vartheta(z,w)\big|_{w=W_\vartheta^+,z=Z_\vartheta^+} = (1+W_\vartheta^+)/Z_\vartheta^+,$$

so that

$$\begin{aligned}\Phi_\vartheta(z,w) = (w-W_\vartheta^+)^2 + O\big((w-W_\vartheta^+)^3\big) - (1+W_\vartheta^+)(1-z/Z_\vartheta^+)\\ + O\big((w-W_\vartheta^+)(1-z/Z_\vartheta^+)\big) + O\big((1-z/Z_\vartheta^+)^2\big).\end{aligned}$$

Thus at Z_ϑ^+, $C_\vartheta(z)$ has a branch point of order 2 and an expansion of the form

$$C_\vartheta(z) = A_\vartheta^+(z) + B_\vartheta^+(z)(1-z/Z_\vartheta^+)^{1/2},$$

where $A_\vartheta^+(z) = \operatorname{cyc}\vartheta + O(z-Z_\vartheta^+)$ and $B_\vartheta^+(z) = -(1+\operatorname{cyc}\vartheta)^{1/2} + O(z-Z_\vartheta^+)$ are analytic functions of z. Furthermore, the constants in the O-terms are uniform in ϑ, since they vary continuously on the compact fundamental domain $[-\pi/2, 3\pi/2)$ of ϑ. Similar arguments give

$$C_\vartheta(z) = A_\vartheta^-(z) + B_\vartheta^-(z)(1-z/Z_\vartheta^-)^{1/2},$$

where $A^-_\vartheta(z) = \operatorname{cyc}(\vartheta-\pi)+O(z-Z^-_\vartheta)$ and $B^+_\vartheta(z) = -\bigl(1+\operatorname{cyc}(\vartheta-\pi)\bigr)^{1/2}+O(z-Z^-_\vartheta)$ for the expansion of $C_\vartheta(z)$ in the neighborhood of $z = Z^-_\vartheta$. These formulae resolve the paradox mentioned above: the singularities of $C_\vartheta(z)$ "blink" at the cusps of the cycloid, where the factor multiplying $(1 - z/Z^\pm_\vartheta)^{1/2}$ vanishes. For the singularity at Z^-_ϑ, this occurs at $\vartheta = 0$, so that $R(z) = C_0(z)$ has just one singularity at $z = Z_0 = Z^+_0$.

We now proceed, as indicated in the introduction, to extract the desired asymptotic information from $C_\vartheta(z)$. We define

$$R^*(z) = \frac{1}{2\pi}\int_{-\pi/2}^{3\pi/2} C_\vartheta(z)\,d\vartheta,$$

a power series in z in which the coefficients of odd powers of z vanish and the coefficient of the even power z^{2m} is the same as the coefficient of the term $x^m y^m$ in $R(x, y)$. Thus we have

$$\frac{R_{m,m}}{m!^2} = \frac{1}{2\pi}\int_{-\pi/2}^{3\pi/2} [z^{2m}]\,C_\vartheta(z)\,d\vartheta.$$

The largest contributions to this integral come from those ϑ for which the singularities of $C_\vartheta(z)$ are closest to the origin; for the singularity at Z^+_ϑ this occurs for ϑ near 0, and for Z^-_ϑ, near π. Accordingly, we set

$$\varepsilon(n) = \left(\frac{48\log n}{n}\right)^{1/2}$$

and break the interval $I = [-\pi/2, 3\pi/2)$ into three parts: $J^+ = [-\varepsilon(n), \varepsilon(n)]$, $J^- = [\pi - \varepsilon(n), \pi + \varepsilon(n)]$, and $K = I \setminus (J^+ \cup J^-)$.

First we consider the integral over ϑ in K. We have $Z^+_\vartheta = \exp -\operatorname{cyc}\vartheta = \exp -\bigl(1-\vartheta^2/8 + O(\vartheta^4)\bigr)$. Thus for ϑ not in J^+, we have $Z^+_\vartheta \ge r$, where $r = \exp -\bigl(1 - \varepsilon(n)^2/16\bigr) = \exp -\bigl(1 - 3\log n/n\bigr)$. Similarly, for ϑ not in J^-, we have $Z^-_\vartheta \le -r$. Thus for ϑ in K, $C_\vartheta(z)$ is analytic throughout the disk of radius r centered at the origin. By Cauchy's theorem, we have

$$\begin{aligned}[z^n]C_\vartheta(z) &= \frac{1}{2\pi i}\oint \frac{C_\vartheta(z)\,d\vartheta}{z^{n+1}}\\ &= O\left(\frac{1}{r^n}\right)\\ &= O\left(\frac{e^n}{n^3}\right),\end{aligned}$$

where the contour integral is taken in the positive sense around the circle of radius r centered at the origin. Thus we have

$$\frac{1}{2\pi}\int_K [z^n]C_\vartheta(z)\,d\vartheta = O\left(\frac{e^n}{n^3}\right).$$

For ϑ in J^+, we have by Darboux's lemma

$$\begin{aligned}[z^n]C_\vartheta(z) &= -(1+\operatorname{cyc}\vartheta)^{1/2}\binom{n-3/2}{n}\left(\frac{1}{Z^+_\vartheta}\right)^n + O\left(\binom{n-5/2}{n}\left(\frac{1}{Z^+_\vartheta}\right)^n\right)\\ &= \frac{e^n}{(2\pi)^{1/2}n^{3/2}}\left(1 + O\left(\frac{(\log n)^2}{n}\right)\right)\exp -(n\vartheta^2/8),\end{aligned}$$

where we have estimated the leading factor by

$$-(1+\operatorname{cyc}\vartheta)^{1/2} = -\bigl(2+O(\vartheta^2)\bigr)^{1/2} = -2^{1/2}\bigl(1+O\bigl(\varepsilon(n)^2\bigr)\bigr) = -2^{1/2}\left(1+O\left(\frac{\log n}{n}\right)\right),$$

the singular point by

$$\begin{aligned} Z_\vartheta^+ &= \exp -\operatorname{cyc}\vartheta = \exp -\bigl(1-\vartheta^2/8+O(\vartheta^4)\bigr) \\ &= \exp -(1-\vartheta^2/8)\bigl(1+O\bigl(\varepsilon(n)^4\bigr)\bigr) = \exp -(1-\vartheta^2/8)\left(1+O\left(\left(\frac{\log n}{n}\right)^2\right)\right), \end{aligned}$$

and the binomial coefficients by

$$\binom{n-3/2}{n} = -\frac{1}{2\pi^{1/2}n^{3/2}}\left(1+O\left(\frac{1}{n}\right)\right)$$

and

$$\binom{n-5/2}{n} = O\left(\frac{1}{n^{5/2}}\right).$$

Thus we have

$$\begin{aligned} &\frac{1}{2\pi}\int_{J^+}[z^n]C_\vartheta(z)\,d\vartheta \\ &\qquad = \frac{e^n}{(2\pi)^{1/2}n^{3/2}}\left(1+O\left(\frac{(\log n)^2}{n}\right)\right)\int_{-\varepsilon(n)}^{\varepsilon(n)}\exp -(n\vartheta^2/8)\,d\vartheta \\ &\qquad = \frac{e^n}{\pi n^2}\left(1+O\left(\frac{(\log n)^2}{n}\right)\right), \end{aligned}$$

where we have evaluated the integral by making the change of variable $\vartheta = 2\xi/n^{1/2}$ to obtain

$$\begin{aligned} \exp -(n\vartheta^2/8)\,d\vartheta &= \frac{2}{n^{1/2}}\int_{-\delta(n)}^{\delta(n)}\exp -(\xi^2/2)\,d\xi \\ &= \frac{2}{n^{1/2}}\int_{-\infty}^{\infty}\exp -(\xi^2/2)\,d\xi \\ &\qquad - \frac{2}{n^{1/2}}\int_{-\infty}^{-\delta(n)}\exp -(\xi^2/2)\,d\xi \\ &\qquad - \frac{2}{n^{1/2}}\int_{\delta(n)}^{\infty}\exp -(\xi^2/2)\,d\xi \\ &= \frac{2^{3/2}\pi^{1/2}}{n^{1/2}} + O\left(\frac{1}{n^{12}(\log n)^{1/2}}\right), \end{aligned}$$

where $\delta(n) = (24\log n)^{1/2}$.

For ϑ in J^-, similar arguments yield

$$\frac{1}{2\pi}\int_{J^+}[z^n]C_\vartheta(z)\,d\vartheta = \pm\frac{e^n}{\pi n^2}\left(1+O\left(\frac{(\log n)^2}{n}\right)\right),$$

where the plus sign is taken for n even and the minus sign for n odd. (The alternation of sign arises from the negative branch point Z_ϑ^- being raised to the power n.) Combining these estimates, we conclude that

$$[z^n]R^*(z) = \frac{2e^n}{\pi n^2}\left(1 + O\left(\frac{(\log n)^2}{n}\right)\right)$$

for even n. For odd n we know that $[z^n]R^*(z) = 0$, although this asymptotic analysis yields only $[z^n]R^*(z) = O\bigl(e^n(\log n)^2/n^3\bigr)$. Since $R_{m,m} = m!^2[z^{2m}]R^*(z)$ and $m!^2 = 2\pi m^{2m+1}e^{-2m}\bigl(1 + O(1/m)\bigr)$, we conclude that $R_{m,m} = m^{2m-1}\bigl(1 + O(1/m)\bigr)$, which is consistent with the exact result cited above. We observe that the limiting value, as n tends to infinity through even values, of the ratio of R_n^*/R_n (the probability that a randomly chosen n-vertex labelled tree is equicolorable) to $\binom{n}{n/2}/2^n \sim (2/\pi n)^{1/2}$ (the probability that n vertices, independently assigned colors by unbiased coin flips, are equicolored) is 2.

3. Rooted trees. The problem of enumerating rooted unlabelled trees was first broached by Cayley [C1] in 1857. The problem is to determine the number r_n of different rooted trees on n vertices, where two trees are to be considered the same if there is an isomorphism between them (that is, a one-to-one correspondence between the vertices that preserves the root, as well as the adjacency relation). Cayley did not quite give either a recurrence or a functional equation for the generating function for these trees but rather gave a curious amalgam of the two that allows the number of rooted trees to be calculated expeditiously.

It was Pólya [P1, P2] who in 1937 first gave an enumeration of rooted trees entirely in terms of the generating function

$$r(z) = \sum_{n\ge1} r_n\, z^n,$$

and it is his path that we shall follow and extend in our work. Note that, as is customary when enumerating unlabelled objects, $r(z)$ is an "ordinary," rather than an "exponential," generating function.

Pólya's first step was to formulate a component principle analogous to (2.1) for ordinary generating functions enumerating unlabelled objects. This principle states that if $f(z)$ is the ordinary generating function for unlabelled components, then

$$g(z) = \exp\sum_{h\ge1}\frac{f(z^h)}{h} \tag{3.1}$$

is the ordinary generating function for unlabelled structures comprising zero or more disjoint components. Since a rooted tree comprises a root together with zero or more disjoint rooted trees (the subtrees adjacent to the root), $r(z)$ satisfies the functional equation

$$r(z) = z\exp\sum_{h\ge1}\frac{r(z^h)}{h}. \tag{3.2}$$

Note that this functional equation is "nonlocal," in that the right-hand side involves the evaluation of r not only at z but at its powers $z^2, z^3, \dots$ as well.

That the asymptotic methods used for labelled trees in section 2 (based on Darboux's lemma) can also be applied to (3.2) was indicated by Pólya and carried out

explicitly by Otter [O]. The first step is to find the singularity of $r(z)$ that is closest to the origin; this corresponds to the radius of convergence z_0 of $r(z)$. Since an unlabelled rooted tree on n vertices has at most $n!$ different labellings, the coefficients of $r(z)$ are greater than or equal to the corresponding coefficients of $R(z)$, and thus $z_0 \le Z_0 = 1/e$. On the other hand, each unlabelled rooted tree corresponds to at least one unlabelled ordered rooted tree (in which the offspring of each vertex are linearly ordered). The latter were enumerated by Cayley [C2], who showed that the number of such trees with n vertices is $\frac{1}{n}\binom{2n-2}{n-1} \le 4^{n-1}$. Thus the coefficients of $r(z)$ are less than the corresponding coefficients of $z/(1-4z)$, so that $z_0 \ge 1/4$.

To find the singularity z_0 more precisely, we write (3.2) as $\Phi\big(z, r(z)\big) = 0$, where

$$\Phi(z,w) = z\exp\big(w + \Psi(z)\big) - w$$

and

$$\Psi(z) = \sum_{h\ge 2} \frac{r(z^h)}{h}.$$

We observe that since $r(z)$ is analytic for z in the disk of radius $1/4$ centered at the origin, $\Psi(z)$, and thus also $\Phi(z,w)$, is analytic for z in the disk of radius $(1/4)^{1/2} = 1/2 > 1/e \ge z_0$ centered at the origin. To locate the singularity, we calculate

$$\frac{\partial}{\partial w}\Phi(z,w) = \Phi(z,w) + w - 1.$$

The singularity occurs when this derivative and $\Phi(z,w)$ vanish simultaneously for $w = r(z)$. This happens only for $w = w_0 = r(z_0) = 1$. Thus z_0 satisfies the equation

$$z_0 = \exp -\big(1 + \Psi(z_0)\big).$$

To determine the numerical value $z_0 = 0.3383\ldots$, we use the formula

$$\begin{aligned}\Psi(z) &= \sum_{h\ge 2}\frac{1}{h}\sum_{n\ge 1} r_n\, z^{nh} \\ &= \sum_{n\ge 1} r_n \left(\log\frac{1}{1-z^n} - z^n\right),\end{aligned}$$

together with the coefficients r_n of the series $r(z)$, which can be calculated recursively from (3.2) (see Table 1).

To expand $r(z)$ in the neighborhood of $z = z_0$, we calculate

$$\frac{\partial^2}{\partial w^2}\Phi(z,w) = \frac{\partial}{\partial w}\Phi(z,w) + 1 = \Phi(z,w) + w \tag{3.2$'$}$$

and

$$\frac{\partial}{\partial z}\Phi(z,w) = \big(\Phi(z,w) + w\big)\big(1 + z\Psi'(z)\big)/z. \tag{3.2$''$}$$

Then we have

$$\frac{\partial^2}{\partial w^2}\Phi(z,w)\Big|_{w=w_0, z=z_0} = 1$$

and

$$\frac{\partial}{\partial z}\Phi(z,w)\big|_{w=w_0,z=z_0} = \big(1 + z_0\Psi'(z_0)\big)/z_0,$$

so that

$$\begin{aligned}\Phi(z,w) = \frac{1}{2}(w-w_0)^2 + O\big((w-w_0)^3\big)\\ - A\,(1-z/z_0) + O\big((w-w_0)(1-z/z_0)\big) + O\big((1-z/z_0)^2\big),\end{aligned}$$

where $A = 1 + z_0\Psi'(z_0)$. To determine the numerical value $A = 1.215\ldots$, we use the formula

$$\begin{aligned}z\Psi'(z) &= \sum_{h\geq 2}\sum_{n\geq 1} nr_n\, z^{nh}\\ &= \sum_{n\geq 1} nr_n \left(\frac{z^n}{1-z^n} - z^n\right).\end{aligned}$$

Thus at $z = z_0$, $r(z)$ has a branch point of order 2 and an expansion of the form

$$r(z) = a(z) + b(z)(1-z/z_0)^{1/2},$$

where $a(z) = 1 + O(z-z_0)$ and $b(z) = -(2A)^{1/2} + O(z-z_0)$ are analytic functions of z. Applying Darboux's lemma, we conclude that $[z^n]r(z)$ is asymptotic to $A^{1/2}z_0^{-n}/n^{3/2}(2\pi)^{1/2}$, where $(A/2\pi)^{1/2} = 0.4399\ldots$.

We now turn to the problem of enumerating equicolorable unlabelled rooted trees. Let $r_{l,m}$ denote the number of red-rooted unlabelled trees with $l \geq 1$ red vertices and $m \geq 0$ blue vertices. Let

$$r(x,y) = \sum_{l\geq 1, m\geq 0} r_{l,m}\, x^l y^m$$

be the bivariate ordinary generating function for red-rooted unlabelled trees. The component principle analogous to (3.1) for bivariate ordinary generating functions is

$$g(x,y) = \exp\sum_{h\geq 1}\frac{f(x^h,y^h)}{h},$$

where $f(x,y)$ is the generating function for components and $g(x,y)$ is the generating function for structures comprising zero or more disjoint components. Since a red-rooted tree comprises a red root (enumerated by x), together with zero or more disjoint blue-rooted trees (enumerated by $r(y,x)$), $r(x,y)$ satisfies the functional equation

$$r(x,y) = x\exp\sum_{h\geq 1}\frac{r(y^h,x^h)}{h}. \tag{3.3}$$

We shall derive from this functional equation the asymptotic behavior of the coefficients $r_{m,m}$.

We begin by making the substitutions $x = z\exp(i\vartheta)$ and $y = z\exp(-i\vartheta)$ and thus defining

$$r_\vartheta(z) = r\big(z\exp(i\vartheta), z\exp(-i\vartheta)\big). \tag{3.4}$$

From (3.3) and (3.4) we obtain

$$r_\vartheta(z) = z\exp\left(i\vartheta + \sum_{h\ge1}\frac{r_{-h\vartheta}(z^h)}{h}\right) \tag{3.5}$$

as the functional equation satisfied by $r_\vartheta(z)$.

As before, it will be convenient to work with relatives of $r_\vartheta(z)$ that are real when ϑ and z are real. Thus we define

$$c_\vartheta(z) = \frac{r_\vartheta(z) + r_{-\vartheta}(z)}{2} \tag{3.6}$$

and

$$s_\vartheta(z) = \frac{r_\vartheta(z) - r_{-\vartheta}(z)}{2i}. \tag{3.7}$$

We can find the functional equations satisfied by $c_\vartheta(z)$ and $s_\vartheta(z)$ by substituting (3.5) into (3.6) and (3.7), then substituting $r_\vartheta(z) = c_\vartheta(z) + i\,s_\vartheta(z)$ and $r_{-\vartheta(z)} = c_\vartheta(z) - i\,s_\vartheta(z)$ into the result to obtain

$$c_\vartheta(z) = z\exp\left(\sum_{h\ge1}\frac{c_{h\vartheta}(z^h)}{h}\right)\cos\left(\vartheta - \sum_{h\ge1}\frac{s_{h\vartheta}(z^h)}{h}\right) \tag{3.8}$$

and

$$s_\vartheta(z) = z\exp\left(\sum_{h\ge1}\frac{c_{h\vartheta}(z^h)}{h}\right)\sin\left(\vartheta - \sum_{h\ge1}\frac{s_{h\vartheta}(z^h)}{h}\right). \tag{3.9}$$

To determine the singularities of $c_\vartheta(z)$ as a function of z with ϑ fixed, we eliminate $s_\vartheta(z)$ from (3.8) and (3.9). Squaring and adding these equations, we obtain

$$c_\vartheta(z)^2 + s_\vartheta(z)^2 = z^2\exp\bigl(2c_\vartheta(z) + 2\Psi_\vartheta(z)\bigr),$$

where

$$\Psi_\vartheta(z) = \sum_{h\ge2}\frac{c_{h\vartheta}(z^h)}{h}.$$

This result allows us to eliminate $s_\vartheta(z)$ from (3.8), obtaining

$$\begin{aligned} c_\vartheta(z) = {}& z\exp\bigl(c_\vartheta(z) + \Psi_\vartheta(z)\bigr) \\ & \times\cos\Bigl(\vartheta - \bigl(z^2\exp\bigl(2c_\vartheta(z) + 2\Psi_\vartheta(z)\bigr) - c_\vartheta(z)^2\bigr)^{1/2} - \Upsilon_\vartheta(z)\Bigr), \end{aligned}$$

where

$$\Upsilon_\vartheta(z) = \sum_{h\ge2}\frac{s_{h\vartheta}(z^h)}{h}.$$

This equation can be written as $\Phi_\vartheta\bigl(z, c_\vartheta(z)\bigr) = 0$, where

$$\Phi_\vartheta(z,w) = z\exp\bigl(w + \Psi_\vartheta(z)\bigr)\cos\Bigl(\vartheta - \bigl(z^2\exp\bigl(2w + 2\Psi_\vartheta(z)\bigr) - w^2\bigr)^{1/2} - \Upsilon_\vartheta(z)\Bigr) - w.$$

To locate the singularities of $c_\vartheta(z)$, we calculate

$$\frac{\partial}{\partial w}\Phi_\vartheta(z,w) = \Phi_\vartheta(z,w) - 1 + z^2 \exp\bigl(2w + 2\Psi_\vartheta(z)\bigr).$$

The singularities occur when this derivative and $\Phi_\vartheta(z,w)$ vanish simultaneously for $z = z_\vartheta^\pm$ and $w = c_\vartheta(z_\vartheta^\pm)$, so that we have

$$z_\vartheta^\pm = \pm \exp -\bigl(c_\vartheta(z_\vartheta^\pm) + \Psi_\vartheta(z_\vartheta^\pm)\bigr). \tag{3.10}$$

Substituting this relation into (3.8) and (3.9) yields

$$c_\vartheta(z_\vartheta^\pm) = \pm \cos\bigl(\vartheta - s_\vartheta(z_\vartheta^\pm) - \Upsilon_\vartheta(z_\vartheta^\pm)\bigr),$$

and

$$s_\vartheta(z_\vartheta^\pm) = \pm \sin\bigl(\vartheta - s_\vartheta(z_\vartheta^\pm) - \Upsilon_\vartheta(z_\vartheta^\pm)\bigr),$$

where, of course, we must take the same sign throughout all three equations. We can again express the solutions to these equations in terms of the cycloid function

$$z_\vartheta^+ = \exp -\bigl(\mathrm{cyc}\bigl(\vartheta - \Upsilon_\vartheta(z_\vartheta^+)\bigr) + \Psi_\vartheta(z_\vartheta^+)\bigr) \tag{3.11}$$

and

$$z_\vartheta^- = -\exp -\bigl(\mathrm{cyc}\bigl(\vartheta - \pi - \Upsilon_\vartheta(z_\vartheta^-)\bigr) + \Psi_\vartheta(z_\vartheta^-)\bigr).$$

To expand $c_\vartheta(z)$ in the neighborhood of $z = z_\vartheta^+$, we calculate

$$\begin{aligned}\frac{\partial^2}{\partial w^2}\Phi_\vartheta(z,w) &= \frac{\partial}{\partial w}\Phi_\vartheta(z,w) + 2z^2 \exp\bigl(2w + 2\Psi_\vartheta(z)\bigr) \\ &= \Phi_\vartheta(z,w) - 1 + 3z^2 \exp\bigl(2w + 2\Psi_\vartheta(z)\bigr)\end{aligned}$$

and

$$\begin{aligned}\frac{\partial}{\partial z}\Phi_\vartheta(z,w) = {}& \bigl(\Phi_\vartheta(z,w) + w\bigr)\bigl(1 + z\Psi'_\vartheta(z)\bigr)/z \\ &+ z^2 \exp\bigl(3w + 3\Psi_\vartheta(z)\bigr)\bigl(1 + z\Psi'_\vartheta(z)\bigr) \\ &+ z \exp\bigl(w + \Psi_\vartheta(z)\bigr)\bigl(z^2 \exp\bigl(2w + 2\Psi_\vartheta(z)\bigr) - w^2\bigr)^{1/2}\ \Upsilon'_\vartheta(z).\end{aligned}$$

Then we have

$$\frac{\partial^2}{\partial w^2}\Phi_\vartheta(z,w)\big|_{w=w_\vartheta^+, z=z_\vartheta^+} = 2$$

and

$$\frac{\partial}{\partial z}\Phi_\vartheta(z,w)\big|_{w=w_\vartheta^+, z=z_\vartheta^+} = \frac{(1+w_\vartheta^+)\bigl(1 + z_\vartheta^+\Psi'(z_\vartheta^+)\bigr) + \bigl(1 - (w_\vartheta^+)^2\bigr)^{1/2} z_\vartheta^+ \Upsilon'_\vartheta(z_\vartheta^+)}{z_\vartheta^+},$$

so that

$$\begin{aligned}\Phi_\vartheta(z,w) = {}& (w - w_\vartheta^+)^2 + O\bigl((w - w_\vartheta^+)^3\bigr) \\ &- A_\vartheta^+ (1 - z/z_\vartheta^+) + O\bigl((w - w_\vartheta^+)(1 - z/z_\vartheta^+)\bigr) + O\bigl((1 - z/z_\vartheta^+)^2\bigr),\end{aligned}$$

where

$$\begin{aligned} A_\vartheta^+ &= (1+w_\vartheta^+)\bigl(1+z_\vartheta^+\Psi'(z_\vartheta^+)\bigr) + \bigl(1-(w_\vartheta^+)^2\bigr)^{1/2} z_\vartheta^+\Upsilon'_\vartheta(z_\vartheta^+) \\ &= \bigl(1+\operatorname{cyc}\bigl(\vartheta-\Upsilon_\vartheta(z_\vartheta^+)\bigr)\bigr)\bigl(1+z_\vartheta^+\Psi'(z_\vartheta^+)\bigr) + \operatorname{cocyc}\bigl(\vartheta-\Upsilon_\vartheta(z_\vartheta^+)\bigr)\, z_\vartheta^+\Upsilon'_\vartheta(z_\vartheta^+). \end{aligned}$$

Thus at $z = z_\vartheta^+$, $c_\vartheta(z)$ has a branch point of order 2 and, in the neighborhood of $z = z_\vartheta^+$, an expansion of the form

$$c_\vartheta^+(z) = a_\vartheta^+(z) + b_\vartheta^+(z)(1-z/z_\vartheta^+)^{1/2},$$

where $a_\vartheta^+(z) = \operatorname{cyc}\bigl(\vartheta-\Upsilon_\vartheta(z_\vartheta^+)\bigr) + O(z-z_\vartheta^+)$ and $b_\vartheta^+(z) = -(A_\vartheta^+)^{1/2} + O(z-z_\vartheta^+)$ are analytic functions of z, and where again the constants in the O-terms are uniform in ϑ. Similar arguments give, in the neighborhood of $z = z_\vartheta^-$, an expansion of the form

$$c_\vartheta^-(z) = a_\vartheta^-(z) + b_\vartheta^-(z)(1-z/z_\vartheta^-)^{1/2},$$

where $a_\vartheta^-(z) = \operatorname{cyc}\bigl(\vartheta-\pi-\Upsilon_\vartheta(z_\vartheta^-)\bigr) + O(z-z_\vartheta^-)$, $b_\vartheta^-(z) = -(A_\vartheta^-)^{1/2} + O(z-z_\vartheta^-)$ and

$$\begin{aligned} A_\vartheta^- &= (-1+w_\vartheta^-)\bigl(1+z_\vartheta^-\Psi'(z_\vartheta^-)\bigr) - \bigl(1-(w_\vartheta^-)^2\bigr)^{1/2} z_\vartheta^-\Upsilon'_\vartheta(z_\vartheta^-) \\ &= \bigl(-1+\operatorname{cyc}\bigl(\vartheta-\pi-\Upsilon_\vartheta(z_\vartheta^-)\bigr)\bigr)\bigl(1+z_\vartheta^-\Psi'(z_\vartheta^-)\bigr) \\ &\qquad - \operatorname{cocyc}\bigl(\vartheta-\pi-\Upsilon_\vartheta(z_\vartheta^-)\bigr)\, z_\vartheta^-\Upsilon'_\vartheta(z_\vartheta^-). \end{aligned}$$

We are now ready to extract the desired asymptotic information from these expansions for $c_\vartheta(z)$. We define

$$r^*(z) = \frac{1}{2\pi}\int_{-\pi/2}^{3\pi/2} c_\vartheta(z)\,d\vartheta,$$

a power series in z in which the coefficients of odd powers of z vanish and the coefficient of the even power z^{2m} is the same as the coefficient of the term $x^m y^m$ in $r(x,y)$. Thus we have

$$r_{m,m} = \frac{1}{2\pi}\int_{-\pi/2}^{3\pi/2} [z^{2m}]\, c_\vartheta(z)\,d\vartheta. \tag{3.12}$$

The estimation of this integral is completely analogous to that in section 2. The only differences are in the locations of the singularities $z_\vartheta^\pm$ and in the constant terms of the functions $a_\vartheta^\pm$ and $b_\vartheta^\pm$. Furthermore, these values affect the leading term of the asymptotics only through their dependence on ϑ in the neighborhoods of $\vartheta = 0$ (for the plus superscript) and $\vartheta = \pi$ (for the minus superscript). We begin with the plus superscript. Simple arguments show that z_ϑ^+ is an even analytic function of ϑ, and $z_0^+ = z_0$, as in the univariate case. Thus in the neighborhood of $\vartheta = 0$ we have

$$z_\vartheta^+ = z_0\left(1 + \frac{\ddot{z}_0^+}{2z_0}\,\vartheta^2 + O(\vartheta^4)\right),$$

where dots indicate differentiation with respect to the subscript (as opposed to primes, which indicate differentiation with respect to a parenthesized argument). To determine $\ddot{z}_0^+$, we use (3.11). For the cycloid function, we have the expansion $\operatorname{cyc}\vartheta = 1 - \vartheta^2/8 + O(\vartheta^4)$ in the neighborhood of $\vartheta = 0$. The function $\Upsilon_\vartheta(z)$ is an odd

analytic function of ϑ, so we have $\Upsilon_\vartheta(z_\vartheta^+) = \dot\Upsilon_0(z_0^+)\vartheta + O(\vartheta^3)$ in the neighborhood of $\vartheta = 0$. And the function $\Psi_\vartheta(z)$ is an even analytic function of ϑ, so we have $\Psi_\vartheta(z_\vartheta^+) = \Psi_0(z_0^+) + \bigl(\ddot\Psi_0(z_0^+) + \Psi_0'(z_0^+)\,\ddot z_0^+\bigr)\vartheta^2/2 + O(\vartheta^4)$ in the neighborhood of $\vartheta = 0$. Combining these results with (3.11) yields $\ddot z_0^+/2z_0 = (B^2 - 4C)/8A$, where $B = 1 - \dot\Upsilon_0(z_0^+)$ and $C = \ddot\Psi_0(z_0^+)$, so that

$$z_\vartheta^+ = z_0 \left(1 + \frac{B^2 - 4C}{8A}\,\vartheta^2 + O(\vartheta^4)\right).$$

The constant terms of $a_\vartheta^+(z) = 1 + O(\vartheta^2) + O(z - z_\vartheta^+)$ and $b_\vartheta^+(z) = -(2A)^{1/2} + O(\vartheta^2) + O(z - z_\vartheta^+)$ are the same as in the univariate case. For the minus superscript, similar calculations yield

$$z_\vartheta^- = z_0 \left(1 + \frac{B^2 - 4C}{8A}\,\vartheta^2 + O(\vartheta^4)\right),$$

$a_\vartheta^-(z) = 1 + O(\vartheta^2) + O(z - z_\vartheta^-)$, and $b_\vartheta^-(z) = -(2A)^{1/2} + O(\vartheta^2) + O(z - z_\vartheta^-)$. With these expansions, we can estimate (3.12) as in section 2 to obtain

$$[z^n]r^*(z) = \frac{2A\,z_0^{-n}}{\pi(B^2 - 4C)^{1/2}\,n^2} \left(1 + O\left(\frac{(\log n)^2}{n}\right)\right) \tag{3.13}$$

for even n. For odd n we know that $[z^n]r^*(z) = 0$.

It remains to determine the numerical values of the constants in (3.13). For B, we start with

$$\dot s_0(z) = \sum_{l\ge1,m\ge0} (l - m) r_{l,m}\, z^{l+m} = q(z),$$

where

$$q(z) = \left(\left(x\frac{\partial}{\partial x} - y\frac{\partial}{\partial y}\right) r(x,y)\right)\Bigg|_{x=z,y=z}.$$

The coefficients of the series $q(z) = \sum_{n\ge1} q_n\, z^n$ can be calculated from the coefficients $r_{l,m}$, which can in turn be calculated recursively from (3.3) (see Table 1). To determine the numerical value $B = 1 - \dot\Upsilon_0(z_0) = 0.8269\ldots$, we use the formula

$$\begin{aligned}\dot\Upsilon_0(z) &= \sum_{h\ge2} \dot s_0(z) \\ &= \sum_{h\ge2} \sum_{n\ge1} q_n\, z^{nh} \\ &= \sum_{n\ge1} q_n \left(\frac{z^n}{1 - z^n} - z^n\right).\end{aligned}$$

For C, we start with

$$\ddot c_0(z) = -\sum_{l\ge1,m\ge0} (l - m)^2 r_{l,m}\, z^{l+m} = -p(z),$$

where

$$p(z) = \left(\left(x\frac{\partial}{\partial x} - y\frac{\partial}{\partial y}\right)^2 r(x,y)\right)\Bigg|_{x=z,y=z}.$$

TABLE 1
Coefficients in the series $r(z)$, $q(z)$, $p(z)$, and $r^*(z)$.

n	r_n	q_n	p_n	r_n^*
1	1	1	1	0
2	1	0	0	1
3	2	0	2	0
4	4	0	8	2
5	9	1	25	0
6	20	2	68	9
7	48	8	192	0
8	115	18	516	44
9	286	52	1438	0
10	719	130	3964	249
11	1842	348	11098	0
12	4766	904	31056	1506
13	12486	2416	87694	0
14	32973	6404	247960	9687
15	87811	17213	704571	0

The coefficients of the series $p(z) = \sum_{n\geq 1} p_n z^n$ can also be calculated from the coefficients $r_{l,m}$ (see Table 1). To determine the numerical value $C = \ddot{\Psi}_0(z_0) = -0.4450\ldots$, we use the formula

$$\begin{aligned}\ddot{\Psi}_0(z) &= \sum_{h\geq 2} h\,\ddot{c}_0(z)\\ &= -\sum_{h\geq 2} h \sum_{n\geq 1} p_n\, z^{nh}\\ &= -\sum_{n\geq 1} p_n \left(\frac{z^n}{(1-z^n)^2} - z^n\right).\end{aligned}$$

Combining these results gives $2A/\pi(B^2-4C)^{1/2} = 0.4931\ldots$ for the constant appearing in (3.13). We observe that the limiting value, as n tends to infinity through even values, of the ratio of r_n^*/r_n (the probability that a randomly chosen n-vertex rooted tree is equicolorable) to $\binom{n}{n/2}/2^n \sim (2/\pi n)^{1/2}$ (the probability that n vertices, independently assigned colors by unbiased coin flips, are equicolored) is $2A^{1/2}/(B^2-4C)^{1/2} = 1.40499\ldots$.

4. Unrooted trees. The enumeration of unrooted unlabelled trees was first undertaken by Cayley, who in 1875 [C3] gave it in terms of a two-parameter enumeration of rooted trees by size and depth. In 1881 [C4], he expressed the numbers u_n of unrooted trees exclusively in terms of the numbers r_n of rooted trees. In 1948, Otter [O] expressed the generating function

$$u(z) = \sum_{n\geq 1} u_n\, z^n$$

for unrooted trees in terms of the generating function

$$r(z) = \sum_{n\geq 1} r_n\, z^n$$

for rooted trees,

$$u(z) = r(z) - \frac{1}{2}r(z)^2 + \frac{1}{2}r(z^2), \tag{4.1}$$

and from this he was able to deduce the asymptotic behavior,

$$u_n \sim \frac{A^{3/2}}{(2\pi)^{1/2}} \frac{z_0^{-n}}{n^{5/2}}, \tag{4.2}$$

where $A = 1.215\ldots$ and $z_0 = 0.3383\ldots$ are as defined in section 3.

If T is an unrooted tree, let v_T denote the number of orbits of its vertices under the action of its automorphism group, let e_T denote the number of orbits of edges, and let s_T denote 1 or 0 depending on whether or not T is *edge-symmetric*, that is, depending on whether or not there is an automorphism of T that exchanges the vertices of some edge of T. Otter established the identity

$$1 = v_T - e_T + s_T. \tag{4.3}$$

If n_T denotes the number of vertices in T, then multiplying (4.3) by z^{n_T} and summing over all unrooted trees T yields for the left-hand side the generating function $u(z)$ for unrooted trees. Since the unrooted tree T can be rooted in v_T different ways, the sum of $v_T\, z^{n_T}$ yields $r(z)$. Similarly, the sum of $e_T\, z^{n_T}$ yields the generating function for trees rooted at an edge rather than a vertex; this is easily seen to be $\frac{1}{2}\big(r(z)^2 + r(z^2)\big)$. Finally, the sum of $s_T\, z^{n_T}$ is the generating function for edge-symmetric trees; this is easily seen to be $\frac{1}{2}r(z^2)$. Combining these results yields Otter's identity (4.1).

To derive the asymptotic behavior (4.2), we again apply Darboux's lemma to the singularity of $u(z)$ that is closest to the origin. This singularity is at z_0, and it arises from the contributions of $r(z)$ and $-\frac{1}{2}r(z)^2$. The term $\frac{1}{2}r(z^2)$ has no singularity closer to the origin than $z_0^{1/2} > z_0$ and thus makes a negligible contribution to the asymptotic behavior. With an eye to what is to come, we shall define the generating function $h(z) = \sum_{n\geq 1} h_n\, z^n$ by

$$h(z) = 2r(z) - r(z)^2, \tag{4.3$'$}$$

so that $u(z) = \frac{1}{2}h(z) + \frac{1}{2}r(z^2)$. We shall show that

$$h_n \sim \frac{2A^{3/2}}{(2\pi)^{1/2}} \frac{z_0^{-n}}{n^{5/2}}, \tag{4.3$''$}$$

which implies (4.2).

To expand $u(z)$ in the neighborhood of $z = z_0$, we must extend the expansion of $r(z)$, obtained in section 3, to higher terms. We have

$$r(z) = a(z) + b(z)(1 - z/z_0)^{1/2}, \tag{4.4}$$

where $a(z) = \sum_{k\geq 0} a_k\,(1 - z/z_0)^k$ and $b(z) = \sum_{k\geq 0} b_k\,(1 - z/z_0)^k$ are analytic at $z = z_0$. We have seen that $a_0 = 1$ and $b_0 = -(2A)^{1/2}$. We shall show now that $a_1 = 2A/3$.

Continuing from (3.2$'$) we have

$$\frac{\partial^3}{\partial w^3}\Phi(z,w) = \frac{\partial}{\partial w}\Phi(z,w) + 1 = \Phi(z,w) + w,$$

so that

$$\frac{\partial^3}{\partial w^3}\Phi(z,w)\Big|_{w=w_0,z=z_0} = 1.$$

Continuing from (3.2″) we have

$$\frac{\partial^2}{\partial w \partial z}\Phi(z,w) = \big(\Phi(z,w)+w\big)\big(1+z\Psi'(z)\big)/z,$$

so that

$$\frac{\partial^2}{\partial w \partial z}\Phi(z,w)\big|_{w=w_0,z=z_0} = \big(1+z_0\Psi'(z_0)\big)/z_0.$$

Combining these results yields

$$\begin{aligned}\Phi(z,w) = \frac{1}{2}(w-w_0)^2 + \frac{1}{6}(w-w_0)^3 + O\big((w-w_0)^4\big)\\ - A\,(1-z/z_0) - A\,(w-w_0)(1-z/z_0)\\ + O\big((w-w_0)^2(1-z/z_0)\big) + O\big((1-z/z_0)^2\big),\end{aligned}$$

where as before $A = 1+z_0\Psi'(z_0)$. Since we have $\Phi\big(z,r(z)\big)=0$, this expansion implies (4.4) with $a_0=1$, $b_0=-(2A)^{1/2}$, and $a_1=2A/3$. Substituting this expansion into the right-hand side of (4.3′) yields that

$$h(z) = f(z) + g(z)(1-z/z_0)^{1/2},$$

where $f(z)$ and $g(z) = \sum_{k\ge 0} g_k\,(1-z/z_0)^k$ are analytic at $z=z_0$, $g_0=0$, and $g_1 = 2(2A)^{3/2}/3$. Applying Darboux's lemma to the singularity of $h(z)$ at $z=z_0$ yields (4.3″) and thus Otter's asymptotic formula (4.2).

To enumerate equicolorable unrooted trees, our first task is to find an analogue of Otter's identity (4.1). Let T be an unrooted tree. If one bicoloring of T has a_T red and b_T blue vertices, then the other bicoloring has b_T red and a_T blue vertices. Thus the polynomial $\frac{1}{2}(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T})$ depends only on T and not on the particular bicoloring considered. We define

$$u(x,y) = \frac{1}{2}\sum_T x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T},$$

where the sum is over all unrooted trees. Our goal is to establish the identity

$$u(x,y) = \frac{1}{2}r(x,y) + \frac{1}{2}r(y,x) - \frac{1}{2}r(x,y)\,r(y,x) + \frac{1}{2}r(xy), \tag{4.5}$$

analogous to (4.1).

Let S be a bicolored unrooted tree, and let a_S and b_S denote the numbers of red and blue vertices, respectively, in S. We define

$$h(x,y) = \sum_S x^{a_S}\,y^{b_S},$$

where the sum is over all bicolored unrooted trees. An unrooted tree has two distinct bicolorings unless it is edge-symmetric, in which case it has just one. This yields

$$u(x,y) = \frac{1}{2}h(x,y) + \frac{1}{2}r(xy),$$

since $r(xy)$ enumerates bicolored edge-symmetric unrooted trees. Thus to establish (4.5) it will suffice to show that

$$h(x,y) = r(x,y) + r(y,x) - r(x,y)\,r(y,x). \tag{4.6}$$

Again using the fact that an unrooted tree has one or two bicolorings depending on whether or not it is edge-symmetric, we have

$$h(x,y) = \frac{1}{2}\sum_T (2 - s_T)(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T}).$$

From (4.3) we have $2 - s_T = 2v_T - 2e_T + s_T$, so that

$$h(x,y) = \sum_T v_T\,(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T}) - \frac{1}{2}\sum_T (2e_T - s_T)(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T}). \tag{4.7}$$

Since an unrooted tree T can be rooted in v_T different ways, we have

$$\sum_T v_T\,(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T}) = r(x,y) + r(y,x).$$

Let d_T denote the number of different ways in which T can be rooted in a directed edge. Then $d_T = 2e_T - s_T$. Thus we have

$$\begin{aligned}\sum_T (2e_T - s_T)(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T}) &= \sum_T d_T\,(x^{a_T}\,y^{b_T} + x^{b_T}\,y^{a_T}) \\ &= 2r(x,y)\,r(y,x),\end{aligned}$$

since each directed-edge-rooted tree can be decomposed in a unique way into a red-rooted tree whose root is the source of a directed edge whose target is the root of a blue-rooted tree, or into a blue-rooted tree whose root is the source of a directed edge whose target is the root of a red-rooted tree. Substituting these results into (4.7) yields (4.6) and thus (4.5).

At this point we can express the generating functions $u^*(z) = \sum_{n\geq 1} u_n^*\,z^n$ and $h^*(z) = \sum_{n\geq 1} h_n^*\,z^n$ for equicolorable and equicolored unrooted trees, respectively, as

$$u^*(z) = \frac{1}{4\pi}\int_{-\pi/2}^{3\pi/2} r_\vartheta(z) + r_{-\vartheta}(z) - r_\vartheta(z)\,r_{-\vartheta}(z) + r(z^2)\,d\vartheta \tag{4.8}$$

and

$$h^*(z) = \frac{1}{2\pi}\int_{-\pi/2}^{3\pi/2} r_\vartheta(z) + r_{-\vartheta}(z) - r_\vartheta(z)\,r_{-\vartheta}(z)\,d\vartheta. \tag{4.9}$$

The coefficients of these generating functions, together with those of $u(z) = \sum_{n\geq 1} u_n z^n$ for unrooted trees, are tabulated in Table 2.

To determine the asymptotic behavior of the coefficients u_n^* and h_n^*, we shall apply Darboux's lemma to (4.8) and (4.9). It will suffice to deal with (4.9), since (4.8) differs merely by a factor of 2 and the additional term $r(z^2)$, which (having no singularity closer to the origin than $z_0^{1/2} > z_0$) makes an asymptotically negligible contribution. To deal with (4.9), we define $h_\vartheta(z)$ to be the integrand,

$$h_\vartheta(z) = r_\vartheta(z) + r_{-\vartheta}(z) - r_\vartheta(z)\,r_{-\vartheta}(z),$$

TABLE 2
Coefficients in the series $u(z)$, $u^*(z)$, and $h^*(z)$.

n	u_n	u_n^*	h_n^*
1	1	0	0
2	1	1	1
3	1	0	0
4	2	1	1
5	3	0	0
6	6	3	4
7	11	0	0
8	23	9	14
9	47	0	0
10	106	37	65
11	235	0	0
12	551	168	316
13	1301	0	0
14	3159	895	1742
15	7741	0	0

which (using (3.6) and (3.5)) we can rewrite as

$$h_\vartheta(z) = 2c_\vartheta(z) - z^2 \exp\bigl(2c_\vartheta(z) + 2\Psi_\vartheta(z)\bigr). \tag{4.10}$$

From (4.10), we see that the singularities of $h_\vartheta(z)$ closest to the origin are, just as for $c_\vartheta(z)$, at z_ϑ^+ and z_ϑ^-. Starting with the singularity at z_ϑ^+, we seek to expand $h_\vartheta(z)$ in a neighborhood of z_ϑ^+ as

$$h_\vartheta^+(z) = f_\vartheta^+(z) + g_\vartheta^+(z)\,(1 - z/z_\vartheta^+)^{1/2},$$

where $f_\vartheta^+(z)$ and $g_\vartheta^+(z)$ are analytic at $z = z_\vartheta^+$. Let us expand $g_\vartheta^+(z)$ as $g_\vartheta^+(z) = \sum_{k\ge 0} g_{\vartheta,k}^+\,(1 - z/z_\vartheta^+)^k$.

We shall show first that

$$g_{\vartheta,0}^+ = 0, \tag{4.11}$$

independently of ϑ. For the first term on the right-hand side of (4.10), we have

$$2c_\vartheta^+(z) = 2a_\vartheta^+(z) + 2b_\vartheta^+(z)(1 - z/z_\vartheta^+)^{1/2} \tag{4.12}$$

in a neighborhood of z_ϑ^+. For the second term, we have

$$z^2 \exp\bigl(2c_\vartheta(z) + 2\Psi_\vartheta(z)\bigr) = z^2 \exp\bigl(2a_\vartheta^+(z) + 2b_\vartheta^+(z)(1 - z/z_\vartheta^+)^{1/2} + 2\Psi_\vartheta(z)\bigr).$$

By (3.10), this expression tends to 1 as z tends to z_ϑ^+. Since $\exp\bigl(2b_\vartheta^+(z)(1-z/z_\vartheta^+)^{1/2}\bigr)$ also tends to 1 in this limit, we conclude that $\exp\bigl(2a_\vartheta^+(z) + 2\Psi_\vartheta(z)\bigr)$ tends to 1 as z tends to z_ϑ^+. Since this last expression is analytic at z_ϑ^+, we have

$$\exp\bigl(2a_\vartheta^+(z) + 2\Psi_\vartheta(z)\bigr) = 1 + O(1 - z/z_\vartheta^+).$$

We also have

$$\exp\bigl(2b_\vartheta^+(z)(1 - z/z_\vartheta^+)^{1/2}\bigr) = 1 + 2b_\vartheta^+(z)(1 - z/z_\vartheta^+)^{1/2} + O(1 - z/z_\vartheta^+);$$

we conclude that

$$z^2 \exp\bigl(2c_\vartheta(z) + 2\Psi_\vartheta(z)\bigr) = 1 + 2b^+_\vartheta(z)(1 - z/z^+_\vartheta)^{1/2} + O(1 - z/z^+_\vartheta).$$

Combining this with (4.12) in (4.10) yields $g^+_\vartheta(z) = O(1 - z/z^+_\vartheta)$, which is (4.11).

Since $g^+_{\vartheta,1}$ is an even analytic function of ϑ, we have

$$g^+_{\vartheta,1} = g^+_{0,1} + O(\vartheta^2).$$

To determine the value of $g^+_{0,1}$, we observe that $g^+_0(z) = g(z)$, so that $g^+_{0,1} = g_1 = 2(2A)^{3/2}/3$. Thus we have

$$g^+_{\vartheta,1} = \frac{2}{3}(2A)^{3/2} + O(\vartheta^2).$$

Combining this with (4.11) yields

$$g^+_\vartheta(z) = \frac{2}{3}(2A)^{3/2}(1 - z/z^+_\vartheta) + O\bigl(\vartheta^2(1 - z/z^+_\vartheta)\bigr) + O\bigl((1 - z/z^+_\vartheta)^2\bigr).$$

Similar arguments give, in the neighborhood of $z = z^-_\vartheta$, an expansion of the form

$$h^-_\vartheta(z) = f^-_\vartheta(z) + g^-_\vartheta(z)\,(1 - z/z^-_\vartheta)^{1/2},$$

where

$$g^-_\vartheta(z) = \frac{2}{3}(2A)^{3/2}(1 - z/z^-_\vartheta) + O\bigl(\vartheta^2(1 - z/z^-_\vartheta)\bigr) + O\bigl((1 - z/z^-_\vartheta)^2\bigr).$$

With these expansions for the singularities of $h_\vartheta(z)$, we can proceed as before to apply Darboux's lemma to the integrand for each value of ϑ, then integrate the result from $-\pi/2$ to $3\pi/2$, with the greatest contributions coming when ϑ is near 0 or π. For even n the results are

$$h^*_n \sim \frac{4A^2}{\pi(B^2 - 4C)^{1/2}}\,\frac{z_0^{-n}}{n^3}$$

for (4.9) and

$$u^*_n \sim \frac{2A^2}{\pi(B^2 - 4C)^{1/2}}\,\frac{z_0^{-n}}{n^3}$$

for (4.8). For n odd, of course, $h^*_n = u^*_n = 0$. We observe that the limiting value, as n tends to infinity through even values, of the ratio of u^*_n/u_n (the probability that a randomly chosen n-vertex unrooted tree is equicolorable) to $\binom{n}{n/2}/2^n \sim (2/\pi n)^{1/2}$ (the probability that n vertices, independently assigned colors by unbiased coin flips, are equicolored) is $2A^{1/2}/(B^2 - 4C)^{1/2} = 1.40499\ldots$.

5. Conclusion. All of our results enumerating equicolorable trees have been obtained by first enumerating equicolored trees, then relying on a relatively simple relationship between the two enumerations. We conclude by mentioning some problems where the relationship is more complicated. First we may consider the enumeration of equicolorable forests (wherein the individual trees need not be equicolorable). It should be relatively easy to enumerate equicolored forests of rooted or unrooted trees,

but the number of equicolorings of a given equicolorable forest depends in a rather complicated way on the structure of the forest. In another direction, we may consider the number of trees that are equitably colorable with three (or more) colors. Again, it should be relatively easy to enumerate equitable tricolorings of trees; results for the labelled case are given by Austin [A]. However, whereas a tree has just two bicolorings, and one is equitable if and only if both are, a tree with n vertices has $3 \cdot 2^{n-1}$ tricolorings, and the number of these that are equitable depends in a rather complicated way on the structure of the tree.

REFERENCES

[A] T. L. Austin, *The enumeration of point-labelled chromatic graphs and trees*, Canad. J. Math., 12 (1960), pp. 535–545.

[B] C. W. Borchardt, *Über eine der Interpolation entsprechende Darstellung der Eliminations-Resultante*, J. Reine Angew. Math., 57 (1960), pp. 111–121.

[C1] A. Cayley, *On the theory of the analytical forms called trees*, Phil. Mag., 13 (1857), pp. 172–176.

[C2] A. Cayley, *On the theory of the analytical forms called trees, second part*, Phil. Mag., 18 (1859), pp. 374–378.

[C3] A. Cayley, *On the analytical forms called trees, with application to the theory of chemical combinations*, Rep. Brit. Assoc. Adv. Sci., (1875), pp. 257–305.

[C4] A. Cayley, *On the analytical forms called trees*, Amer. J. Math., 4 (1881), pp. 266–268.

[C5] A. Cayley, *A theorem on trees*, Quart. J. Math., 23 (1889), pp. 376–378.

[C6] A. Cayley, *Collected Mathematical Papers*, Cambridge University Press, London, 1897.

[D] G. Darboux, *Mémoire sur l'approximation des fonctions des très grands nombres, et sur une classe étendue des développements en série*, J. Math. Pures Appl., 4 (1878), pp. 5–56, 377–416.

[G] S. Glicksman, *On the representation and enumeration of trees*, Proc. Cambridge Philos. Soc., 59 (1963), pp. 509–517.

[H] M. L. J. Hautus and D. A. Klarner, *The diagonal of a double power series*, Duke Math. J., 38 (1971), pp. 229–235.

[K] D. E. Knuth and H. S. Wilf, *A short proof of Darboux's lemma*, Appl. Math. Lett., 2 (1989), pp. 139–140.

[M1] J. W. Moon, *Various proofs of Cayley's formula for counting trees*, in A Seminar on Graph Theory, F. Harary, ed., Holt, Rinehart and Winston, New York, 1967, pp. 70–78.

[M2] J. W. Moon, *Counting Labelled Trees*, Canadian Mathematical Society, Montreal, Canada, 1970.

[O] R. Otter, *The number of trees*, Ann. of Math. (2), 49 (1948), pp. 583–599.

[P1] G. Pólya, *Kombinatorische Anzahlbestimmungen für Gruppen, Graphen und chemische Verbindungen*, Acta Math., 68 (1937), pp. 145–254.

[P2] G. Pólya and R. C. Read, *Combinatorial Enumeration of Groups, Graphs and Chemical Compounds*, Springer-Verlag, New York, Berlin, 1987.

[S1] H. I. Scoins, *The number of trees with nodes of alternate parity*, Proc. Cambridge Philos. Soc., 58 (1962), pp. 72–76.

[S2] J. J. Sylvester, *On the change of systems of independent variables*, Quart. J. Pure Appl. Math., 1 (1957), pp. 42–56.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 116–120

A NOTE ON A QUESTION OF C. D. SAVAGE*

MICHAEL NAATZ†

Abstract. Given a graph G and an orientation σ of some of its edges, consider the graph $AO_\sigma(G)$ which is defined as follows: The vertices are the acyclic orientations of G which agree with σ, and two of these are adjacent if they differ only by the reversal of a single edge. $AO_\sigma(G)$ is easily seen to be bipartite. The purpose of this note is to show that it need not contain a Hamilton path even if both partite sets have the same cardinality. This answers a question of C. D. Savage [*SIAM Rev.*, 39 (1997), pp. 605–629] and sheds new light onto two well-known open questions in the field of combinatorial Gray codes.

Key words. Hamilton path, Hamilton cycle, acyclic orientation, poset, linear extension, adjacent transposition

AMS subject classifications. 05C45, 05C20, 05C30, 06A07

PII. S0895480100369195

1. Introduction.

The area of combinatorial Gray codes deals with the following problem: We are given a class of combinatorial objects and a rule that decides when two of these objects are considered similar. The task is now to generate all objects in the class in such a way that successive objects are similar. The most classical example consists of the 0/1-strings of given length as objects and the rule which says that two strings are similar when they differ only in one bit. We refer to Savage's survey [10] for an overview of this flourishing direction of research.

A simple but important observation is the fact that a combinatorial Gray code can be seen as a Hamilton path in a certain graph: Just take as vertices the combinatorial objects to be generated, and let two of them be adjacent if they are similar. This graph turns out to be bipartite for the classes of objects and the similarity rules we want to deal with in this paper. In a bipartite graph, a Hamilton path can exist only when the *parity difference* is at most one; i.e., the cardinalities of the two partite sets differ by at most one. For a Hamilton cycle to be possible, the parity difference has to be zero. The question we want to treat here is whether such necessary conditions are also sufficient in certain special cases.

Suppose we are given a graph G and an orientation σ of some of its edges. An orientation of all edges of G is called *acyclic* if it does not contain any oriented cycles. Let us define a graph $AO_\sigma(G)$ as follows: The vertices are the acyclic orientations of G which agree with σ, and two of these are adjacent if they differ only by the reversal of a single edge. This graph is easily seen to be bipartite by the following argument. For every vertex v of $AO_\sigma(G)$, count the number of edges of G which have the same orientation in v and some arbitrary fixed vertex w; two vertices for which this number has the same parity can obviously not be adjacent. We are now in a position to state the question posed by Savage.

QUESTION 1.1 (Savage [10]). *Does $AO_\sigma(G)$ contain a Hamilton cycle whenever the parity difference allows?*

*Received by the editors March 8, 2000; accepted for publication (in revised form) November 2, 2000; published electronically January 16, 2001. This research was supported by the graduate school "Algorithmische Diskrete Mathematik," Deutsche Forschungsgemeinschaft grant GRK 219/3.
http://www.siam.org/journals/sidma/14-1/36919.html
†Technische Universität Berlin, Fachbereich Mathematik MA 6-1, Straße des 17. Juni 136, D-10623 Berlin, Germany (naatz@math.tu-berlin.de).

Before we answer this question in the negative by constructing an infinite number of counterexamples, we give a short review of two related problems motivating the question.

The first problem deals with combinatorial Gray codes for linear extensions of posets. Suppose we are given a poset, i.e., a ground set P together with a binary relation $\leq_P$ on P which is reflexive, transitive, and antisymmetric. By $x <_P y$ we denote $x \leq_P y$ and $x \neq y$. A linear extension of a poset is a permutation $p_1 p_2 \dots p_n$ of the elements of P such that $p_i \leq_P p_j$ implies $i \leq j$. Let us regard two linear extensions as similar if they differ only by an adjacent transposition; i.e., one can be constructed from the other by interchanging two successive elements. Now consider the graph $G(P, \leq_P)$ which has the linear extensions of a given poset $(P, \leq_P)$ as vertices and the pairs of similar extensions as edges. Such graphs are called *adjacent transposition graphs* or *graphs of linear extensions*. They are bipartite because each edge consists of an even and an odd permutation. Ruskey made the following conjecture in 1988.

CONJECTURE 1.2 (Ruskey [8]). *A graph of linear extensions has a Hamilton path if the parity difference is at most one.*

This conjecture is still unsolved despite considerable efforts to settle it. There are, however, many partial results [5, 9, 13, 15] showing that the conjecture is true in many special cases. When we regard $\leq_P$ as the prescribed orientation of some edges of the complete graph on the ground set P, this problem is easily seen to be intimately related to Question 1.1.

The second problem which is of interest in this context is the construction of a Gray code for the acyclic orientations of a graph G in such a way that successive orientations differ only by the reversal of a single edge; i.e., we are dealing with $AO_\sigma(G)$ in the special case that none of the edges has a prescribed orientation. We want to denote this graph by $AO(G)$ and call it the *acyclic orientation graph* of G. Edelman (cf. [10]) asked the following question.

QUESTION 1.3 (Edelman). *Does an acyclic orientation graph contain a Hamilton cycle whenever the parity difference is zero?*

This problem has not been as thoroughly studied as the linear extensions above, but in some special cases the answer to the question is "yes," as demonstrated by Savage, Squire, and West in [11]. In the course of that research it also turned out that even seemingly simple cases, such as the acyclic orientations of the complete bipartite graph $K_{m,n}$ with mn odd, pose great difficulties.

2. The example. We will now construct an infinite family of pairs (G, σ) such that the two partite sets of $AO_\sigma(G)$ have the same size, but the graph does not contain a Hamilton path. The basic structure that makes the examples work is the so-called standard example from order theory (see Trotter's monograph [14] for details on the use of this family of posets). The standard example is also the main ingredient in the construction by Pruesse and Ruskey [7] demonstrating that the Cartesian product of a K_2 with a graph of the form $AO_\sigma(G)$ need not contain a Hamilton cycle. However, the connection to order theory is not mentioned in that paper.

A remark on notation and figures is in order: It is easy to see that adding transitive arcs to σ or deleting such arcs does not change $AO_\sigma(G)$, viewed as an abstract graph. This enables us to regard σ as a poset and treat the unoriented edges of G separately. This is not only helpful when utilizing order theoretic constructions but also provides clear figures in which the oriented edges are represented by a Hasse diagram (no transitive arcs, all orientations upward), and the unoriented ones are depicted as dotted lines.

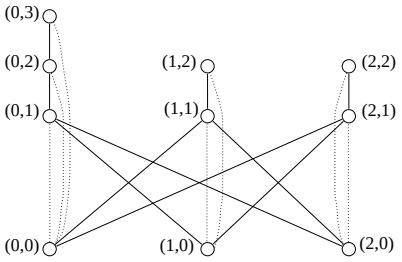


FIG. 2.1. *Hasse diagram of* $(P_2, \leq_2)$ *(solid lines) and unoriented edges of* G_2 *(dotted lines).*

EXAMPLE 2.1. *For* $n \geq 2$*, construct a poset on the ground set* $P_n := (\{0, \ldots, n\} \times \{0, 1, 2\}) \cup \{(0, 3)\}$ *by setting* $(i_1, i_2) <_n (j_1, j_2)$ *if and only if*

$$(i_1 \neq j_1 \text{ and } 0 = i_2 < j_2) \text{ or } (i_1 = j_1 \text{ and } 0 < i_2 < j_2).$$

The subposet induced by $\{0, \ldots, n\} \times \{0, 1\}$ *is the standard example* S_{n+1}*. Construct a graph* G_n *by adding to the comparability graph of* $(P_n, \leq_n)$ *the edges that join* $(i, 0)$ *with* (i, j) *for* $j > 0$*. Figure* 2.1 *shows the result of the construction for* $n = 2$*.*

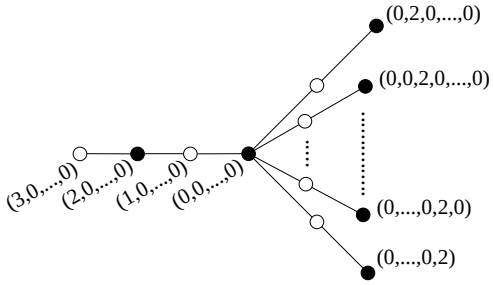
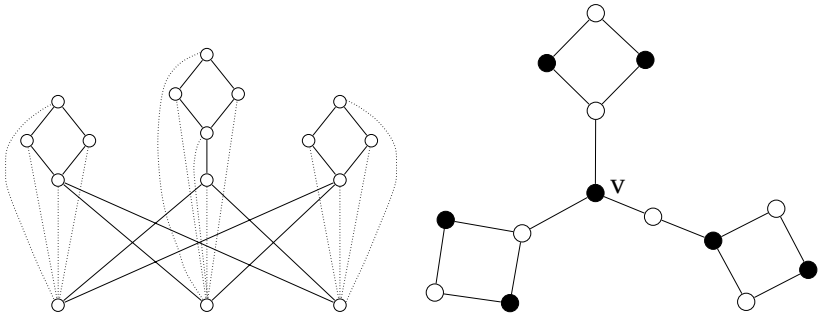


FIG. 2.2. $AO_{\leq_n}(G_n)$ *with some vertices labeled by in-degree vectors; dots stand for omitted vertices; vertex colors indicate bipartition.*

Now consider the graph $AO_{\leq_n}(G_n)$. In an acyclic orientation which contains an arc from (i, j) to $(i, 0)$, the edges that join $(i, 0)$ with (i, j') for $j' < j$ must be directed from (i, j') to $(i, 0)$ because otherwise $(i, 0)$ is contained in a directed cycle. A directed cycle is also created if two distinct vertices of the form $(i, 0)$ have positive in-degree. Hence, we have two necessary conditions for an orientation of G_n that respects $\leq_n$ to be acyclic. A moment of thought shows that these conditions are also sufficient. This implies that each vertex of $AO_{\leq_n}(G_n)$ is uniquely determined by the vector $(d_0, d_1, \ldots, d_n)$, where d_i is the in-degree of vertex $(i, 0)$. It is easy to see that two vertices are adjacent if and only if the in-degree vector of one vertex can be obtained from the other by increasing one component by one. We conclude that the graph in Figure 2.2 is isomorphic to $AO_{\leq_n}(G_n)$. Obviously both partite sets have the same size. The graph cannot contain a Hamilton path because the central vertex (with all-zero in-degree label) has at least three incident edges each of which disconnects the graph when removed.

Note that this family of examples can be easily modified by replacing the maximal elements of $(P_n, \leq_n)$ with more complicated structures and adding the necessary unoriented edges. In this way one can. for example, obtain constructions as in Figure 2.3

demonstrating that a Hamilton path cannot be forced by additionally forbidding vertices of degree one in $AO_{\preceq}(G)$.



(a) G and $\preceq$, to be read in the same way as in Figure 2.1.

(b) $AO_{\preceq}(G)$; vertex colors indicate bipartition; v is a cut vertex.

FIG. 2.3. *A modified version of Example* 2.1.

3. Conclusion. In view of the above examples, one could suspect that either Conjecture 1.2 is wrong, or the answer to Question 1.3 is "no"—or both. Recent results, however, suggest to be cautious here. Properties shared by all graphs of linear extensions and also all acyclic orientation graphs need not carry over to graphs of the form $AO_\sigma(G)$. It has already been mentioned that the Cartesian product of a K_2 and a graph of the form $AO_\sigma(G)$ need not contain a Hamilton cycle, in contrast to the situation for acyclic orientation graphs [7] and graphs of linear extensions [6]. Also, it has been shown in [4] that for every poset $(P, \leq_P)$, the vertex connectivity of the linear extension graph $G(P, \leq_P)$ equals the minimal degree. This statement also holds for acyclic orientation graphs. There are (at least) two different proofs: The first possibility is to use the result by Savage and Zhang [12] that $AO(G)$ is $(n-c)$-connected, where n is the number of vertices of G, and c is the number of connected components. This can be combined with the observation of Fisher et al. [2] that a depth-first search tree for each connected component, oriented away from the root, yields an orientation of G whose degree in $AO(G)$ is exactly $n-c$. The second proof derives the connectivity of $AO(G)$ as a corollary from a general theorem on tope graphs of oriented matroids; see [1] and [3] for details. However, Figure 2.3 shows an example for a graph of the form $AO_\sigma(G)$ which has minimal degree two but contains several cut vertices, e.g., the vertex v.

Acknowledgment. The author thanks the referees for detailed comments which improved the presentation of this paper.

REFERENCES

[1] R. CORDOVIL AND K. FUKUDA, *Oriented matroids and combinatorial manifolds*, European J. Combin., 14 (1993), pp. 9–15.

[2] D. C. FISHER, K. FRAUGHNAUGH, L. LANGLEY, AND D. B. WEST, *The number of dependent arcs in an acyclic orientation*, J. Combin. Theory Ser. B, 71 (1997), pp. 73–78.

[3] K. FUKUDA, *Notes on acyclic orientations and the shelling lemma*, Theoret. Comput. Sci., to appear.

[4] M. Naatz, *The graph of linear extensions revisited*, SIAM J. Discrete Math., 13 (2000), pp. 354–369.
[5] G. Pruesse and F. Ruskey, *Generating the linear extensions of certain posets by transpositions*, SIAM J. Discrete Math., 4 (1991), pp. 413–422.
[6] G. Pruesse and F. Ruskey, *Generating linear extensions fast*, SIAM J. Comput., 23 (1994), pp. 373–386.
[7] G. Pruesse and F. Ruskey, *The prism of the acyclic orientation graph is Hamiltonian*, Electron. J. Combin., 2 (1995).
[8] F. Ruskey, *Research problem* 91, Discrete Math., 70 (1988), p. 112.
[9] F. Ruskey and C. D. Savage, *Hamilton cycles that extend transposition matchings in Cayley graphs of S_n*, SIAM J. Discrete Math., 6 (1993), pp. 152–166.
[10] C. D. Savage, *A survey of combinatorial Gray codes*, SIAM Rev., 39 (1997), pp. 605–629.
[11] C. D. Savage, M. B. Squire, and D. B. West, *Gray code results for acyclic orientations*, Congr. Numer., 96 (1993), pp. 185–204.
[12] C. D. Savage and C.-Q. Zhang, *A note on the connectivity of acyclic orientation graphs*, Discrete Math., 184 (1998), pp. 281–287.
[13] G. Stachowiak, *Hamilton paths in graphs of linear extensions for unions of posets*, SIAM J. Discrete Math., 5 (1992), pp. 199–206.
[14] W. T. Trotter, *Combinatorics and Partially Ordered Sets: Dimension Theory*, Johns Hopkins Ser. Math. Sci., The Johns Hopkins University Press, Baltimore, MD, 1992.
[15] D. B. West, *Generating linear extensions by adjacent transpositions*, J. Combin. Theory Ser. B, 58 (1993), pp. 58–64.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 121–137

CYCLIC CHROMATIC NUMBER OF 3-CONNECTED PLANE GRAPHS*

HIKOE ENOMOTO†, MIRKO HORŇÁK‡, AND STANISLAV JENDROL'‡

Abstract. Let G be a 3-connected plane graph. Plummer and Toft [*J. Graph Theory*, 11 (1987), pp. 507–515] conjectured that $\chi_c(G) \leq \Delta^*(G) + 2$, where $\chi_c(G)$ is the cyclic chromatic number of G and $\Delta^*(G)$ the maximum face size of G. Horňák and Jendrol' [*J. Graph Theory*, 30 (1999), pp. 177–189] and Borodin and Woodall [*SIAM J. Discrete Math.*, submitted] independently proved this conjecture when $\Delta^*(G)$ is large enough. Moreover, Borodin and Woodall proved a stronger statement that $\chi_c(G) \leq \Delta^*(G) + 1$ holds if $\Delta^*(G) \geq 122$. In this paper, we prove that $\chi_c(G) \leq \Delta^*(G) + 1$ holds if $\Delta^*(G) \geq 60$.

Key words. cyclic coloring, cyclic chromatic number

AMS subject classification. 05C15

PII. S0895480198346150

1. Introduction. In this paper, we consider only 3-connected plane graphs. For a vertex v, $\deg(v)$ denotes the degree of v. For a face f, $\deg(f)$ denotes the face size of f. Two vertices u and v of a 3-connected plane graph G are *cyclically adjacent* if they are incident with a common face of G. A vertex coloring of G is called a *cyclic coloring* if it assigns different colors to any pair of cyclically adjacent vertices. The minimum number of colors necessary for a cyclic coloring of G is called the *cyclic chromatic number* of G and is denoted by $\chi_c(G)$. The notion of the cyclic chromatic number was introduced by Ore and Plummer [6], and Plummer and Toft [7] conjectured $\chi_c(G) \leq \Delta^*(G) + 2$, where $\Delta^*(G)$ is the maximum face size of G (see also [5, Problem 2.5]). Recently, Horňák and Jendrol' [4] and Borodin and Woodall [2] proved that this conjecture is true when $\Delta^*(G)$ is large enough. Moreover, Borodin and Woodall proved a stronger inequality that $\chi_c(G) \leq \Delta^*(G) + 1$ if $\Delta^*(G) \geq 122$.

In this paper, we prove the same inequality with a better bound on $\Delta^*(G)$.

THEOREM 1. *Let G be a 3-connected plane graph. Then $\chi_c(G) \leq \Delta^*(G) + 1$ if $\Delta^*(G) \geq 60$.*

The proof technique is standard in some sense; that is, we use discharging and reducible configurations. However, the discharging rules and the reducible configurations used in this paper are quite different from those used in [2].

2. Preliminaries. It is easily seen that the following theorem implies Theorem 1.

THEOREM 2. *Let M be a constant ≥ 60, and suppose G is a 3-connected plane graph satisfying $\Delta^*(G) \leq M$. Then $\chi_c(G) \leq M + 1$.*

*Received by the editors October 19, 1998; accepted for publication (in revised form) November 2, 2000; published electronically January 31, 2001.
http://www.siam.org/journals/sidma/14-1/34615.html

†Department of Mathematics, Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522, Japan (enomoto@math.keio.ac.jp). The work of this author has been partly supported by the Ministry of Education, Science, Sports and Culture of Japan, Grant-in-Aid for Scientific Research (B), 10440032.

‡Department of Geometry and Algebra, P.J.Šafárik University, Jesenná 5, 041 54 Košice, Slovakia (hornak@duro.science.upjs.sk, jendrol@kosice.upjs.sk). The work of these authors has been supported by Slovak VEGA grant 1/4377/97.

By way of contradiction, let G be a minimum counterexample to Theorem 2. That is, we assume that $\chi_c(H) \le M+1$ holds if H is a 3-connected plane graph satisfying $\Delta^*(H) \le M$, and either $|V(H)| < |V(G)|$ or $|V(H)| = |V(G)|$ and $|E(H)| < |E(G)|$.

For an edge xy of G, the graph obtained from G by contracting xy is denoted by G/xy, and xy is called *contractible* if G/xy is 3-connected. If xy is contractible, $\chi_c(G/xy) \le M+1$ by the minimality of G. (Note that $|V(G/xy)| = |V(G)| - 1$ and $\Delta^*(G/xy) \le \Delta^*(G)$.) Therefore G/xy has a cyclic coloring using at most $M+1$ colors. This induces a coloring of $V(G)-\{x\}$ using at most $M+1$ colors that assigns different colors to any pair of vertices $u, v \in V(G)-\{x\}$ that are cyclically adjacent in G. Such a coloring is called a *partial cyclic coloring* of G except at x.

In the rest of this paper, a cyclic coloring or a partial cyclic coloring always means a coloring using at most $M+1$ colors.

LEMMA 3. *If* $\deg(x) = 3$, *G has a partial cyclic coloring except at x.*

Proof. By [3], x is incident with a contractible edge. Hence there is a partial cyclic coloring of G except at x by the above remark. □

The set of vertices adjacent with x is called the neighborhood of x and is denoted by $N(x)$. The set of vertices cyclically adjacent with x is called the *cyclic neighborhood* of x. The number of cyclically adjacent vertices with x is called the *cyclic degree* of x and is denoted by $\mathrm{cd}(x)$. Note that if $\deg(x) = n$ and $d_1, \dots, d_n$ are the sizes of the faces incident with x, then $\mathrm{cd}(x) = \sum_{i=1}^{n}(d_i - 2)$. If φ is a partial cyclic coloring of G except at x, and if $\mathrm{cd}(x) \le M$, then φ can be extended to a cyclic coloring of G. This fact is frequently used without explicitly mentioning it.

The following lemma was proved by Borodin and Woodall [2], but for the sake of readability, we include the proof.

LEMMA 4. *For any* $x \in V(G)$, $\mathrm{cd}(x) \ge M+1$.

Proof. If $\deg(x) = 3$, G has a partial cyclic coloring except at x. This can be extended to a cyclic coloring of G if $\mathrm{cd}(x) \le M$. This contradicts the assumption that G is a minimum counterexample. Therefore we may assume that $\mathrm{cd}(x) \ge M+1$ for any vertex x of degree 3.

Next suppose $\deg(x) \ge 4$ and let $y_1, \dots, y_n$ be the neighbors of x in this order. Let G' be the graph obtained from G by adding the edges $y_iy_{i+1} (1 \le i \le n)$ if they are not adjacent in G (taking $y_{n+1} = y_1$). If xy_i is a contractible edge of G' for some i, G' has a partial cyclic coloring except at x. This is also a partial cyclic coloring of G and can be extended to a cyclic coloring of G if $\mathrm{cd}(x) \le M$. Suppose xy_i is noncontractible for all i, $1 \le i \le n$. Then $\deg_{G'}(y_i) = 3$ for some i by [1, Corollary 4]. This implies that $\deg_G(y_i) = 3$ and y_i is incident with two triangles in G. Then $\mathrm{cd}(y_i) \le \Delta^*(G) \le M$, a contradiction. □

LEMMA 5. *Suppose xy_1y_2 is a triangle of G,* $\deg(x) = 3$, *and* $\deg(y_i) \ge 4 (i = 1, 2)$. *Let f be the face incident with y_1 and y_2 but not with x. Then* $\deg(f) = M$.

Proof. Suppose $\deg(f) < M$. Then $\Delta^*(G - y_1y_2) \le M$. If $G - y_1y_2$ is 3-connected, $G - y_1y_2$ has a cyclic coloring, which is also a cyclic coloring of G. Hence we may assume that $G - y_1y_2$ is not 3-connected. Let S be a 2-cut of $G - y_1y_2$. Then y_1 and y_2 should belong to different components of $G - y_1y_2 - S$. Therefore $x \in S$. Let $N_G(x) = \{y_1, y_2, z\}$. If $\{x, z\}$ is a cut of $G - y_1y_2$, then $\{y_1, z\}$ is a cut of G. Therefore $z \notin S$. By symmetry, we may assume that z and y_1 belong to different components of $G - y_1y_2 - S$. Then $(S - \{x\}) \cup \{y_1\}$ is a 2-cut of G. □

LEMMA 6. *None of the configurations in Figure* 1 *is contained in G.*

Proof. In case (a'), x_1, y_1, and z_1 are incident with a common face. In cases (c) through (f), let f_1 be the face incident with x_1y_1 of face size > 4. In cases (b) through

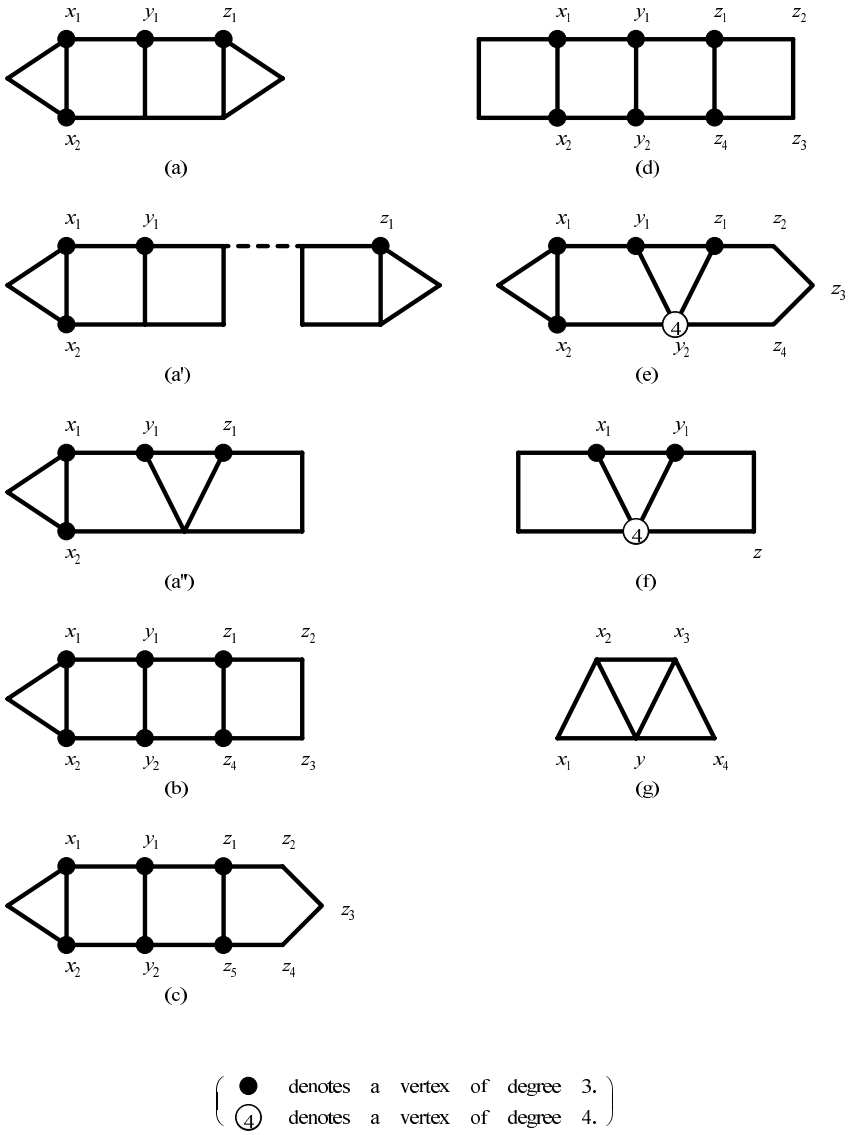


FIG. 1. *Reducible configurations.*

(d), let f_2 be the face incident with x_2y_2 other than the quadrangle $x_1y_1y_2x_2$. For a partial coloring φ, let $C(f_i)$ be the set of colors appearing on the boundary of f_i.

(a) (a') (a") G has a partial coloring φ except at x_1. Suppose φ cannot be extended to a cyclic coloring of G. Note that $\mathrm{cd}(x_1) = \mathrm{cd}(x_2) = M + 1$ by Lemma 4. Hence all the colors appear exactly once in the cyclic neighborhood of x_1. If we can color x_2 with a color different from $\varphi(x_2)$, it is extendable to a cyclic coloring of G. Also, all the colors except $\varphi(x_2)$ appear exactly once in the cyclic neighborhood of x_2. The same arguments apply to z_1. Therefore if we uncolor y_1, we can color both x_2 and z_1 with $\varphi(y_1)$. Then we can color y_1 and x_1 in this order.

(b) G has a partial cyclic coloring φ except at x_2. If $\varphi(z_i) \notin C(f_2)$ for $i = 1$ or $i = 2$, we can color x_2 with $\varphi(z_i)$. Therefore we may assume $\{\varphi(z_1), \varphi(z_2)\} \subseteq C(f_2)$. Uncolor z_4 and color x_2 and z_4 in this order.

(c) We may assume that φ is a partial cyclic coloring of G except at x_2, and $\{\varphi(z_1), \varphi(z_2)\} \subseteq C(f_2)$. If $\varphi(z_3) \in C(f_2)$, uncolor z_5 and color x_2 and z_5 in this order. Therefore we may assume that $\varphi(z_3) \notin C(f_2)$. If we uncolor x_1, we can color x_2 with $\varphi(x_1)$. By the same arguments as above, we may assume that $\varphi(z_3) \notin C(f_1) - \{\varphi(x_1)\}$ for any partial cyclic coloring φ of G except at x_2. If $\varphi(z_3) \neq \varphi(x_1)$, we can color y_1 with $\varphi(z_3)$. Therefore we may assume that $\varphi(z_3) = \varphi(x_1)$. Then we can exchange the colors of x_1 and y_1, since $\varphi(x_1) = \varphi(z_3) \neq \varphi(z_5)$. Then y_1 and z_3 have the same color.

(d) Since x_2y_2 is contractible, G has a partial cyclic coloring φ except at x_2. If $\varphi(z_i) \notin C(f_2)$ for $i = 1$ or $i = 2$, we can color x_2 with $\varphi(z_i)$. Therefore we may assume that $\{\varphi(z_1), \varphi(z_2)\} \subseteq C(f_2)$. If $\varphi(y_2)$ appears more than once in the cyclic neighborhood of x_2, uncolor z_4 and color x_2 and z_4. If $\varphi(y_2)$ appears exactly once in the cyclic neighborhood of x_2, uncolor y_2 and z_4, color x_2 with $\varphi(y_2)$, and then color y_2 and z_4.

(e) G has a partial cyclic coloring φ except at x_1. If $\varphi(z_4) \notin C(f_1)$, we can color x_1 with $\varphi(z_4)$. Therefore we may assume that $\varphi(z_4) \in C(f_1)$. If $\varphi(z_3) \in C(f_1)$, uncolor z_1, then color x_1 and z_1. Therefore we may assume that $\varphi(z_3) \notin C(f_1)$ for any partial cyclic coloring φ of G except at x_1. If $\varphi(x_2) \neq \varphi(z_3)$, we can color x_1 with $\varphi(z_3)$. Therefore we may assume that $\varphi(x_2) = \varphi(z_3)$. Uncolor x_2 and y_1. Then color y_1 with $\varphi(z_3)$ and color x_2. In this partial cyclic coloring except at x_1, $\varphi(z_3)$ appears in the boundary of f_1, a contradiction.

(f) G has a partial cyclic coloring φ except at x_1. If $\varphi(z) \notin C(f_1)$, we can color x_1 with $\varphi(z)$. If $\varphi(z) \in C(f_1)$, uncolor y_1, then color x_1 and y_1.

(g) If $G - x_2y$ is 3-connected, $G - x_2y$ has a cyclic coloring φ by the minimality of G. Then φ is also a cyclic coloring of G. Hence we may assume that $G - x_2y$ is not 3-connected. The only possibility of a 2-cut in $G - x_2y$ is $\{x_1, x_3\}$. Since x_2 and y belong to different components in $G - x_2y - \{x_1, x_3\}$, x_2 and x_4 are not adjacent. Similarly, we may assume that $\{x_2, x_4\}$ is the 2-cut in $G - x_3y$. Since $\deg(x_2) > 3$, there exists a vertex $z \in N(x_2) - \{x_1, y, x_3\}$ and let D be the component of $G - \{x_1, x_2, x_3\}$ that contains z. Since $G - \{x_1, x_2\}$ is connected, $V(D) \cap N(x_3) \neq \emptyset$. Similarly, since $G - \{x_2, x_3\}$ is connected, $V(D) \cap N(x_1) \neq \emptyset$. This is a contradiction, since x_1 and x_3 should belong to different components of $G - yx_3 - \{x_2, x_4\}$. □

3. Proof of Theorem 2. Let G be a minimum counterexample to Theorem 2. Then we can apply all the results in section 2.

For $v \in V(G)$, define

$$\Phi(v) = 1 - \frac{\deg(v)}{2} + \sum_f \frac{1}{\deg(f)},$$

where f runs through the faces incident with v. It is easily seen that

$$\sum_{v \in V(G)} \Phi(v) = 2$$

by Euler's formula (see [6]). We regard $\Phi(v)$ as the initial charge at v. Define

$$\gamma(d_1, \ldots, d_n) = 1 - \sum_{i=1}^{n} \left(\frac{1}{2} - \frac{1}{d_i} \right) = 1 - \frac{n}{2} + \sum_{i=1}^{n} \frac{1}{d_i}.$$

That is,

$$\Phi(v) = \gamma(d_1, \ldots, d_n)$$

if $f_1, \ldots, f_n$ are the faces incident with v, and $\deg(f_i) = d_i$ $(1 \le i \le n)$.

It is obvious that $\Phi(v) \le 0$ if $\deg(v) \ge 6$. Note that the number of triangles incident with v is at most $2\deg(v)/3$ by Lemma 6(g). Hence $\Phi(v) = \gamma(d_1, d_2, d_3, d_4, d_5) \le \gamma(3,3,4,3,4) = 0$ if $\deg(v) = 5$. If $\deg(v) = 4$, by Lemma 4, either $\Phi(v) \le \gamma(3,5,5,5) = -1/15$ or $\Phi(v) \le \gamma(3,4,d_3, M+2-d_3) \le \gamma(3,4,4,M-2) = -1/6 + 1/(M-2) < 0$ or $\Phi(v) \le \gamma(3,3,d_3,M+3-d_3) \le \gamma(3,3,4,M-1) = -1/12 + 1/(M-1) < 0$.

LEMMA 7. *Suppose* $\Phi(v) > 0$, $\deg(v) = 3$, f_1, f_2, f_3 *are the faces incident with* v, *and* $k = \deg(f_1) \le l = \deg(f_2) \le m = \deg(f_3)$. *Then either*

(1) $k = 3$ *and* $4 \le l \le 6$, *or*

(2) $k = l = 4$.

Proof. Note that $\mathrm{cd}(v) = k + l + m - 6 \ge M + 1$ by Lemma 4, and $\gamma(k,l,m) < \gamma(k, l-1, m+1)$ when $k < l \le m < M$. It is easily seen that the lemma follows from the fact that $\gamma(3,7,M-3) = -1/42 + 1/(M-3) \le 0$ and $\gamma(4,5,M-2) = -1/20 + 1/(M-2) \le 0$. □

Let $\varepsilon_i = 1/(M-i)$. Note that

$$\gamma(3,4,M) = \frac{1}{12} + \varepsilon_0,$$

$$\gamma(3,5,M-1) = \frac{1}{30} + \varepsilon_1,$$

$$\gamma(3,6,M-2) = \varepsilon_2,$$

and

$$\gamma(4,4,M-1) = \varepsilon_1.$$

Let

$$V_3 = \{v \in V(G) \,|\, \deg(v) = 3\},$$

and

$$V^+ = \{v \in V(G) \,|\, \Phi(v) > 0\}.$$

Define an equivalence relation $\sim$ on V_3 as follows:

(o) $u \sim u$.

(i) $u \sim v$ if u and v are incident with a common triangle.

(ii) $u \sim v$ if u and v are adjacent and both faces incident with the edge uv are quadrangles.

(iii) If all the vertices on the boundary cycle of a quadrangle have degree 3, the vertices on the quadrangle are equivalent to each other.

(iv) $u \sim v$ only if this is derived by repeated applications of (o)–(iii).

LEMMA 8. *Let* W *be an equivalence class of* V_3 *that contains a vertex of* V^+. *Then* W *comes from one of the configurations in Figure* 2.

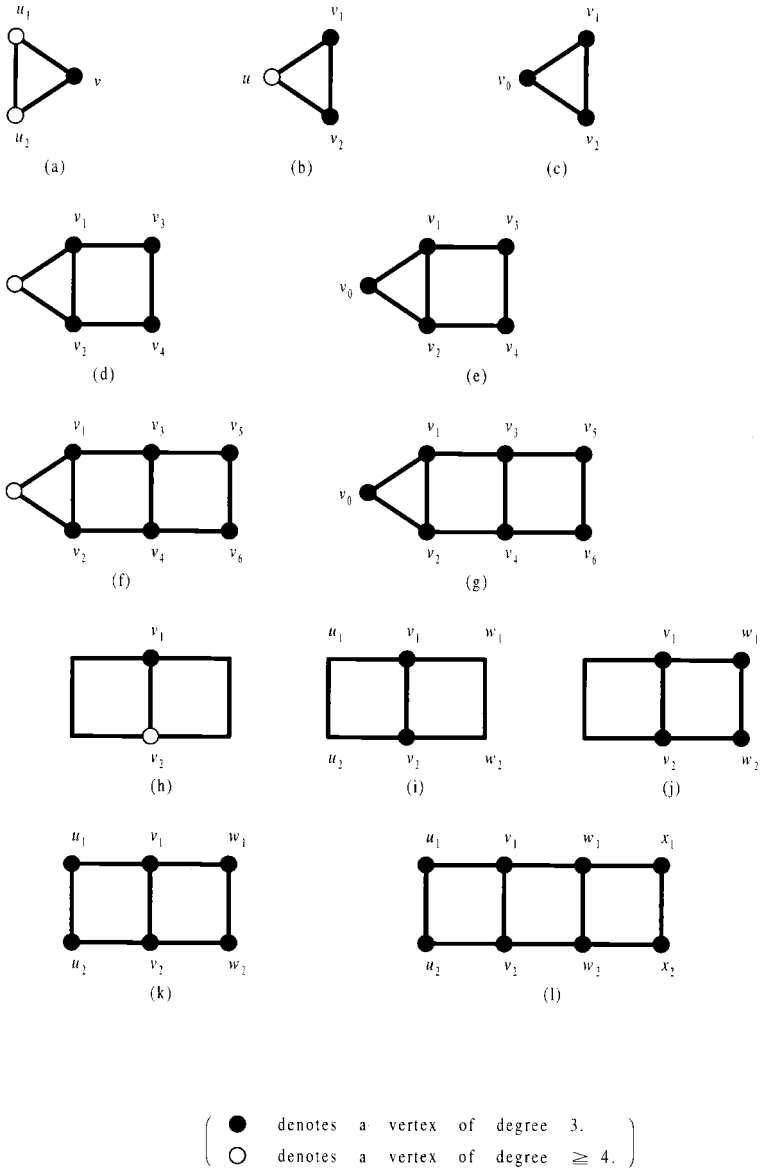


FIG. 2. *Equivalence classes.*

Proof. This is easily checked by Lemma 7 and Lemma 6(b), (d). □

Next we define discharging rules as in Figure 3. (Rule (b) (resp., (d), (j), (k)) is applied when (a) (resp., (c), (h), (i)) is not applied.)

Let $t(u, v)$ be the amount of the discharge from u to v. Then the new charge at

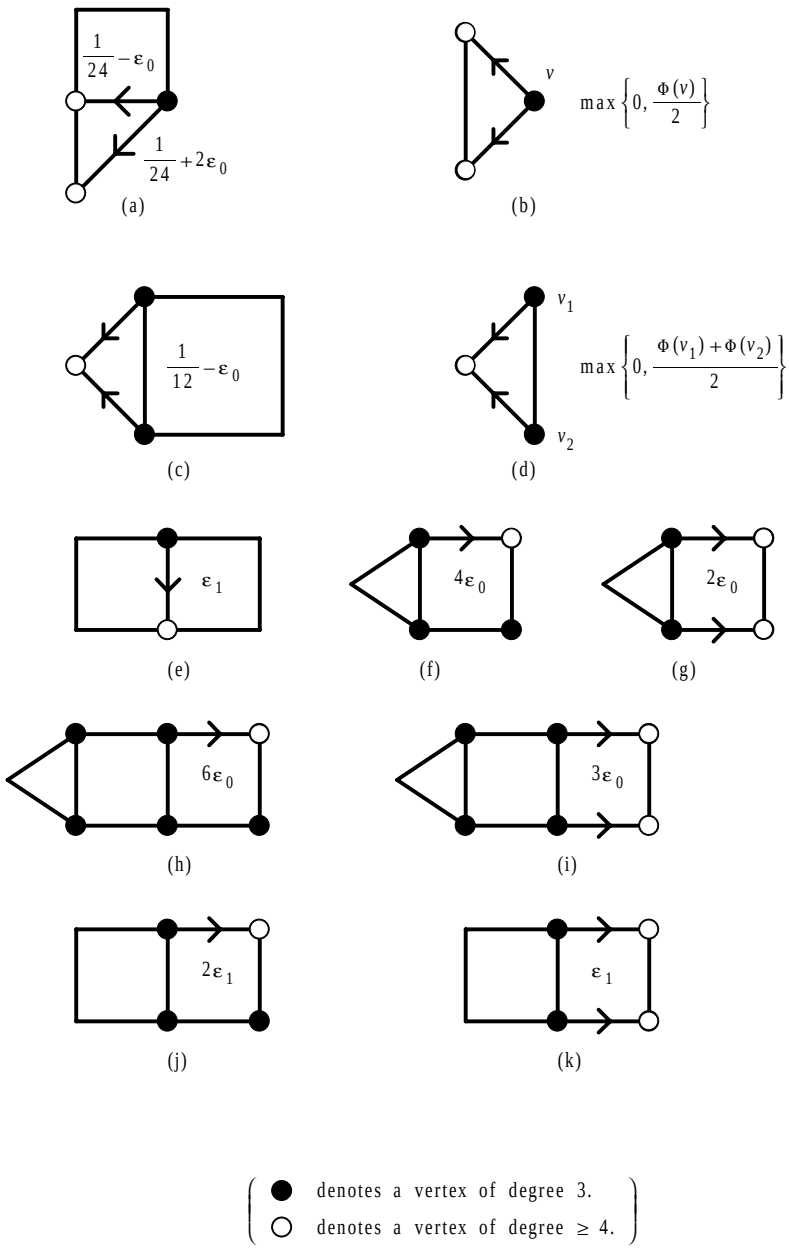


FIG. 3. *Discharging rules.*

v is

$$\Phi'(v) = \Phi(v) - \sum_{w \in N(v)} t(v,w) \quad \text{if} \quad \deg(v) = 3$$

and

$$\Phi'(v) = \Phi(v) + \sum_{u \in N(v)} t(u,v) \quad \text{if} \quad \deg(v) \geq 4.$$

Since $M \geq 60$,

$$\frac{1}{12} + \varepsilon_0 \geq 6\varepsilon_0 > 2\varepsilon_1,$$

and

$$\frac{1}{12} + \varepsilon_0 > \frac{1}{24} + 2\varepsilon_0.$$

Therefore $t(u,v) \leq 1/12 + \varepsilon_0$. Moreover, if uv is not incident with a triangle, then $t(u,v) \leq 6\varepsilon_0$.

CLAIM 1. *Let W be an equivalence class of V_3 under the relation $\sim$. Then $\sum_{v \in W} \Phi'(v) \leq 0$.*

Proof. For $v \in V_3$, define $t(v) = \sum_{w \in N(v)} t(v,w)$. We may assume that W comes from one of the configurations in Figure 2.

(a) $t(v) = t(v,u_1) + t(v,u_2) \geq \Phi(v)$ by rules (a) and (b).

(b) Let f be the face incident with v_1v_2 other than the triangle uv_1v_2. If $\deg(f) \geq 5$,

$$t(v_1,u) + t(v_2,u) \geq \Phi(v_1) + \Phi(v_2)$$

by rule (d). Suppose f is a quadrangle $v_1v_2w_2w_1$. Then $\{w_1,w_2\} \not\subset V_3$ by the definition of $\sim$. Then

$$t(v_1,w_1) + t(v_2,w_2) = 4\varepsilon_0 = \sum_{i=1}^{2} (\Phi(v_i) - t(v_i,u))$$

by rules (c), (f), and (g).

(c) Let f_1, f_2, f_3 be the faces adjacent to the triangle $v_0v_1v_2$. We may assume that $k = \deg(f_1) \leq l = \deg(f_2) \leq m = \deg(f_3)$. We may also assume that $\{v_0,v_1,v_2\} \cap V^+ \neq \emptyset$. Therefore $4 \leq k \leq 6$. Note that $k + m \geq k + l \geq M + 4$ by Lemma 4. If $k \geq 5$,

$$\sum_{i=0}^{2} \Phi(v_i) = -\frac{1}{2} + \frac{2}{k} + \frac{2}{l} + \frac{2}{m} \leq -\frac{1}{2} + \frac{2}{5} + 4\varepsilon_1 \leq 0.$$

Suppose $k = 4$. We may assume that v_1v_2 is incident with a quadrangle. Then

$$t(v_1) + t(v_2) = 4\varepsilon_0 = \sum_{i=0}^{2} \Phi(v_i).$$

(d) Note that

$$\sum_{i=1}^{2} (\Phi(v_i) - t(v_i)) = 4\varepsilon_0$$

by rule (c). Let f be the face incident with v_3v_4 other than $v_1v_2v_4v_3$. If $\deg(f) \geq 5$,

$$\sum_{i=1}^{4} \Phi(v_i) - \sum_{i=1}^{2} t(v_i) \leq 4\varepsilon_0 + 2\gamma(4,5,M)$$
$$= 6\varepsilon_0 - \frac{1}{10} \leq 0.$$

Suppose f is a quadrangle. Then

$$\sum_{i=1}^{4} \Phi(v_i) - \sum_{i=1}^{2} t(v_i) = 6\varepsilon_0 = \sum_{i=3}^{4} t(v_i)$$

by rules (h) and (i).

(e) Similar to case (d), since $\sum_{i=0}^{2} \Phi(v_i) = 4\varepsilon_0$.

(f) Let f be the face incident with v_5v_6 other than $v_3v_4v_6v_5$. By Lemma 6(b) and (c), $\deg(f) \geq 6$. Therefore

$$\sum_{i=1}^{6} \Phi(v_i) - \sum_{i=1}^{2} t(v_i) \leq 6\varepsilon_0 + 2\gamma(4,6,M) = 8\varepsilon_0 - \frac{1}{6} < 0.$$

(g) Similar to case (f).

(h) $\Phi(v_1) \leq \varepsilon_1 = t(v_1, v_2)$.

(i) $\Phi(v_1) + \Phi(v_2) \leq 2\varepsilon_1 < 4\varepsilon_1 = \sum_{i=1}^{2}(t(v_i,u_i) + t(v_i,w_i))$ by rules (j) and (k).

(j) Let f be the face incident with w_1w_2 other than $v_1v_2w_2w_1$. If $\deg(f) \geq 5$,

$$\sum_{i=1}^{2}(\Phi(v_i) + \Phi(w_i)) \leq 2\varepsilon_1 + 2\gamma(4,5,M-1)$$
$$< 2\varepsilon_1 = \sum_{i=1}^{2} t(v_i).$$

Suppose f is a quadrangle. Then

$$\sum_{i=1}^{2}(\Phi(v_i) + \Phi(w_i)) \leq 4\varepsilon_1 = \sum_{i=1}^{2}(t(v_i) + t(w_i)).$$

(k) Let f_1 be the face incident with u_1u_2 other than $u_1u_2v_2v_1$ and f_2 the face incident with w_1w_2 other than $v_1v_2w_2w_1$. We may assume that $\deg(f_1) \leq \deg(f_2)$. If $\deg(f_1) \geq 5$,

$$\sum_{i=1}^{2}(\Phi(u_i) + \Phi(v_i) + \Phi(w_i)) \leq 2\varepsilon_1 + 4\gamma(4,5,M-1)$$
$$= 6\varepsilon_1 - \frac{1}{5} < 0.$$

Suppose $\deg(f_1) = 4$. Then $\deg(f_2) \geq 5$ by Lemma 6(d). Therefore

$$\sum_{i=1}^{2}(\Phi(u_i) + \Phi(v_i) + \Phi(w_i) - t(u_i))$$
$$\leq 4\varepsilon_1 + 2\gamma(4,5,M-1) - 2\varepsilon_1$$
$$= 4\varepsilon_1 - \frac{1}{10} < 0.$$

(l) By Lemma 6(d), u_1u_2 and x_1x_2 are incident with a face of size ≥ 5. Therefore

$$\begin{aligned}\sum_{i=1}^{2}(\Phi(u_i)+\Phi(v_i)+\Phi(w_i)+\Phi(x_i)) &\leq 4\varepsilon_1+4\gamma(4,5,M-1)\\ &= 8\varepsilon_1-\frac{1}{5}<0.\end{aligned}$$

This completes the proof of Claim 1. □

In the rest of the proof, suppose $\deg(v)=n\geq 4$, $N(v)=\{v_1,\ldots,v_n\}$, the edge vv_i is incident with f_{i-1} and f_i (taking suffices mod n), $d_i=\deg(f_i)$, and $t_i=t(v_i,v)$.

CLAIM 2. $\Phi'(v)\leq 0$ *when* $n=4$.

Proof. Note that at most two triangles are incident with v by Lemma 6(g).

Case 1. $d_1=d_2=3$.

By Lemma 4, $\deg(v_2)\geq 4$. Then by Lemma 5, $\deg(v_1)\geq 4$ and $\deg(v_3)\geq 4$. Therefore

$$\begin{aligned}\Phi'(v)=\Phi(v)+t_4 &\leq \gamma(3,3,4,M-1)+3\varepsilon_0\\ &< 4\varepsilon_1-\frac{1}{12}<0.\end{aligned}$$

Case 2. $d_1=d_3=3$.

By symmetry, it is enough to show that $\Phi(v)/2+t_1+t_2\leq 0$. Since

$$\Phi(v)\leq\gamma(3,3,4,M-1)=\varepsilon_1-\frac{1}{12}<0,$$

we may assume that $t_1>0$. In particular, $\deg(v_1)=3$. If $\deg(v_2)\geq 4$, $d_2=M$ by Lemma 5. It is easily seen that $t_1>1/24-\varepsilon_1/2$ only if rule (a) is applied. Since $d_4=M$ in this case,

$$\Phi(v)=\gamma(3,3,M,M)=2\varepsilon_0-\frac{1}{3},$$

and

$$\frac{1}{2}\Phi(v)+t_1\leq 3\varepsilon_0-\frac{1}{8}<0.$$

Suppose $\deg(v_2)=3$ and let f be the face incident with v_1v_2 other than vv_1v_2. Let $d'=\deg(f)$. Since $t_1>0$, either $4\leq d'\leq 6$ or $4\leq d_4\leq 6$. If $4\leq d'\leq 6$,

$$d_i\geq M+4-d'$$

for $i=2,4$. Then

$$\Phi(v)\leq\gamma(3,3,M-2,M-2)=2\varepsilon_2-\frac{1}{3},$$

and therefore

$$t_1+t_2\leq 2\left(\frac{1}{12}-\varepsilon_0\right)<-\frac{1}{2}\Phi(v).$$

If $4 \le d_4 \le 6$,

$$\begin{aligned}\Phi(v_1) + \Phi(v_2) &\le \gamma(3,4,M) + \gamma(3,M-2,M-3)\\ &\le 3\varepsilon_3 - \frac{1}{12} < 0.\end{aligned}$$

Case 3. $d_1 = 3$, $d_i \ge 4$ $(2 \le i \le 4)$

Note that $t_3 + t_4 \le 6\varepsilon_0 + \varepsilon_1$. In fact, suppose $t_3 + t_4 > 6\varepsilon_0 + \varepsilon_1$. We may assume that $t_4 > 3\varepsilon_0$. Then $\{d_3, d_4\} = \{4, M\}$. If $d_3 = 4$, it is easily seen that $t_3 \le \varepsilon_1$. Suppose $d_3 = M$ and $d_4 = 4$. By Lemma 6(a), $t_4 \ne 6\varepsilon_0$. Therefore the only possibility is $t_4 = 4\varepsilon_0$. On the other hand, $t_3 \ne 6\varepsilon_0$ by Lemma 6(a), and $t_3 \ne 4\varepsilon_0$ by Lemma 6(a").

Since

$$\Phi(v) \le \gamma(3,4,4,M-2) = \varepsilon_2 - \frac{1}{6} < -(6\varepsilon_0 + \varepsilon_1),$$

we may assume that $\deg(v_1) = 3$. First, suppose $\deg(v_2) \ge 4$. Then $d_2 = M$ by Lemma 5. The only possibility that $t_1 > 1/6 - 8\varepsilon_2$ is $t_1 = 1/24 + 2\varepsilon_0$. This is possible only if $d_4 = M$. Therefore

$$\begin{aligned}\Phi(v) \le \gamma(3,4,M,M) &= 2\varepsilon_2 - \frac{5}{12}\\ &< -\left(6\varepsilon_0 + \varepsilon_1 + \frac{1}{24} + 2\varepsilon_0\right).\end{aligned}$$

Finally, suppose $\deg(v_2) = 3$. Let f be the face incident with v_1v_2 other than vv_1v_2, and $d' = \deg(f)$. We may assume that $4 \le d' \le 6$ or $4 \le d_4 \le 6$. If $4 \le d' \le 6$,

$$t_1 + t_2 \le 2\left(\frac{1}{12} - \varepsilon_0\right),$$

since $1/12 - \varepsilon_0 > 1/30 + \varepsilon_1$. Therefore

$$\begin{aligned}\Phi(v) + \sum_{i=1}^{4} t_i &\le \gamma(3,4,M-2,M-2) + 2\left(\frac{1}{12} - \varepsilon_0\right) + (6\varepsilon_0 + \varepsilon_1)\\ &< 7\varepsilon_2 - \frac{1}{4} < 0.\end{aligned}$$

Suppose $5 \le d_4 \le 6$ and $d_4 \le d_2$. Then

$$t_1 + t_2 \le 2\left(\frac{1}{30} + \varepsilon_1\right),$$

and

$$\Phi(v) \le \gamma(3,4,5,M-3) = \varepsilon_3 - \frac{13}{60}.$$

Note that $t_4 = 0$. Suppose

$$t_3 > -\Phi(v) - t_1 - t_2 > \frac{3}{20} - 3\varepsilon_3 > 4\varepsilon_1.$$

The only possibility is $t_3 = 6\varepsilon_0$. In particular, $d_2 = M$. Then

$$\begin{aligned}\Phi(v_1) + \Phi(v_2) &\le \gamma(3,5,M-1) + \gamma(3,M-2,M)\\ &= -\frac{2}{15} + \varepsilon_2 + \varepsilon_1 + \varepsilon_0 < 0,\end{aligned}$$

and

$$\Phi(v) + t_3 \le \gamma(3,4,5,M) + 6\varepsilon_0 = 7\varepsilon_0 - \frac{13}{60} < 0.$$

Suppose $d_4 = 4$. By Lemma 6(f), $d_2 \ge 5$. If $d_2 \ge 6$,

$$t_1 + t_2 \le \frac{1}{12} + \varepsilon_0 + \varepsilon_2,$$

and

$$\Phi(v) \le \gamma(3,4,6,M-4) = \varepsilon_4 - \frac{1}{4}.$$

If $t_3 = 0$ or $t_4 = 0$,

$$\Phi(v) + \sum_{i=1}^{4} t_i \le 9\varepsilon_4 - \frac{1}{6} < 0.$$

If $t_3 > 0$ and $t_4 > 0$, then $d_3 = 4$ and $d_2 \ge M-1$. In this case,

$$\begin{aligned}\Phi(v_1) + \Phi(v_2) &\le \gamma(3,4,M) + \gamma(3,M-1,M)\\ &< 3\varepsilon_1 - \frac{1}{12} < 0.\end{aligned}$$

Therefore $t_1 = t_2 = 0$, and

$$\begin{aligned}\Phi(v) + \sum_{i=1}^{4} t_i &\le \gamma(3,4,4,M-1) + 6\varepsilon_0 + \varepsilon_1\\ &< 8\varepsilon_1 - \frac{1}{6} < 0.\end{aligned}$$

Hence we may assume that $d_2 = 5$. Then $t_3 = 0$. Suppose

$$\begin{aligned}t_4 &> -\Phi(v) - t_1 - t_2\\ &\ge \frac{13}{60} - \varepsilon_3 - \left(\frac{1}{12} + \varepsilon_0\right) - \left(\frac{1}{30} + \varepsilon_1\right)\\ &> \frac{1}{10} - 3\varepsilon_3 > 2\varepsilon_1.\end{aligned}$$

The only possibility is $d_3 = M$ and $t_4 \ge 4\varepsilon_0$. However, $t_4 \ne 6\varepsilon_0$ by Lemma 6(a), and $t_4 \ne 4\varepsilon_0$ by Lemma 6(e).

Case 4. $d_i \ge 4$ $(1 \le i \le 4)$.

Since $\Phi(v) \le \gamma(4,4,4,M-3) = \varepsilon_3 - 1/4 < 0$, we may assume that $t_1 > 1/4\,(1/4 - \varepsilon_3) > 3\varepsilon_0$. Hence we may assume that $d_1 = 4$, $d_4 = M$, and $\deg(v_2) = 3$. Note that $t_4 < 3\varepsilon_0$ if $t_1 = 6\varepsilon_0$, and $t_4 \le 4\varepsilon_0$ if $t_1 = 4\varepsilon_0$ by Lemma 6(a'). Hence $t_1 + t_4 <$

$9\varepsilon_0$. Moreover, $t_2 \le \varepsilon_1$. If $t_3 \le 3\varepsilon_0$, $\Phi(v) + \sum_{i=1}^{4} t_i < \varepsilon_3 - 1/4 + 9\varepsilon_0 + \varepsilon_1 + 3\varepsilon_0 < 0$. Suppose $t_3 > 3\varepsilon_0$. Then $d_2 = M$ or $d_3 = M$. In either case,

$$\begin{aligned}\Phi(v) + \sum_{i=1}^{4} t_i &\le \gamma(4,4,M,M) + 9\varepsilon_0 + \varepsilon_1 + 6\varepsilon_0 \\ &< 18\varepsilon_1 - \frac{1}{2} < 0.\end{aligned}$$

This completes the proof of Claim 2. □

In the rest of the proof, we assume that $n \ge 5$ and show that $\Phi'(v) \le 0$. Let

$$\sigma_i = -\frac{1}{2} + \frac{1}{2d_{i-1}} + \frac{1}{2d_i} + \frac{1}{n},$$

and $\tau_i = \sigma_i + t_i$. It is easily seen that

$$\sum_{i=1}^{n} \sigma_i = \Phi(v),$$

and

$$\sum_{i=1}^{n} \tau_i = \Phi'(v).$$

Suppose, on the contrary, that $\Phi'(v) > 0$. Then $\tau_i > 0$ for some i.

Claim 3. *Suppose $\tau_i > 0$, and $d_{i-1} \ge d_i$. Suppose furthermore that $d_{i-1} \le d_{i+1}$ if $\deg(v_i) = \deg(v_{i+1}) = 3$. Then one of the following holds:*

(a) *$n = 5$, $d_{i-1} = d_i = 3$, and $\tau_i = 1/30$.*

(b) *$n = 5$, $\deg(v_i) = 3$, $\deg(v_{i+1}) \ge 4$, $d_i = 3$, $d_{i-1} = 4$, and $\tau_i = 1/30 - \varepsilon_0$.*

(c) *$n = 5$, $\deg(v_i) = \deg(v_{i+1}) = 3$, $d_i = 3$, $d_{i-1} = 5$, and $\tau_i \le \varepsilon_1$.*

(d) *$5 \le n \le 9$, $\deg(v_i) = \deg(v_{i+1}) = 3$, $d_i = 3$, $d_{i-1} = 4$, and $\tau_i \le -13/120 + 1/n$.*

Proof. First, suppose $t_i = 0$. If $d_{i-1} \ge 4$,

$$\tau_i = \sigma_i \le -\frac{1}{2} + \frac{1}{2 \cdot 3} + \frac{1}{2 \cdot 4} + \frac{1}{n} \le -\frac{1}{120}.$$

If $d_{i-1} = d_i = 3$,

$$\tau_i = -\frac{1}{2} + \frac{1}{6} + \frac{1}{6} + \frac{1}{n} = -\frac{1}{6} + \frac{1}{n}.$$

In this case, $\tau_i > 0$ only if $n = 5$.

Next suppose $t_i > 0$. Then $\deg(v_i) = 3$ and $d_i \le 4$. Suppose $d_i = 4$. Then $d_{i-1} = 4$ or $d_{i-1} \ge M - 1$. If $d_{i-1} = 4$,

$$\begin{aligned}\tau_i &\le -\frac{1}{2} + \frac{1}{8} + \frac{1}{8} + \frac{1}{n} + \varepsilon_1 \\ &\le -\frac{1}{20} + \varepsilon_1 < 0.\end{aligned}$$

If $d_{i-1} \ge M - 1$,

$$\begin{aligned}\tau_i &\le -\frac{1}{2} + \frac{1}{8} + \frac{1}{2(M-1)} + \frac{1}{n} + 6\varepsilon_0 \\ &\le -\frac{7}{40} + 7\varepsilon_0 < 0.\end{aligned}$$

In the rest of the proof of Claim 3, we assume $d_i = 3$ and let d' be the size of the face incident with the edge v_iv_{i+1} other than vv_iv_{i+1}. If $4 \le d' \le 6$, then $d_{i-1} \ge M-2$ and $t_i \le \max\{1/12-\varepsilon_0, 1/24+2\varepsilon_0\}$. (Note that $1/12-\varepsilon_0 > 1/30+\varepsilon_1$.) Hence

$$\tau_i \le -\frac{1}{2}+\frac{1}{6}+\frac{1}{2(M-2)}+\frac{1}{n}+\max\left\{\frac{1}{12}-\varepsilon_0, \frac{1}{24}+2\varepsilon_0\right\} < -\frac{1}{24}.$$

Suppose $d' \ge 7$. Then $4 \le d_{i-1} \le 6$. If $\deg(v_{i+1}) > 3$,

$$\begin{aligned}\tau_i &< -\frac{1}{2}+\frac{1}{6}+\frac{1}{8}+\frac{1}{n}+\max\left\{\frac{1}{24}-\varepsilon_0, \frac{1}{2}\left(\frac{1}{30}+\varepsilon_1\right)\right\}\\ &\le -\frac{1}{6}+\frac{1}{n}.\end{aligned}$$

Therefore $\tau_i > 0$ only if $n = 5$. Moreover,

$$\tau_i < 0 \quad \text{if} \quad d_{i-1} \ge 5,$$

and

$$\tau_i = \frac{1}{30}-\varepsilon_0 \quad \text{if} \quad d_{i-1} = 4.$$

Finally, suppose $\deg(v_{i+1}) = 3$. If $d_{i-1} = 6$,

$$\begin{aligned}\tau_i &\le -\frac{1}{2}+\frac{1}{6}+\frac{1}{12}+\frac{1}{n}+\varepsilon_2\\ &\le -\frac{1}{20}+\varepsilon_2 < 0.\end{aligned}$$

If $d_{i-1} = 5$,

$$\begin{aligned}\tau_i &\le -\frac{1}{2}+\frac{1}{6}+\frac{1}{10}+\frac{1}{n}+\frac{1}{30}+\varepsilon_1\\ &= -\frac{1}{5}+\frac{1}{n}+\varepsilon_1.\end{aligned}$$

Therefore $\tau_i > 0$ only if $n = 5$. If $d_{i-1} = 4$,

$$\begin{aligned}\tau_i &\le -\frac{1}{2}+\frac{1}{6}+\frac{1}{8}+\frac{1}{n}+\frac{1}{12}+\varepsilon_0\\ &\le -\frac{13}{120}+\frac{1}{n}.\end{aligned}$$

Therefore $\tau_i > 0$ only if $n \le 9$. □

By Claim 3, we may assume that $n \le 9$. We may also assume that $\tau_1 > 0$ (by renumbering the vertices in $N(v)$ if necessary). Note that $d_1 + d_n \le 8$ in all cases of Claim 3. Let $T^+ = \sum_{\tau_i>0}\tau_i$ and $T^- = \sum_{\tau_i<0}\tau_i$. Then $\Phi'(v) = T^+ + T^-$.

By Lemma 4,

$$\mathrm{cd}(v) = \sum_{i=1}^{n} d_i - 2n \ge M+1.$$

Hence

$$\sum_{i=2}^{n-1} d_i \ge 2n+53.$$

This implies that $d_p \geq \lceil \frac{2n+53}{n-2} \rceil$ for some p, $2 \leq p \leq n-1$.

CLAIM 4. *Suppose* $d_p \geq \lceil \frac{2n+53}{n-2} \rceil$. *Then* $\tau_p < 0$. *More precisely, the following holds:*

(a) *If* $t_p = 0$, $\tau_p \leq -23/210$ *when* $n = 5$ *and* $\tau_p \leq -7/51$ *when* $n \geq 6$.
(b) *If* $d_{p-1} = 4$, $\min\{\tau_p, \tau_{p-1} + \tau_p\} \leq -3/10 + 1/n$.
(c) *If* $d_{p-1} = 3$ *and* $\deg(v_{p-1}) = \deg(v_p) = 3$, $\min\{\tau_p, \tau_{p-1} + \tau_p\} \leq -31/60 + 2/n$.
(d) *If* $d_{p-1} = 3$, $\deg(v_{p-1}) \geq 4$ *and* $\deg(v_p) = 3$, $\tau_p \leq -1/4 + 1/n$.

Proof. Suppose $t_p = 0$. Then

$$\tau_p = \sigma_p \leq -\frac{1}{2} + \frac{1}{6} + \frac{1}{2d_p} + \frac{1}{n} \leq \begin{cases} -\dfrac{23}{210}, & n = 5, \\ -\dfrac{7}{51}, & n \geq 6. \end{cases}$$

This proves (a).

Suppose $t_p > 0$. Then $\deg(v_p) = 3$ and $d_{p-1} \leq 4$. Suppose $d_{p-1} = 4$. If $t_p = 6\varepsilon_0$, $d_{p-2} \neq 3$ by Lemma 6(a). Then

$$\tau_p = -\frac{1}{2} + \frac{1}{8} + \frac{1}{2M} + \frac{1}{n} + 6\varepsilon_0 \leq -\frac{4}{15} + \frac{1}{n},$$

and

$$\tau_{p-1} \leq -\frac{1}{2} + \frac{1}{8} + \frac{1}{8} + \frac{1}{n} + \varepsilon_0 \leq -\frac{7}{30} + \frac{1}{n},$$

since $t_{p-1} \leq \varepsilon_0$. Hence

$$\tau_{p-1} + \tau_p \leq -\frac{1}{2} + \frac{2}{n} \leq -\frac{3}{10} + \frac{1}{n}.$$

If $t_p \leq 4\varepsilon_0$,

$$\tau_p \leq -\frac{1}{2} + \frac{1}{8} + \frac{1}{2M} + \frac{1}{n} + 4\varepsilon_0 \leq -\frac{3}{10} + \frac{1}{n}.$$

This proves (b).

Suppose $d_{p-1} = 3$ and let d' be the size of the face incident with $v_{p-1}v_p$ other than $vv_{p-1}v_p$. If $t_p > 0$, $4 \leq d' \leq 6$. Suppose $\deg(v_{p-1}) = 3$. Then

$$\tau_p \leq -\frac{1}{2} + \frac{1}{6} + \frac{1}{2M} + \frac{1}{n} + \frac{1}{12} - \varepsilon_0 \leq -\frac{31}{120} + \frac{1}{n},$$

$$\tau_{p-1} + \tau_p \leq 2\left(-\frac{31}{120} + \frac{1}{n}\right) = -\frac{31}{60} + \frac{2}{n}.$$

Suppose $\deg(v_{p-1}) \geq 4$. Then

$$\tau_p \leq -\frac{1}{2} + \frac{1}{6} + \frac{1}{2M} + \frac{1}{n} + \frac{1}{24} + 2\varepsilon_0 \leq -\frac{1}{4} + \frac{1}{n}.$$

This completes the proof of Claim 4. □

Note that the same claim holds for $\min\{\tau_{p+1}, \tau_{p+1} + \tau_{p+2}\}$.

Suppose $n \geq 6$. Since $\tau_p < 0$ and $\tau_{p+1} < 0$,

$$T^+ \leq \left(-\frac{13}{120} + \frac{1}{n}\right) \times (n-2) \leq \frac{7}{30}.$$

Note that $\min\{\tau_p, \tau_{p-1} + \tau_p\} \leq -2/15$ in cases (a), (b), and (c) of Claim 4. If $\deg(v_{p-1}) = \deg(v_{p+2}) = 3$, $T^- \leq -2/15 \times 2$. Then $T^+ + T^- \leq -1/30$. Suppose $\deg(v_{p-1}) \geq 4$ and $\deg(v_{p+2}) \geq 4$. Then

$$T^+ \leq \left(-\frac{13}{120} + \frac{1}{n}\right) \times (n-4) \leq \frac{7}{60},$$

and

$$T^- \leq -\frac{1}{12} \times 2 = -\frac{1}{6}.$$

Hence $T^+ + T^- \leq -1/20$. In the remaining case, case (d) of Claim 4 applies for exactly one of $(p-1, p)$ and $(p+1, p+2)$. Therefore

$$T^+ \leq \left(-\frac{13}{120} + \frac{1}{n}\right) \times (n-3) \leq \frac{7}{40},$$

and

$$T^- \leq -\frac{1}{12} - \frac{2}{15} = -\frac{13}{60}.$$

Hence $T^+ + T^- \leq -1/24$. This completes the proof of Theorem 2 when $n \geq 6$.

Finally, suppose $n = 5$ and, without loss of generality, $p = 5$. If the situation of Claim 3(d) is not possible, then

$$T^+ + T^- \leq 3 \cdot \frac{1}{30} - 2 \cdot \frac{1}{20} = 0.$$

Thus in what follows we suppose without loss of generality that $\deg(v_2) = \deg(v_3) = 3$, $d_2 = 3$ and $\min\{d_1, d_3\} = 4$.

Suppose first $d_1 + d_3 \geq 9$. Then

$$\tau_2 + \tau_3 \leq -\frac{1}{2} + \frac{1}{6} + \frac{1}{8} + \frac{1}{5} - \frac{1}{2} + \frac{1}{6} + \frac{1}{10} + \frac{1}{5} = -\frac{1}{24},$$
$$\Phi(v_2) + \Phi(v_3) \leq \gamma(3,4,M) + \gamma(3,5,M) = \frac{7}{60} + 2\varepsilon_0,$$

and therefore

$$\tau_2 + \tau_3 \leq \frac{13}{120}.$$

If $\deg(v_4) \geq 4$, then

$$T^+ + T^- \leq \frac{13}{120} + \frac{1}{30} - \frac{1}{10} - \frac{1}{20} = -\frac{1}{120},$$

and if $\deg(v_4) = 3$, then

$$T^+ + T^- \leq \frac{13}{120} + \frac{11}{120} - 2 \cdot \frac{1}{10} = 0.$$

Finally, suppose $d_1 = d_3 = 4$. If the situation of Claim 4(d) applies, then $\deg(v_4) \geq 4$, $\deg(v_5) = 3$ and $d_4 = 3$. This contradicts Lemma 5. Now we suppose that $\min\{\tau_1, \tau_1 + \tau_2\} = \tau_1 \leq -1/10$ and $\min\{\tau_5, \tau_5 + \tau_4\} \leq -1/10$. If $\tau_4 \leq 0$, then

$$T^+ + T^- \leq 2 \cdot \frac{11}{120} - 2 \cdot \frac{1}{10} = -\frac{1}{60}.$$

On the other hand, $\tau_4 > 0$ only if $\deg(v_4) = 3$ and $d_4 = 3$. If $\deg(v_5) \geq 4$,

$$\tau_5 = \sigma_5 \leq -\frac{1}{2} + \frac{1}{6} + \frac{1}{2(M-3)} + \frac{1}{5} < -\frac{7}{60},$$

and therefore

$$T^+ + T^- < 2 \cdot \frac{11}{120} + \frac{1}{30} - \varepsilon_0 - \frac{7}{60} - \frac{1}{10} = -\varepsilon_0.$$

If $\deg(v_5) = 3$,

$$\begin{aligned} \Phi(v_5) + \Phi(v_4) &\leq \gamma(3, M-3, M) + \gamma(3, 4, M) \\ &= -\frac{1}{12} + 2\varepsilon_0 + \varepsilon_3 < 0, \end{aligned}$$

and so $t_4 = 0$ and $\tau_4 < 0$.

This completes the proof of Theorem 2. □

REFERENCES

[1] K. Ando, H. Enomoto, and A. Saito, *Contractible edges in 3-connected graphs*, J. Combin. Theory Ser. B, 42 (1987), pp. 87–93.

[2] O.V. Borodin and D.R. Woodall, *Cyclic coloring of 3-polytopes with large maximum face size*, SIAM J. Discrete Math., submitted.

[3] R. Halin, *Untersuchungen über minimale n-fach zusammenhängende Graphen*, Math. Ann., 182 (1969), pp. 175–188.

[4] M. Horňák and S. Jendrol', *On a conjecture by Plummer and Toft*, J. Graph Theory, 30 (1999), pp. 177–189.

[5] T. Jensen and B. Toft, *Graph Coloring Problems*, John Wiley, New York, 1995.

[6] O. Ore and M.D. Plummer, *Cyclic coloration of plane graphs*, in Recent Progress in Combinatorics, W.T. Tutte, ed., Academic Press, New York, 1969, pp. 287–293.

[7] M.D. Plummer and B. Toft, *Cyclic coloration of 3-polytopes*, J. Graph Theory, 11 (1987), pp. 507–515.

SIAM J. DISCRETE MATH.
Vol. 14, No. 1, pp. 138–141

APPROXIMATE EDGE SPLITTING*

MICHEL X. GOEMANS†

Abstract. We show that, in any undirected graph, splitting-off can be performed while preserving all cuts of value at most 4/3 times the minimum value, and this is the best possible. This generalizes a classical splitting-off result of Lovász.

Key words. graph connectivity, edge splitting

AMS subject classification. 05C40

PII. S0895480199358023

1. Introduction.

In an undirected graph, splitting off two edges incident to a vertex s, say (s,u) and (s,v), means deleting them and adding the edge (u,v). Classical splitting-off theorems, such as those of Lovász [5] (exercise 6.53) and Mader [6], show that splitting off can be performed while preserving certain connectivity properties of the graph. Edge splitting is an important operation for connectivity problems. For example, suppose we would like to make a graph $G=(V,E)$ k-edge-connected by adding the minimum number of edges. A beautiful result of Frank [2] shows that it is sufficient to add a vertex s to the graph, add the minimum even number of edges between s and V to make it k-edge-connected (and this is an easy task), and finally perform splitting off while preserving k-edge-connectivity between the vertices in V (using Lovász's splitting-off result). For extensions of this result, see [2] and the survey [3]. As another (less algorithmic) illustration of the use of edge splitting, Nagamochi, Nishimura, and Ibaraki [7] have shown inductively using edge splitting that there are at most $\binom{n}{2}$ cuts of value strictly less than 4/3 times the minimum cut value in any undirected graph on n vertices. (See [4] for a sketch of a more direct proof.)

To describe the result, we need the following notation. Let $G=(V,E)$ be an undirected graph, possibly with multiple edges. For any set $S \subset V$, let $\delta(S)$ be the set of edges with exactly one endpoint in S, and let $d(S)=|\delta(S)|$ be the value of the corresponding cut. For simplicity, we write $d(s)$ for $d(\{s\})$ for any vertex s. Also, we let $d(s,A)$ denote $|\{(s,u)\in E : u\in A\}|$ (with repetitions counted if there are multiple edges).

When splitting off two edges (s,u) and (s,v), observe that the value of any cut $\delta(S)$ does not increase, and decreases precisely if $u,v\in S$ and $s\notin S$ (or similarly with S replaced by its complement $\bar{S}$). Let λ' denote the minimum edge-connectivity between any two vertices distinct from s, i.e., $\lambda'=\min_{\emptyset\neq S\subset V'} d(S)$, where $V'=V-s$, and let N be the neighbor set of s, i.e., $N=\{u\in V' : (s,u)\in E\}$.

The classical splitting-off result of Lovász [5] (exercise 6.53) shows that if $\lambda'\geq 2$ and $d(s)$ is even, then for any $u\in N$, there exists an edge (s,v) such that splitting off (s,u) and (s,v) does not reduce λ'. Since splitting off changes the value of cuts

*Received by the editors June 24, 1999; accepted for publication (in revised form) September 5, 2000; published electronically January 31, 2001.
http://www.siam.org/journals/sidma/14-1/35802.html
†Department of Mathematics, MIT, Cambridge, MA 02139 (goemans@math.mit.edu). This work was done at the University of Waterloo and at CORE, Université catholique de Louvain, and was supported in part by DONET European Community contract ERB FMRX-CT98-0202 and NSF contract 9623859-CCR.

by an even number, Lovász's result can be interpreted as saying that the cuts of minimum value and minimum value plus one can all be preserved while performing splitting off. By repeated applications of Lovász's result, one can isolate any vertex while maintaining the connectivity between the other vertices.

Recently, Benczúr [1] introduced the notion of approximate splitting off in which the goal is to preserve all cuts of value less than α times the minimum, for some value of α. Since the values of the cuts $\delta(S)$ and $\delta(V'-S)$ become identical once s is completely isolated in the graph, we should not always be able to preserve both $d(S)$ and $d(V'-S)$. As a result, we say that (s,u) and (s,v) are *admissible* for k-splitting off if $\min(d(S), d(V'-S))$ (for $\emptyset \neq S \neq V'$) is preserved whenever this quantity is less than k. Using the polygon representation of cuts of value less than $\frac{6}{5}\lambda'$, Benczúr has shown the existence of an admissible pair of edges for $\frac{6}{5}\lambda'$-splitting off when $d(s)$ is even.

In this short note, we show that if s is even, then for any edge (s,u), there exists an edge (s,v) such that this pair of edges is admissible for $\frac{4\lambda'+2}{3}$-splitting off. By repeated applications of this result, one can isolate s while maintaining all cuts of value less than $\frac{4\lambda'+2}{3}$. Observe that for $\lambda' \geq 2$, the values of cuts of value λ' and $\lambda'+1$ are maintained by $(4\lambda'+2)/3$-splitting off since $\lambda'+1 < \frac{4\lambda'+2}{3}$. Thus, our result generalizes Lovász's result.

THEOREM 1.1. *For any vertex s with $d(s)$ even and for any edge (s,u), there exists an edge (s,v) such that the pair (s,u) and (s,v) is admissible for $(4\lambda'+2)/3$-splitting off.*

Observe that, for $\lambda'=0$, the statement is vacuous. Similarly, for $\lambda'=1$, the only cuts we need to preserve are of value 1 and this is done by any choice of v (since the cut values do not change or decrease by 2). Thus the only interesting cases are when $\lambda' \geq 2$.

This approximate splitting-off theorem is the best possible. Consider indeed K_5 in which the edges nonadjacent to a specific vertex s are duplicated M times. Then $\lambda' = 3M+1$, but if we split off (s,u) and (s,v), then $d(\{u,v\})$ decreases while $d(\{u,v\}) = 4M+2 = \frac{4\lambda'+2}{3}$. This shows that there is no admissible pair of edges.

By repeatedly using Theorem 1.1, we derive that vertex s can be isolated in the graph, as follows.

COROLLARY 1.2. *If vertex s has even degree, then the edges incident to s can be partitioned into $d(s)/2$ admissible pairs for $(4\lambda'+2)/3$-splitting off.*

2. The proof. In order to prove Theorem 1.1, we first need the following simple lemma [1].

LEMMA 2.1. *Let $d(s) > 0$ be even. Then (s,u) and (s,v) is not an admissible pair for k-splitting off if and only if there exists a set $S \subset V'$ with $u,v \in S$ such that* (i) *$d(S) < k$ and* (ii) *$d(s,S) \leq \frac{1}{2}d(s)$.*

Proof. After splitting off, only sets containing both u and v see their $d(.)$ value change (by 2 units). Thus, for $\min(d(S), d(V'-S))$ not to be preserved, we need either $u,v \in S$ or $u,v \in V-S'$. By complementing S (in V') if needed, we can restrict our attention to sets S with $u,v \in S$. As a result, $\min(d(S), d(V'-S))$ will decrease if and only if $d(S)-2 < d(V'-S)$. Since $d(S)-d(s,S) = d(V'-S)-d(s,V'-S)$ and $d(s,S)+d(s,V'-S) = d(s)$, the condition $d(S)-2 < d(V'-S)$ is equivalent to $2d(s,S)-2 < d(s)$. Since $d(s)$ is even, this is equivalent to $d(s,S) \leq d(s)/2$. This can also be written as $d(S) \leq d(V'-S)$, and the condition $\min(d(S), d(V'-S)) < k$ is therefore equivalent to $d(S) < k$. This proves the lemma. □

Theorem 1.1 follows from Lemma 2.1 and the following result.

LEMMA 2.2. *Let $d(s) > 0$ be even and let $u \in N$. There exists $v \in N$ such that there is no set $S \subset V'$ with* (i) $u, v \in S$, (ii) $d(s, S) \leq \frac{1}{2}d(s)$, *and* (iii) $d(S) < \frac{4\lambda'+2}{3}$.

For the proof of this lemma, we need 3-set submodularity (see [5, exercise 6.48(c)]).

LEMMA 2.3 (3-set submodularity; see [5, ex. 6.48(c)]). *For any 3 sets A, B, C, we have*

$$d(A) + d(B) + d(C) \geq d(A - B - C) + d(B - C - A) + d(C - A - B) + d(A \cap B \cap C) + 2d(V - A - B - C, A \cap B \cap C).$$

3-set submodularity simply follows from evaluating the contribution of any edge to both the left-hand side and the right-hand side.

Proof of Lemma 2.2. We can assume that $\lambda' \geq 2$ and $d(s) \geq 4$ since otherwise the statement is trivial (just take for v the endpoint of an edge (s, v) distinct from (s, u)).

Assume that for every $v \in N$, there exists S_v such that (i) $u, v \in S_v$, (ii) $d(s, S_v) \leq \frac{1}{2}d(s)$, and (iii) $d(S_v) < \frac{4\lambda'+2}{3}$. For a given $v \in N$, we can furthermore assume that S_v is chosen to maximize $d(s, S_v)$ among the sets satisfying (i)–(iii).

First choose $i, j \in N$ such that $d(s, S_i \cup S_j)$ is maximum. Because of (ii), we have that

$$d(s, S_i \cup S_j) = d(s, S_i) + d(s, S_j) - d(s, S_i \cap S_j) \leq \frac{1}{2}d(s) + \frac{1}{2}d(s) - d(s, S_i \cap S_j).$$

Since $u \in S_i \cap S_j$, we have that $d(s, S_i \cap S_j) > 0$, implying that $d(s, S_i \cup S_j) < d(s)$. Thus, there exists $k \in N - (S_i \cup S_j)$.

Since k was not chosen instead of i or j, we have that $(S_i - S_j - S_k) \cap N \neq \emptyset$ and $(S_j - S_i - S_k) \cap N \neq \emptyset$.

By 3-set submodularity, we have that

$$\begin{aligned} 4\lambda' + 2 &> d(S_i) + d(S_j) + d(S_k) \\ &\geq d(S_i - S_j - S_k) + d(S_j - S_i - S_k) + d(S_k - S_i - S_j) \\ &\quad + d(S_i \cap S_j \cap S_k) + 2, \end{aligned}$$

where we have used the fact that the edge (s, u) contributes 1 unit to $d(V - S_i - S_j - S_k, S_i \cap S_j \cap S_k)$. Hence,

$$d(S_i - S_j - S_k) + d(S_j - S_i - S_k) + d(S_k - S_i - S_j) + d(S_i \cap S_j \cap S_k) < 4\lambda',$$

which is a contradiction since all these sets contain an element of N ($k \in S_k - S_i - S_j$ and $u \in S_i \cap S_j \cap S_k$) . □

Although the proof of Theorem 1.1 is essentially existential, one can find pairs of edges incident to v which are admissible for $k\lambda'$-splitting in polynomial time (for k fixed). Indeed, using [7], one can enumerate all cuts of value less than $k\lambda'$ in time $O(nm^2 + n^{2k}m)$, where n is the number of vertices and m is the number of edges, and then check which pairs are admissible. For $k = 4/3$, Nagamochi, Nishimura, and Ibaraki [7] show that all these cuts can in fact be enumerated in time $O(m^2 n + mn^2 \log n)$.

REFERENCES

[1] A. A. Benczúr, *Cut Structures and Randomized Algorithms in Edge-Connectivity Problems*, Ph.D. dissertation, MIT, Cambridge, MA, 1997.

[2] A. Frank, *Augmenting graphs to meet edge-connectivity requirements*, SIAM J. Discrete Math., 5 (1992), pp. 22–53.

[3] A. Frank, *Connectivity augmentation problems in network design*, in Mathematical Programming: State of the Art 1994, J. R. Birge and K. G. Murty, eds., The University of Michigan, Ann Arbor, MI, 1994, pp. 34–63.

[4] M. X. Goemans and V. S. Ramakrishnan, *Minimizing submodular functions over families of sets*, Combinatorica, 15 (1995), pp. 499–513.

[5] L. Lovász, *Combinatorial Problems and Exercises*, North-Holland, Amsterdam, New York, 1979 (2nd ed., 1993).

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

[7] H. Nagamochi, K. Nishimura, and T. Ibaraki, *Computing all small cuts in an undirected network*, SIAM J. Discrete Math., 10 (1997), pp. 469–481.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 143–161

EXTREMAL PROPERTIES FOR DISSECTIONS OF CONVEX 3-POLYTOPES*

JESÚS A. DE LOERA†, FRANCISCO SANTOS‡, AND FUMIHIKO TAKEUCHI§

Abstract. A dissection of a convex d-polytope is a partition of the polytope into d-simplices whose vertices are among the vertices of the polytope. Triangulations are dissections that have the additional property that the set of all its simplices forms a simplicial complex. The size of a dissection is the number of d-simplices it contains. This paper compares triangulations of maximal size with dissections of maximal size. We also exhibit lower and upper bounds for the size of dissections of a 3-polytope and analyze extremal size triangulations for specific nonsimplicial polytopes: prisms, antiprisms, Archimedean solids, and combinatorial d-cubes.

Key words. dissection, triangulation, mismatched region, lattice polytope, combinatorial d-cube, prism, antiprism, Archimedean solid

AMS subject classifications. 52B45, 52B05, 52B70, 52B55

PII. S0895480199366238

1. Introduction.

Let $\mathcal{A}$ be a point configuration in $\boldsymbol{R}^d$ with its convex hull $conv(\mathcal{A})$ having dimension d. A set of d-simplices with vertices in $\mathcal{A}$ is a *dissection* of $\mathcal{A}$ if no pair of simplices has an interior point in common and their union equals $conv(\mathcal{A})$. A dissection is a *triangulation* of $\mathcal{A}$ if in addition any pair of simplices intersects at a common face (possibly empty). The *size* of a dissection is the number of d-simplices it contains. We say that a dissection is *mismatching* when it is not a triangulation (i.e., it is not a simplicial complex). In this paper we study mismatching dissections of maximal possible size for a convex polytope and compare them with maximal triangulations. This investigation is related to the study of Hilbert bases and the hierarchy of covering properties for polyhedral cones which is relevant in algebraic geometry and integer programming (see [5, 10, 24]). Maximal dissections are relevant also in the enumeration of interior lattice points and its applications (see [2, 15] and references therein).

It was first shown by Lagarias and Ziegler that dissections of maximal size turn out to be, in general, larger than maximal triangulations, but their example uses interior points [16]. Similar investigations were undertaken for mismatching minimal dissections and minimal triangulations of convex polytopes [4]. In this paper we augment previous results by showing that it is possible to have *simultaneously*, in the same 3-polytope, that the size of a mismatching minimal (maximal) dissection is smaller (larger) than any minimal (maximal) triangulation. In addition, we show that the gap between the size of a mismatching maximal dissection and a maximal triangulation can grow linearly on the number of vertices and that this occurs already for a family of simplicial convex 3-polytopes. A natural question is how different

*Received by the editors December 22, 1999; accepted for publication (in revised form) December 12, 2000; published electronically February 23, 2001.
http://www.siam.org/journals/sidma/14-2/36623.html

†Department of Mathematics, University of California, Davis, CA 95616 (deloera@math.ucdavis.edu). The research of this author was partially supported by NSF grant DMS-0073815.

‡Departmento de Matemáticas, Estadistica y Comput., Universidat de Cantabria, E-39005 Santander, Cantabria, Spain (santos@matesco.unican.es). The research of this author was partially supported by grant PB97-0358 of the Spanish Dirección General de Investigación Científica y Técnica.

§Department of Information Science, University of Tokyo, Tokyo, 113-0033, Japan (fumi@is.s.u-tokyo.ac.jp).

are the upper and lower bounds for the size of mismatching dissections versus those bounds known for triangulations (see [21]). We prove lower and upper bounds on their size with respect to the number of vertices for dimension three and exhibit examples showing that our technique of proof fails already in dimension four. Here is the first summary of results.

THEOREM 1.1.

(1) *There exists an infinite family of convex simplicial* 3*-polytopes with increasing number of vertices whose mismatching maximal dissections are larger than their maximal triangulations. This gap is linear in the number of vertices* (*Corollary* 2.2).
(2) (a) *There exists a lattice* 3*-polytope with eight vertices containing no other lattice point other than its vertices whose maximal dissection is larger than its maximal triangulations.*
    (b) *There exists a* 3*-polytope with eight vertices for which, simultaneously, its minimal dissection is smaller than minimal triangulations and maximal dissection is larger than maximal triangulations* (*Proposition* 2.3).
(3) *If* D *is a mismatching dissection of a* 3*-polytope with* n *vertices, then the size of* D *is at least* $n-2$. *In addition, the size of* D *is bounded above by* $\binom{n-2}{2}$ (*Proposition* 3.2).

A consequence of our third point is that the result of [4], stating a linear gap between the size of minimal dissections and minimal triangulations, is best possible. The results are discussed in sections 2 and 3.

The last section presents a study of maximal and minimal triangulations for combinatorial d-cubes, three-dimensional prisms and antiprisms, as well as other Archimedean polytopes. The following theorem and table summarize the main results.

THEOREM 1.2.

(1) *There is a constant* $c > 1$ *such that for every* $d \geq 3$ *the maximal triangulation among all possible combinatorial* d*-cubes has size at least* $c^d d!$ (*Proposition* 4.1).
(2) *For a three-dimensional* m*-prism, in any of its possible coordinatizations, the size of a minimal triangulation is* $2m-5+\lceil \frac{m}{2} \rceil$. *For an* m*-antiprism, in any of its possible coordinatizations, the size of a minimal triangulation is* $3m-5$ (*Proposition* 4.3). *The size of a maximal triangulation of an* m*-prism depends on the coordinatization, and in certain natural cases it is* $(m^2+m-6)/2$ (*Proposition* 4.4).
(3) *Table* 1 *specifies sizes of the minimal and maximal triangulations for some Platonic and Archimidean solids. These results were obtained via integer programming calculations using the approach described in* [8]. *All computations used the canonical symmetric coordinatizations for these polytopes* [6]. *The number of vertices is indicated in parenthesis* (*Remark* 4.5).

2. Maximal dissections of 3-polytopes. We introduce some important definitions and conventions: We denote by Q_m a convex m-gon with m an even positive integer. Let v_1v_2 and u_1u_2 be two edges parallel to Q_m, orthogonal to each other, on opposite sides of the plane containing Q_m, and such that the four segments v_iu_j intersect the interior of Q_m. We suppose that v_1v_2 and u_1u_2 are not parallel to any diagonal or edge of Q_m. The convex hull P_m of these points has $m+4$ vertices and it is a simplicial polytope. We will call the north (respectively, south) vertex of Q_m the one which maximizes (respectively, minimizes) the scalar product with the vector

TABLE 1
Sizes of extremal triangulations of Platonic and Archimidean solids.

P	$\lvert T_{min}(P)\rvert$	$\lvert T_{max}(P)\rvert$
Icosahedron (12)	15	20
Dodecahedron (20)	23	36
Cuboctahedron (12)	13	17
Icosidodecahedron (30)	45	?
Truncated Tetrahedron (12)	10	13
Truncated Octahedron (24)	27	?
Truncated Cube (24)	25	48
Small Rhombicuboctahedron (24)	35	?
Pentakis Dodecahedron (32)	54	?
Rhombododecahedron (14)	12	21

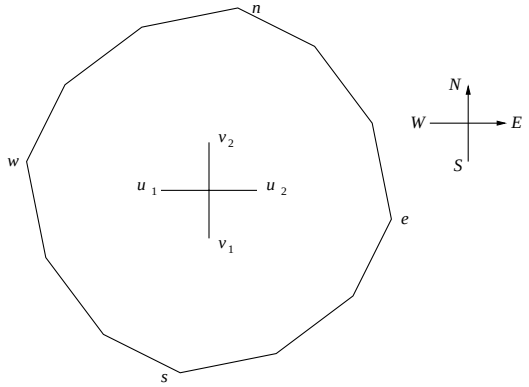


FIG. 1. *North, south, east, and west vertices.*

$v_2 - v_1$. Similarly, we will call east (west) the vertex which maximizes (minimizes) the scalar product with $u_2 - u_1$. We denote these four vertices n, s, e and w, respectively. See Figure 1.

We say that a directed path of edges inside Q_m is *monotone in the direction* v_1v_2 (respectively, u_1u_2) when the vertices of the path appear in the path following the same order given by the scalar product with $v_2 - v_1$ (respectively, $u_2 - u_1$). An equivalent formulation is that any line orthogonal to v_1v_2 cuts the path in at most one point. We remark that by our choice of v_1v_2 and u_1u_2 all vertices of Q_m are ordered by the values of their scalar products with $v_2 - v_1$ and also with respect to $u_2 - u_1$. In the same way, a sequence of vertices of Q_m is *ordered in the direction* of v_1v_2 (respectively, u_1u_2) if the order is the same as the one provided by using the values of the scalar products of the points with the vector $v_2 - v_1$ (respectively, $u_2 - u_1$). Consider the two orderings induced by the directions of v_1v_2 and u_1u_2 on the set of vertices of Q_m. Let us call *horizontal* (respectively, *vertical*) any edge joining two consecutive vertices in the direction of v_1v_2 (respectively, of u_1u_2). As an example, if Q_m is regular, then the vertical edges in Q_m form a zig-zag path as shown in Figure 2.

Our examples in this section will be based on the following observation and are inspired by a similar analysis of maximal dissections of dilated empty lattice tetrahedra in $\boldsymbol{R}^3$ by Lagarias and Ziegler [16]: Let R_m be the convex hull of the $m + 2$ vertices consisting of the m-gon Q_m and v_1, v_2. R_m is exactly one half of the polytope P_m. Consider a triangulation T_0 of Q_m and a path Γ of edges of T_0 monotone with

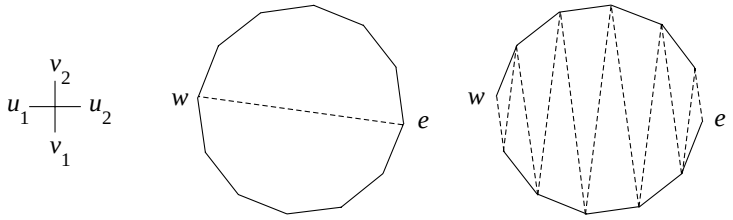


FIG. 2. *The minimal monotone path (middle) and the maximal monotone path made by the vertical edges (right) in the direction u_1u_2.*

respect to the direction u_1u_2. Observe that Γ divides T_0 in two regions, which we will call the "north" and the "south." Then, the following three families of tetrahedra form a triangulation T of R_m: the edges of Γ joined to the edge v_1v_2, the southern triangles of T_0 joined to v_1, and the northern triangles of T_0 joined to v_2 (see Figure 3). Moreover, all the triangulations of R_m are obtained in this way: Any triangulation T of R_m induces a triangulation T_0 of Q_m. The link of v_1v_2 in T is a monotone path of edges contained in T_0 and it divides T_0 in two regions joined, respectively, to v_1 and v_2.

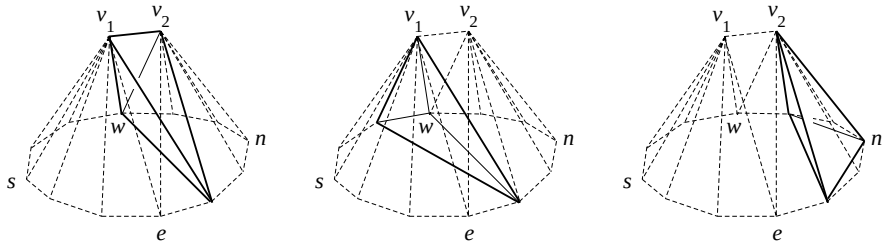


FIG. 3. *Three types of tetrahedra in R_m.*

Using the Cayley trick, one can also think of the triangulations of R_m as the fine mixed subdivisions of the Minkowski sum $Q_m + v_1v_2$ (see [13] and references therein).

The size of a triangulation of R_m equals $m - 2 + |\Gamma|$, where $|\Gamma|$ is the number of edges in the path Γ. There is a unique minimal path in Q_m of length one (Figure 2, middle) and a unique maximal path of length $m - 1$ (Figure 2, right). Hence the minimal and maximal triangulations of R_m have, respectively, $m - 1$ and $2m - 3$ tetrahedra. The maximal triangulation is unique, but the minimal one is not: after choosing the diagonal in Γ the rest of the polygon Q_m can be triangulated in many ways. From the above discussion regarding R_m we see that we could independently triangulate each of the two halves of P_m with any number of tetrahedra from $m - 1$ to $2m - 3$. Hence P_m has dissections of sizes going from $2m - 2$ to $4m - 6$. Among the triangulations of P_m, we will call *halving triangulations* those that triangulate the two halves of P_m. Equivalently, the halving triangulations are those which do not contain any of the four edges v_iu_j.

PROPOSITION 2.1. *Let P_m be as described above with Q_m being a regular m-gon. No triangulation of P_m has more than $\frac{7m}{2} + 1$ tetrahedra. On the other hand, there are mismatching dissections of P_m with $4m - 6$ tetrahedra.*

Proof. Let T be a triangulation of P_m. It is an easy application of Euler's formulas for the 3-ball and 2-sphere that the number of tetrahedra in a triangulation of any

3-ball without interior vertices equals the number of vertices plus interior edges minus three (such a formula appears for instance in [9]). Hence our task is to prove that T has at most $\frac{5m}{2}$ interior edges. For this, we classify the interior edges according to how many vertices of Q_m they are incident to. There are only four edges not incident to any vertex of Q_m (the edges v_iu_j, $i,j \in \{1,2\}$). Moreover, T contains at most $m-3$ edges incident to two vertices of Q_m (i.e., diagonals of Q_m), since in any family of more than $m-3$ such edges there are pairs which cross each other. Thus, it suffices to prove that T contains at most $\frac{3m}{2}-1$ edges incident to just one vertex of Q_m, i.e., of the form v_ip or u_ip with $p \in Q_m$.

Let p be any vertex of Q_m. If p equals w or e, then the edges pv_1 and pv_2 are both in the boundary of P_m; for any other p, exactly one of pv_1 and pv_2 is on the boundary and the other one is interior. Moreover, we claim that if pv_i is an interior edge in a triangulation T, then the triangle pv_1v_2 appears in T. This is so because there is a plane containing pv_i and having v_{3-i} as the unique vertex on one side. At the same time the link of pv_i is a cycle going around the edge. Hence v_{3-i} must appear in the link of pv_i. It follows from the above claim that the number of interior edges of the form pv_i in T equals the number of vertices of Q_m other than w and e in the link of v_1v_2. In a similar way, the number of interior edges of the form pu_i in T equals the number of vertices of Q_m other than n and s in the link of u_1u_2. In other words, if we call $\Gamma_u = \mathit{link}_T(v_1v_2) \cap Q_m$ and $\Gamma_v = \mathit{link}_T(u_1u_2) \cap Q_m$ (the u, v in the index and of the vertices are reversed because in this way Γ_u is monotone with respect to u_1u_2, and Γ_v with respect to v_1v_2), then the number of interior edges in T incident to exactly one vertex of Q_m equals $|\mathit{vertices}(\Gamma_v)| + |\mathit{vertices}(\Gamma_u)| - 4$. Our goal is to bound this number. As an example, Figure 4 shows the intersection of Q_m with a certain triangulation of P_m ($m = 12$). The link of v_1v_2 in this triangulation is the chain of vertices and edges $wabu_1nu_2ce$ (the star of v_1v_2 is marked in thick and grey in the figure). Γ_u consists of the chains wab and ce and the isolated vertex n. In turn, the link of u_1u_2 is the chain nv_1s and Γ_v consists of the isolated vertices n and s.

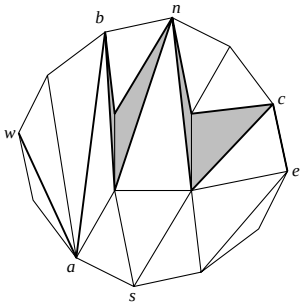


FIG. 4. *Illustration of the proof of Proposition* 2.1.

Observe that Γ_v has at most three connected components because it is obtained by removing from $\mathit{link}_T(u_1u_2)$ (a path) the parts of it incident to v_1 and v_2, if any. Each component is monotone in the direction of v_1v_2 and the projections of any two components to a line parallel to v_1v_2 do not overlap. The sequence of vertices of Q_m ordered in the direction of v_1v_2 can have a pair of consecutive vertices contained in Γ_v only where there is a horizontal edge in Γ_v or in the (at most) two discontinuities of Γ_v. This is true because Q_m is a regular m-gon.

We denote n_{hor} the number of horizontal edges in Γ_v and n'_{hor} this number plus the number of discontinuities in Γ_v (hence $n'_{hor} \leq n_{hor} + 2$). Every nonhorizontal

edge of Γ_v produces a jump of at least two in the v_1v_2-ordering of the vertices of P_m; hence we have

$$|vertices(\Gamma_v)| - 1 - n'_{hor} \leq \frac{m-1-n'_{hor}}{2}.$$

Analogously, and with the obvious similar meaning for n_{vert} and n'_{vert},

$$|vertices(\Gamma_u)| - 1 - n'_{vert} \leq \frac{m-1-n'_{vert}}{2}.$$

Since $\Gamma_u \cup \Gamma_v$ can be completed to a triangulation of Q_m, and exactly four noninterior edges of Q_m are horizontal or vertical, we have $n_{hor}+n_{vert} \leq (m-3)+4 = m+1$, i.e., $n'_{hor} + n'_{vert} \leq m+5$. Hence

$$|vertices(\Gamma_v)| + |vertices(\Gamma_u)| \leq \left\lfloor \frac{2m+2+n'_{hor}+n'_{vert}}{2} \right\rfloor \leq \left\lfloor \frac{3m+7}{2} \right\rfloor = \frac{3m}{2} + 3.$$

Thus, there are at most $\frac{3m}{2} - 1$ interior edges in T of the form pv_i or pu_i and at most $\frac{5m}{2}$ interior edges in total, as desired. □

COROLLARY 2.2. *The polytope P_m described above has the following properties:*

- *It is a simplicial 3-polytope with $m+4$ vertices.*
- *Its maximal dissection has at least $4m-6$ tetrahedra.*
- *Its maximal triangulation has at most $\frac{7m}{2}+1$ tetrahedra.*

In particular, the gap between sizes of the maximal dissection and maximal triangulation is linear on the number of vertices.

Three remarks are in order: First, the size of the maximal triangulation for P_m may depend on the coordinates or, more specifically, on which diagonals of Q_m intersect the tetrahedron $v_1v_2u_1u_2$. Second, concerning the size of the minimal triangulation of P_m, we can easily describe a triangulation of P_m with only $m+5$ tetrahedra: let the vertices n, s, e, and w be as defined above (see Figure 1) and let us call northeast, northwest, southeast, and southwest the edges in the arcs ne, nw, se, and sw in the boundary of Q_m. Then, the triangulation consists of the five tetrahedra $v_1v_2u_1u_2$, $v_1v_2u_1w$, $v_1v_2u_2e$, $v_1u_1u_2s$, and $v_2u_1u_2n$ (shown in the left part of Figure 5) together with the edges v_2u_2, v_2u_1, v_1u_2, and v_1u_1 joined, respectively, to the northeast, northwest, southeast, and southwest edges of Q_m. The right part of Figure 5 shows the result of slicing through the triangulation by the plane containing the polygon Q_m.

Finally, although the corollary above states a difference between maximal dissections and maximal triangulations only for P_m with $m > 14$, experimentally we have observed there is a gap already for $m = 8$. Now we discuss two other interesting examples. The following proposition constitutes the proof of Theorem 1.1 (2).

PROPOSITION 2.3.

(1) *Consider the following eight points in $\boldsymbol{R}^3$:*
   - *The vertices $s = (0,0,0)$, $e = (1,0,0)$, $w = (0,1,0)$, and $n = (1,1,0)$ of a square in the plane $z = 0$.*
   - *The vertices $v_1 = (-1,0,1)$ and $v_2 = (1,1,1)$ of a horizontal edge above the square, and*
   - *the vertices $u_1 = (0,1,-1)$ and $u_2 = (2,0,-1)$ of a horizontal edge below the square.*

   These eight points are the vertices of a polytope P whose only integer points are precisely its eight vertices and with the following properties:

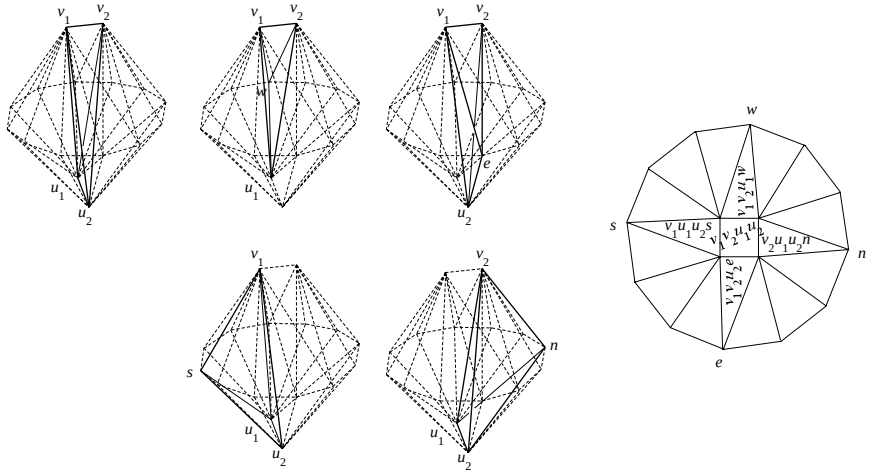


FIG. 5. *For the triangulation of P_m with $m+5$ tetrahedra, its five central tetrahedra (left) and the intersection of the triangulation with the polygon Q_m (right) are shown. The four interior vertices are the intersection points of the edges v_1u_1, v_1u_2, v_2u_1, and v_2u_2 with the plane containing Q_m.*

(a) *Its (unique) maximal dissection has* 12 *tetrahedra. All of them are unimodular, i.e., they have volume* $1/6$.
(b) *Its (several) maximal triangulations have* 11 *tetrahedra.*

(2) *For the* 3*-polytope with vertices* $u_1 = (1,0,0)$, $w = (1,0,1)$, $v_1 = (-1,0,0)$, $s = (-1,0,-1)$, $v_2 = (0,1,1)$, $n = (1,1,1)$, $u_2 = (0,1,-1)$, $e = (-1,1,-1)$, *the sizes of its (unique) minimal dissection and (several) minimal triangulations are* 6 *and* 7*, respectively, and the sizes of its (several) maximal triangulations and (unique) maximal dissection are* 9 *and* 10*, respectively.*

Proof. The polytopes constructed are quite similar to P_4 constructed earlier except that Q_4 is nonregular (in part (2)) and the segments u_1u_2 and v_1v_2 are longer and are not orthogonal, thus ending with different polytopes. The polytopes are shown in Figure 6. Figure 7 describes a maximal dissection of each of them in five parallel slices. Observe that both polytopes have four vertices in the plane $y = 0$ and another four in the plane $y = 1$. Hence the first and last slices in parts (a) and (b) of Figure 7 completely describe the polytope.

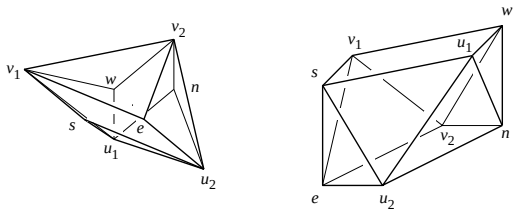


FIG. 6. *The two polytopes in Proposition* 2.3.

(1) The vertices in the planes $y = 0$ and $y = 1$ form convex quadrangles whose only integer points are the four vertices. This proves that the eight points are in convex position and that the polytope P contains no integer point other than its vertices. Let us now prove the assertions on maximal dissections and triangulations

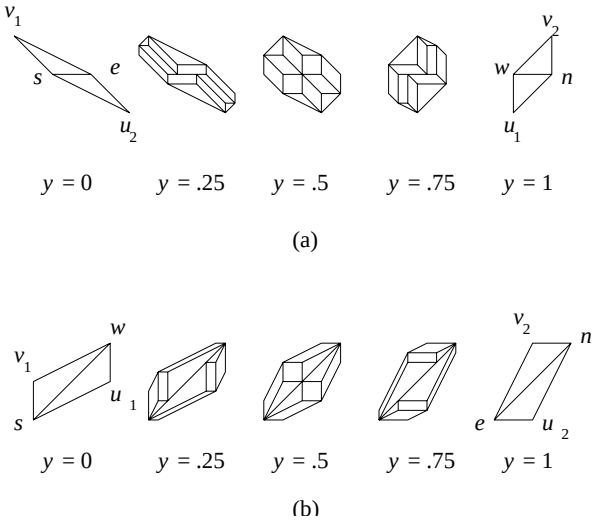


FIG. 7. *Five 2-dimensional slices of the maximal dissections of the polytopes in Proposition* 2.3. *The first and last slices are two facets of the polytopes containing all the vertices.*

of P:

(a) Consider the paths of length three $\Gamma_v = \{esnw\}$ and $\Gamma_u = \{sewn\}$, which are monotone, respectively, in the directions orthogonal to v_1v_2 and u_1u_2. Using them, we can construct two triangulations of size five of the polytopes $conv(nsewv_1v_2)$ and $conv(nsewu_1u_2)$, respectively. However, they do not fill P completely. There is space left for the tetrahedra swv_1u_1 and env_2u_2. This gives a dissection of P with 12 tetrahedra. All the tetrahedra are unimodular, so no bigger dissection is possible.

(b) A triangulation of size 11 can be obtained using the same idea as above, but with paths Γ_v and Γ_u of lengths three and two, respectively, which can be taken from the same triangulation of the square $nswe$.

To prove that no triangulation has bigger size, it suffices to show that P does not have any unimodular triangulation. This means all tetrahedra have volume $1/6$. We start by recalling a well-known fact (see Corollary 4.5 in [25]). A lattice tetrahedron has volume $1/6$ if and only if each of its vertices v lies in a *consecutive* lattice plane parallel to the supporting plane of the opposite facet to v. Two parallel planes are said to be consecutive if their equations are $ax+by+cz = d$ and $ax+by+cz = d-1$.

Suppose that T is a unimodular triangulation of P. We will first prove that the triangle u_1u_2e is in T. The triangular facet u_1u_2s of P, lying in the hyperplane $x + 2y + 2z = 0$, has to be joined to a vertex in the plane $x + 2y + 2z = 1$. The two possibilities are e and v_1. With the same argument, if the tetrahedron $u_1u_2sv_1$ is in T, its facet $u_1u_2v_1$, which lies in the hyperplane $2x + 4y + 3z = 1$, will be joined to a vertex in $2x + 4y + 3z = 2$, and the only one is e. This finishes the proof that u_1u_2e is a triangle in T. Now u_1u_2e is in the plane $x + 2y + z = 1$ and must be joined to a vertex in $x+2y+z = 2$, i.e., to w. Hence u_1u_2ew is in T and, in particular, T uses the edge ew. P is symmetric under the rotation of order two on the axis $\{z = 0, x = \frac{1}{2}\}$. Applying this symmetry to the previous arguments we conclude that T uses the edge ns too. However, this is impossible since the edges ns and ew cross each other.

(2) This polytope almost fits the description of P_4, except for the fact that the edges v_1u_1, v_2u_2 intersect the boundary and not the interior of the planar quadrangle

nsew. With the general techniques we have described, it is easy to construct halving dissections of this polytope with sizes from 6 to 10. Combinatorially, the polytope is a 4-antiprism. Hence, Proposition 4.3 shows that its minimal triangulation has seven tetrahedra. The rest of the assertions in the statement were proved using the integer programming approach proposed in [8], which we describe in Remark 4.5. We have also verified them by enumerating all triangulations [19, 29]. It is interesting to observe that if we perturb the coordinates a little so that the planar quadrilateral $u_1v_1u_2e$ becomes a tetrahedron with the right orientation and without changing the face lattice of the polytope, then the following becomes a triangulation with 10 tetrahedra: $\{u_1u_2se,\ u_1u_2ev_1,\ u_1u_2v_1w,\ u_1u_2wn,\ v_1v_2en,\ v_1v_2nw,\ u_1v_1se,\ v_1u_2ew,\ u_2wne,\ v_1wne\}$. □

3. Bounds for the size of a dissection. Let D be a dissection of a d-polytope P. Say two $(d-1)$-simplices S_1 and S_2 of D *intersect improperly* in a $(d-1)$-hyperplane H if both lie in H, are not identical, and they intersect with a nonempty relative interior. Consider the following auxiliary graph: take as nodes the $(d-1)$-simplices of a dissection, and say that two $(d-1)$-simplices are adjacent if they intersect improperly in a certain hyperplane. A *mismatched region* is the subset of $\boldsymbol{R}^d$ that is the union of $(d-1)$-simplices over a connected component of size larger than one in such a graph. Later, in Proposition 3.4, we will show some of the complications that can occur in higher dimensions.

Define the *simplicial complex of a dissection* as all the simplices of the dissection together with their faces, where only faces that are identical (in $\boldsymbol{R}^d$) are identified. This construction corresponds intuitively to an *inflation* of the dissection, where for each mismatched region we move the two groups of $(d-1)$-simplices slightly apart leaving the relative boundary of the mismatched region joined. Clearly, the simplicial complex of a dissection may be not homeomorphic to a ball.

The deformed d-simplices intersect properly, and the mismatched regions become holes. The numbers of vertices and d-simplices do not change.

LEMMA 3.1. *All mismatched regions for a dissection of a convex 3-polytope P are convex polygons with all vertices among the vertices of P. Distinct mismatched regions have disjoint relative interiors.*

Proof. Let Q be a mismatched region and H the plane containing it. Since a mismatched region is a union of overlapping triangles, it is a polygon in H with a connected interior. If two triangles forming the mismatched region have interior points in common, they should be facets of tetrahedra in different sides of H. Otherwise the two tetrahedra would have interior points in common, contradicting the definition of dissection. Triangles which are facets of tetrahedra in one side of H cover Q. Triangles coming from the other side of H also cover Q.

Now take triangles coming from one side. As mentioned above, they have no interior points in common. Their vertices are among the vertices of the tetrahedra in the dissection, thus among the vertices of the polytope P. Hence the vertices of the triangles are in convex position, thus the triangles are forming a triangulation of a convex polygon in H whose vertices are among the vertices of P.

For the second claim, suppose there were distinct mismatched regions having an interior point in common. Then their intersection should be an interior segment for each. Let Q be one of the mismatched regions. It is triangulated in two different ways each coming from the tetrahedra in one side of the hyperplane. The triangles in either triangulation cannot intersect improperly with the interior segment. Thus the two triangulations of Q have an interior diagonal edge in common. This means the

triangles in Q consist of more than one connected component of the auxiliary graph, contradicting the definition of mismatched region. □

PROPOSITION 3.2.

(1) *The size of a mismatching dissection D of a convex 3-polytope with n vertices is at least $n-2$.*
(2) *The size of a dissection of a 3-polytope with n vertices is bounded from above by $\binom{n-2}{2}$.*

Proof. (1) Do an inflation of each mismatched region. This produces as many holes as mismatched regions, say, m of them. Each hole is bounded by two triangulations of a polygon. This is guaranteed by the previous lemma. Denote by k_i the number of vertices of the polygon associated with the ith mismatched region. In each of the holes introduce an auxiliary interior point. The point can be used to triangulate the interior of the holes by *filling in* the holes with the coning of the vertex with the triangles it sees. We now have a triangulated ball.

Denote by $|D|$ the size of the original dissection. The triangulated ball has then $|D|+\sum_{i=1}^{m} 2(k_i-2)$ tetrahedra in total. The number of interior edges of this triangulation is the number of interior edges in the dissection, denoted by $e_i(D)$, plus the new additions; for each hole of length k_i we added k_i interior edges. In a triangulation T of a 3-ball with n boundary vertices and n' interior vertices, the number of tetrahedra $|T|$ is related to the number of interior edges e_i of T by the formula $|T| = n+e_i-n'-3$. The proof is a simple application of Euler's formula for triangulated 2-spheres and 3-balls and we omit the easy details.

Thus, we have the following equation:

$$|D| + \sum_{i=1}^{m} 2(k_i-2) = n + e_i(D) + \sum_{i=1}^{m} k_i - m - 3.$$

This can be rewritten as $|D| = n + e_i(D) - \sum_{i=1}^{m} k_i + 3m - 3$. Taking into account that $e_i(D) \geq \sum_{i=1}^{m} 2(k_i-3)$ (because diagonals in a polygon are interior edges of the dissection), we get an inequality

$$|D| \geq n + \sum_{i=1}^{m} k_i - 3m - 3.$$

Finally, note that in a mismatching dissection we have $m \geq 1$ and $k_i \geq 4$. This gives the desired lower bound.

(2) Now we look at the proof of the upper bound on dissections. Given a 3-dissection, we add tetrahedra of volume zero to complete a triangulation with flat simplices that has the same number of vertices. One can also think we are *filling in* the holes created by an inflation with (deformed) tetrahedra.

Lemma 3.1 states that mismatched regions were of the shape of convex polygons. The 2-simplices forming a mismatched region were divided into two groups (those becoming apart by an inflation). The two groups formed different triangulations of a convex polygon, and they had no interior edges in common. In this situation, we can make a sequence of flips (see [17]) between the two triangulations with the property that any edge once disappeared does not appear again (see Figure 8). We add one abstract, volume zero tetrahedron for each flip and obtain an abstract triangulation of a 3-ball.

The triangulation with flat simplices we created is a triangulated 3-ball with n vertices. By adding a new point in a fourth dimension, and coning from the boundary

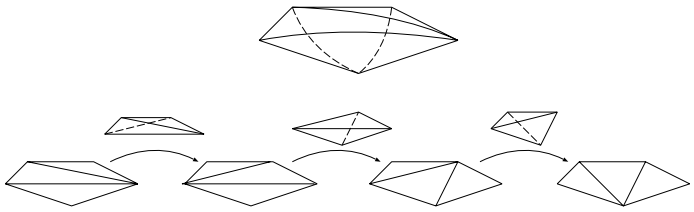


FIG. 8. *Filling in holes with tetrahedra according to flips.*

2-simplices to the point, we obtain a triangulated 3-sphere containing the original 3-ball in its boundary. From the upper bound theorem for spheres (for an introduction to this topic, see [30]) its size is bounded from above by the number of facets of a cyclic 4-polytope minus $2n-4$, the number of 2-simplices in the boundary of D. The four-dimensional cyclic polytope with $n+1$ vertices is well known to have $(n+1)(n-2)/2$ facets (see [11, p. 63]), which completes the proof after a trivial algebraic calculation. □

OPEN PROBLEM 3.3. *What is the correct upper bound theorem for dissections of d-dimensional polytopes with $d \geq 4$?*

In our proof of Proposition 3.2 we built a triangulated PL-ball from a three-dimensional dissection using the flip connectivity of triangulations of a convex n-gon. Unfortunately the same cannot be applied in higher dimensions, as the flip connectivity of triangulations of d-polytopes is known to be false for convex polytopes in general [22]. Even worse, however, the easy property we used from Lemma 3.1 that mismatched regions are convex polyhedra fails in dimension $d \geq 4$.

PROPOSITION 3.4. *The mismatched regions of a dissection of a convex 4-polytope can be nonconvex polyhedra.*

Proof. The key idea is as follows. Suppose we have a 3-dimensional convex polytope P and two triangulations T_1 and T_2 of it with the following properties: removing from P the tetrahedra that T_1 and T_2 have in common, the rest is a nonconvex polyhedron P' such that the triangulations T_1' and T_2' of it obtained from T_1 and T_2 do not have any interior 2-simplex in common (actually, something weaker would suffice: that their common interior triangles, if any, do not divide the interior of the polytope).

In these conditions, we can construct the dissection we want as a bipyramid over P, coning T_1 to one of the apices and T_2 to the other one. The bipyramid over the nonconvex polyhedron P' will be a mismatched region of the dissection.

For a concrete example, start with Schönhardt's polyhedron whose vertices are labeled $1, 2, 3$ in the lower face and $4, 5, 6$ in the top face. This is a nonconvex polyhedron made, for example, by twisting the three vertices on the top of a triangular prism. Add two antipodal points 7 and 8 close to the "top" triangular facets (those not breaking the quadrilaterals); see Figure 9. For example, take as coordinates for the points $1 = (10, 0, 0)$, $2 = (-6, 8, 0)$, $3 = (-6, -8, 0)$, $4 = (10, -0.1, 10)$, $5 = (-6.1, 8, 10)$, $6 = (-5.9, -8.1, 10)$, $7 = (0, 0, 10.1)$, $8 = (0, 0, -0.1)$.

Let P' be this nonconvex polyhedron and let $T_1' = \{1278, 1378, 2378, 1247, 2457, 2357, 3567, 1367, 1467\}$ and $T_2' = \{4578, 4678, 5678, 1248, 2458, 2358, 3568, 1368, 1468\}$. T_1' cones vertex 7 to the rest of the boundary of P', and T_2' cones vertex 8. Any common interior triangle of T_1' and T_2' would use the edge 78. But the link of 78 in T_1' contains only the points 1, 2, and 3, and the link in T_2' contains only 4, 5, and 6.

Let P be the convex hull of the eight points, and let T_1 and T_2 be obtained from

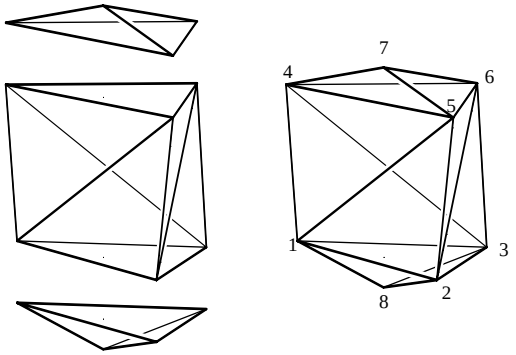


FIG. 9. *The mismatched region of a four-dimensional dissection.*

T_1' and T_2' by adding the three tetrahedra 1245, 2356, and 1346. □

4. Optimal dissections for specific polytopes. The regular cube has been widely studied for its smallest dissections [12, 14, 18]. This receives the name of *simplexity* of the cube. In contrast, because of the type of simplices inside a regular d-cube, a simple volume argument shows that the maximal size of a dissection is $d!$, the same as for triangulations. On the other hand, we know that the size of the maximal triangulation of a *combinatorial* cube can be larger than that: For example, the combinatorial 3-cube obtained as the prism over a trapezoid (vertices on a parabola for instance) has triangulations of size 7. Figure 10 shows a triangulation with seven simplices for those coordinatizations, where the edges AB and GH are not coplanar. The tetrahedron $ABGH$ splits the polytope into two nonconvex parts, each of which can be triangulated with three simplices. To see this, suppose that our polytope is a very small perturbation of a regular 3-cube. In the regular cube, $ABGH$ becomes a diagonal plane which divides the cube into two triangular prisms, $ABCDGH$ and $ABEFGH$. In the nonregular cube, the diagonals AH and BG, respectively, become nonconvex. Any pair of triangulations of the two prisms, each using the corresponding diagonal, together with tetrahedron $ABGH$ give a triangulation of the perturbed cube with seven tetrahedra. The boundary triangulation is shown in the flat diagram. It is worth noticing that for the regular cube the boundary triangulation we showed does not extend to a triangulation of the interior.

One can then ask, "What is the general growth for the size of a maximal dissection of a combinatorial cube?" To answer this question, at least partially, we use the above construction and we adapt an idea of Haiman, originally devised to produce small triangulations of regular cubes [12]. The idea is that from triangulations of a d_1-cube and a d_2-cube of sizes s_1 and s_2, respectively, we can get triangulations of the $(d_1 + d_2)$-cube by first subdividing it into $s_1 \times s_2$ copies of the product of two simplices of dimensions d_1 and d_2 and then triangulating each such piece. We recall that any triangulation of the Cartesian product of a d_1-simplex and a d_2-simplex has $\binom{d_1+d_2}{d_1}$ maximal simplices. Hence in total we have a triangulation of the $(d_1 + d_2)$-cube into $s_1 \times s_2 \times \binom{d_1+d_2}{d_1}$ maximal simplices. Recursively, if one starts with a triangulation of size s of the d-cube, one obtains triangulations for the rd-cube of size $(rd)!(\frac{s}{d!})^r$. In Haiman's context one wants s to be small, but here we want it to be big.

More precisely, denote by $f(d)$ the function $\max_{C:\ d\text{-cube}}(\max_{T \text{ of } C} |T|)$ and call

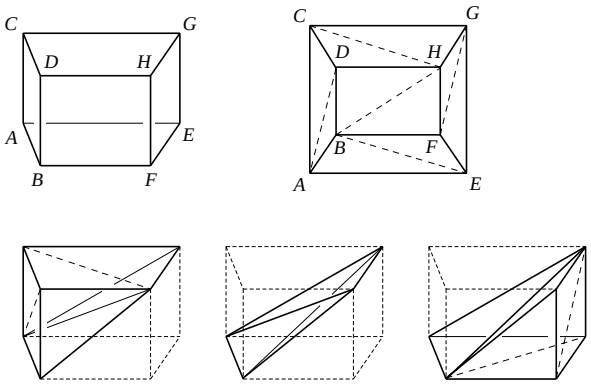


FIG. 10. *A triangulation of a combinatorial* 3*-cube into seven tetrahedra.*

$g(d) = (f(d)/d!)^{1/d}$. Haiman's argument shows that if $f(d_1) \geq {c_1}^{d_1} d_1!$ and $f(d_2) \geq {c_2}^{d_2} d_2!$ for certain constants c_1 and c_2, then $f(d_1 + d_2) \geq {c_1}^{d_1} {c_2}^{d_2} (d_1 + d_2)!$ (put differently, that $g(d_1 + d_2) \geq \left(g(d_1)^{d_1} g(d_2)^{d_2}\right)^{1/(d_1+d_2)}$). The value on the right-hand side is the weighted geometric mean of $g(d_1)$ and $g(d_2)$. In particular, if both $g(d_1)$ and $g(d_2)$ are ≥ 1 and one of them is > 1, then $g(d_1 + d_2)$ is > 1 as well.

We have constructed above a triangulation of size 7 for the Klee–Minty 3-cube, which proves $g(3) \geq \sqrt[3]{7/6} = 1.053$. With Haiman's idea we can now construct "large" triangulations of certain 4-cubes and 5-cubes, which prove, respectively, that $g(4) \geq \sqrt[4]{7/6} = 1.039$ and $g(5) \geq \sqrt[5]{7/6} = 1.031$ (take $d_1 = 3$ and d_2 equal to one and two, respectively). Finally, since any $d > 5$ can be expressed as a sum of 3's and 4's, we have $g(d) \geq \min\{g(3), g(4)\} \geq 1.039$ for any $d > 5$. Hence we have the following proposition.

PROPOSITION 4.1. *For the family of combinatorial* d*-cubes with* $d > 2$ *the function* $f(d) = \max_{C:\ d\text{-cube}}(\max_{T \text{ of } C} |T|)$ *admits the lower bound* $f(d) \geq c^d d!$ *where* $c \geq 1.031$.

Exactly as in Haiman's paper, the constant c can be improved (asymptotically) if one starts with larger triangulations for the smaller-dimensional cubes. Using computer calculations (see Remark 4.5), we obtained a maximal triangulation for the Klee–Minty 4-cube with 38 maximal simplices, which shows that $g(d) \geq \sqrt[4]{38/24} = 1.122$ for every d divisible by 4 (see [1] for a complete study of this family of cubes). We omit listing the triangulation here, but it is available from the authors by request.

OPEN PROBLEM 4.2. *Is the sequence* $g(d)$ *bounded? In other words, is there an upper bound of type* $c^d d!$ *for the function* $f(d)$*? Observe that the same question for* minimal *triangulations of the* regular d*-cube* (*whether there is a lower bound of type* $c^d d!$ *for some* $c > 0$) *is open as well. See* [26] *for the best lower bound known.*

We continue our discussion with the study of optimal triangulations for three-dimensional prisms and antiprisms. We will call an m*-prism* any 3-polytope with the combinatorial type of the product of a convex m-gon with a line segment. An m*-antiprism* will be any 3-polytope whose faces are two convex m-gons and $2m$ triangles, each m-gon being adjacent to half of the triangles. Vertices of the two m-gons are connected with a band of alternately up and down pointing triangles.

Each such polyhedron has a regular coordinatization in which all the faces are regular polygons and a realization space which is the set of all possible coordinatizations that yield the same combinatorial information [20]. Our first result is valid in

the whole realization space.

PROPOSITION 4.3. *For any three-dimensional m-prism, in any of its possible coordinatizations, the number of tetrahedra in a minimal triangulation is* $2m-5+\lceil \frac{m}{2} \rceil$.

For any three-dimensional m-antiprism, in any of its possible coordinatizations, the number of tetrahedra in a minimal triangulation is $3m-5$.

Proof. In what follows we use the word *cap* to refer to the m-gon facets appearing in a prism or antiprism. We begin our discussion proving that any triangulation of the prism or antiprism has at least the size we state, and then we will construct triangulations with exactly that size.

We first prove that every triangulation of the m-prism requires at least $2m-5+\lceil \frac{m}{2} \rceil$ tetrahedra. We call a tetrahedron of the m-prism *mixed* if it has two vertices on the top cap and two vertices on the bottom cap of the prism; otherwise we say that the tetrahedron is *top-supported* when it has three vertices on the top (respectively, *bottom-supported*). For example, Figure 11 shows a triangulation of the regular 12-prism in three slices. Parts (a) and (c) represent, respectively, the bottom and top caps. Part (b) is the intersection of the prism with the parallel plane at equal distance to both caps. In this intermediate slice, bottom or top supported tetrahedra appear as triangles, while mixed tetrahedra appear as quadrilaterals.

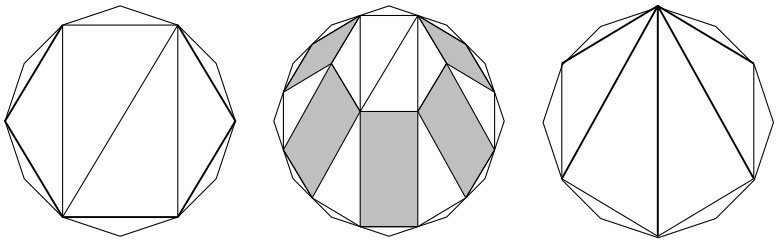


FIG. 11. *A minimal triangulation of the regular* 12*-prism.*

Because all triangulations of an m-gon have $m-2$ triangles there are always exactly $2m-4$ tetrahedra that are bottom- or top-supported. In the rest, we show there are at least $\lceil \frac{m}{2} \rceil - 1$ mixed tetrahedra. Each mixed tetrahedra *marks* an edge of the top, namely, the edge it uses from the top cap. Of course, several mixed tetrahedra could mark the same top edge. Group together top-supported tetrahedra that have the same bottom vertex. This grouping breaks the triangulated top m-gon into polygonal regions. Note that every edge between two of these regions must be marked. For example, in part (c) of Figure 11 the top cap is divided into six regions by five marked edges (the thick edges in the figure). Let r equal the number of regions under the equivalence relation we set. There are $r-1$ interior edges separating the r regions, and all of them are marked. Some boundary edges of the top cap may be marked too (none of them is marked in the example of Figure 11).

We can estimate the marked edges in another way: There are m edges on the boundary of the top, which appear partitioned among some of the regions (it could be the case some region does not contain any boundary edge of the m-gon). We claim that no more than *two* boundary edges per region will be unmarked ($*$). This follows because a boundary edge is not marked only when the top-supported tetrahedron that contains it has the point in the bottom cap that is directly under one of the vertices of the edge. In a region, at most two boundary edges can satisfy this. Hence we get

at least $m-2r$ marked edges on the boundary of the top and at least $(r-1)+(m-2r) = m-r-1$ marked edges in total. Thus the number of mixed tetrahedra is at least the maximum of $r-1$ and $m-r-1$. In conclusion, we get that, indeed, the number of mixed tetrahedra is bounded below by $\lceil \frac{m}{2} \rceil - 1$. Note that we use only the combinatorics and convexity of the prism in our arguments. We will show that minimal triangulations achieve this lower bound but then observe that if m is even, in a minimal triangulation we must have $r = m/2$ and no boundary edge can be marked, as is the case in Figure 11. If m is odd, then we must have $r \in \{(m-1)/2, (m+1)/2\}$ and at most one boundary edge can be marked.

The proof that any triangulation of an m-antiprism includes at least $3m-5$ tetrahedra is similar. There are $2m-4$ top-supported and bottom-supported tetrahedra in any triangulation and there are $r-1$ marked edges between the regions in the top. The only difference is that, instead of claim $(*)$, one has at most *one* unmarked boundary edge per region. Thus there are at least $m-r$ marked edges in the boundary of the top and in total at least $(r-1)+(m-r) = m-1$ marked edges in the top. Hence there exist at least $(2m-4)+(m-1) = 3m-5$ tetrahedra in any triangulation.

For an m-antiprism we can easily create a triangulation of size $3m-5$ by choosing any triangulation of the bottom m-gon and then coning a chosen vertex v of the top m-gon to the $m-2$ triangles in that triangulation and to the $2m-3$ triangular facets of the m-antiprism which do not contain v. This construction is exhibited in Figure 12. Parts (a) and (c) show the bottom and top caps triangulated (each with its five marked edges) and part (b) shows an intermediate slice with the five mixed tetrahedra appearing as quadrilaterals.

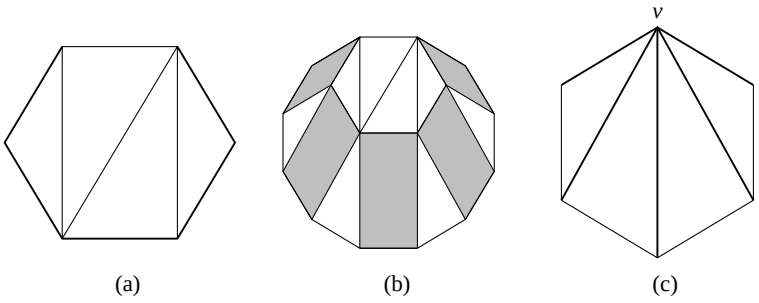


FIG. 12. *A minimal triangulation of the regular* 6*-antiprism.*

For an m-prism, let u_i and v_i, $i = 1, \ldots, m$ denote the top and bottom vertices, respectively, so that the vertices of each cap are labeled consecutively and $u_i v_i$ is always an edge of the prism.

If m is even we can chop off the vertices u_i for odd i and v_j for even j, so that the prism is decomposed into m tetrahedra and an $(\frac{m}{2})$-antiprism. The antiprism can be triangulated into $\frac{3m}{2} - 5$ tetrahedra, which gives a triangulation of the prism into $\frac{5m}{2} - 5$ tetrahedra, as desired. Actually, this is how the triangulation of Figure 11 can be obtained from that of Figure 12.

If m is odd we do the same, except that we chop off only the vertices $u_1, \ldots, u_{m-2}$ and $v_2, \ldots, v_{m-1}$ (no vertex is chopped in the edge $u_m v_m$). This produces $m-1$ tetrahedra and an $(\frac{m+1}{2})$-antiprism. We triangulate the antiprism into $\frac{3m+3}{2} - 5$ tetrahedra and this gives a triangulation of the m-prism into $\frac{5m+1}{2} - 5$ tetrahedra. □

We have seen that the coordinates are not important when calculating minimal triangulations of the three-dimensional prisms and antiprisms. On the other hand, the difference in size of the maximal triangulation can be quite dramatic. Below we prove that in certain coordinatizations it is roughly $\frac{m^2}{2}$ and show experimental data indicating that for the regular prism it is close to $\frac{m^2}{4}$.

PROPOSITION 4.4. *Let A_m be a prism of order m with all its side edges parallel.*

(1) *The size of a maximal triangulation of A_m is bounded as*

$$\left\lceil \frac{m^2+6m-16}{4} \right\rceil \leq \max_{T \text{ of } A_m} |T| \leq \frac{m^2+m-6}{2}.$$

(2) *The upper bound is achieved if the two caps (m-gon facets) are parallel and there is a direction in which the whole prism projects onto one of its side quadrangular facets. (For a concrete example, let one of the m-gon facets have vertices on a parabola and let A_m be the product of it with a segment.)*

Proof. Let the vertices of the prism be labeled $u_1, \ldots, u_m$ and $v_1, \ldots, v_m$ so that the u_i's and the v_j's form the two caps, vertices in each cap are labeled consecutively, and $u_i v_i$ is always a side edge.

For the upper bound in part (1), we have to prove that a triangulation of A_m has at most $\frac{m^2+m-6}{2} - 2m + 3 = \frac{m(m-3)}{2}$ interior diagonals. The possible diagonals are the edges $u_i v_j$, where $i - j$ is not in $\{-1, 0, 1\}$ modulo m. This gives exactly twice the number we want. However, for any i and j the diagonals $u_i v_j$ and $u_j v_i$ intersect, so only one of them can appear in each triangulation.

We now prove that the upper bound is achieved if A_m is in the conditions of part (2). In fact, the condition on A_m that we will need is that for any $1 \leq i < j \leq k < l \leq m$, the point v_j sees the triangle $v_i u_k u_l$ from the same side as v_k and v_l (i.e., "from above" if we call top cap the one containing the v_i's). With this we can construct a triangulation with $\frac{m^2+m-6}{2} = \binom{m-1}{2} + 2m - 4$ tetrahedra as follows.

First cone the vertex v_1 to any triangulation of the bottom cap (this gives $m-2$ tetrahedra). The $m-2$ upper boundary facets of this cone are visible from v_2, and we cone them to it (again $m-2$ tetrahedra). The new $m-2$ upper facets are visible from v_3 and we cone them to it ($m-2$ tetrahedra more). Now one of the upper facets of the triangulation is $v_1 v_2 v_3$, part of the upper cap, but the other $m-3$ are visible from v_4, so we cone them and introduce $m-4$ tetrahedra. Continuing the process, we will introduce $m-4$, $m-5, \ldots, 2, 1$ tetrahedra when coning the vertices $v_5, v_6, \ldots, v_{m-1}, v_m$, which gives a total of $\binom{m-1}{2} + 2m - 4$ tetrahedra, as desired.

The triangulation we have constructed is the *placing triangulation* [17] associated with any ordering of the vertices finishing with $v_1, \ldots, v_m$. A different description of the same triangulation is that it cones the bottom cap to v_1, the top cap to u_m, and its mixed tetrahedra are all the possible $v_i v_{i+1} u_j u_{j+1}$ for $1 \leq i < j \leq m-1$. This gives $\binom{m-1}{2}$ mixed tetrahedra and $\binom{m-1}{2} + 2m - 4$ tetrahedra in total.

We finally prove the lower bound stated in part (1). Without loss of generality, we can assume that our prism has its two caps parallel (if not, do a projective transformation keeping the side edges parallel). Then, A_m can be divided into two prisms in the conditions of part (2) of sizes k and l with $k + l = m + 2$: take any two side edges of A_m which possess parallel supporting planes and cut A_m along the plane containing both edges. By part (2), we can triangulate the two subprisms with $\binom{k+1}{2} - 3$ and $\binom{l+1}{2} - 3$ tetrahedra, respectively, taking care that the two triangulations use the same diagonal in the dividing plane. This gives a triangulation of A_m with

$\binom{k+1}{2} + \binom{l+1}{2} - 6 = \frac{k^2+l^2+m-10}{2}$ tetrahedra. This expression achieves its minimum when k and l are as similar as possible, i.e., $k = \lfloor \frac{m}{2} \rfloor + 1$ and $l = \lceil \frac{m}{2} \rceil + 1$. Plugging these values in the expression gives a triangulation of size $\lceil \frac{m^2+6m-16}{4} \rceil$. □

Based on an integer programming approach we can compute maximal triangulations of specific polytopes (see Remark 4.5). Our computations with regular prisms up to $m = 12$ show that the size of their maximal triangulations achieve the lower bound stated in part (1) of Proposition 4.4 (see Table 2). In other words, they show that the procedure of dividing them into two prisms of sizes $\lfloor \frac{m}{2} \rfloor + 1$ and $\lceil \frac{m}{2} \rceil + 1$ in the conditions of part (2) of Proposition 4.4 and triangulating the subprisms independently yields maximal triangulations.

We have also computed maximal sizes of triangulations for the regular m-antiprisms up to $m = 12$, which turn out to follow the formula $\lfloor \frac{m^2+8m-16}{4} \rfloor$. A construction of a triangulation of this size for every m can be made as follows: Let the vertices of the regular m-antiprism be labeled $u_1, \ldots, u_m$ and $v_1, \ldots, v_m$ so they are forming the vertices of the two caps consecutively in this order and $v_i u_i$ and $u_i v_{i+1}$ are side edges. We let $v_{m+1} = v_1$. The triangulation is made by placing the vertices in any ordering finishing with $v_1, v_2, v_m, v_3, v_{m-1}, \ldots, v_{\lceil \frac{m}{2} \rceil + 1}$. The tetrahedra used are the bottom-supported tetrahedra with apex v_1, top-supported tetrahedra with apex $u_{\lceil \frac{m}{2} \rceil}$, and the mixed tetrahedra $v_i v_{i+1} u_j u_{j+1}$ for $1 \le i \le j \le \lfloor \frac{m}{2} \rfloor$ and $u_i u_{i+1} v_j v_{j+1}$ for $\lfloor \frac{m}{2} \rfloor + 1 \le i < j \le m$.

We conjecture that these formulas for regular base prisms and antiprisms actually give the sizes of their maximal triangulations for every m, but we do not have a proof.

TABLE 2
Sizes of maximal triangulations of prisms and antiprisms.

m	3	4	5	6	7	8	9	10	11	12
Prism (regular base)	3	6	10	14	19	24	30	36	43	50
Antiprism (regular base)	4	8	12	17	22	28	34	41	48	56

Remark 4.5. How can one find minimal and maximal triangulations in specific instances? The approach we followed for computing Tables 1 and 2 and some of the results in Proposition 2.3 is the one proposed in [8], based on the solution of an integer programming problem. We think of the triangulations of a polytope as the vertices of the following high-dimensional polytope: Let A be a d-dimensional polytope with n vertices. Let N be the number of d-simplices in A. We define P_A as the convex hull in $\boldsymbol{R}^N$ of the set of incidence vectors of all triangulations of A. For a triangulation T the *incidence vector* v_T has coordinates $(v_T)_\sigma = 1$ if $\sigma \in T$ and $(v_T)_\sigma = 0$ if $\sigma \notin T$. The polytope P_A is the *universal polytope* defined in general by Billera, Filliman, and Sturmfels [3], although it appeared in the case of polygons in [7]. In [8], it was shown that the vertices of P_A are precisely the integral points inside a polyhedron that has a simple description in terms of the oriented matroid of A (see [8] for information on oriented matroids). The concrete integer programming problems were solved using *C-plex Linear Solver*^{TM}. The program to generate the linear constraints is a small C^{++} program written by Samuel Peterson and the first author. Source code, brief instructions, and data files are available via ftp at http://www.math.ucdavis.edu/~deloera. An alternative implementation by Tajima is also available [27, 28]. He used his program to corroborate some of these results.

It should be mentioned that a simple variation of the ideas in [8] provides enough

equations for an integer program whose feasible vertices are precisely the 0/1-vectors of dissections. The incidence vectors of dissections of $conv(A)$, for a point set A, are just the 0/1 solutions to the system of equations $\langle x, v_T \rangle = 1$, where v_T's are the incidence vectors for every regular triangulation T of the Gale transform A^* (regular triangulations in the Gale transform are the same as chambers in A). Generating all these equations is as hard as enumerating all the chambers of A. Nevertheless, it is enough to use those equations coming from placing triangulations (see [23, section 32]), which gives a total of about n^{d+1} equations if A has n points and dimension d.

Acknowledgments. We are grateful to Alexander Below and Jürgen Richter-Gebert for their help and ideas in the proofs of Propositions 3.2 and 3.4. Alexander Below made Figure 9 using the package Cinderella. The authors thank Akira Tajima and Jörg Rambau for corroborating many of the computational results. We thank Samuel Peterson for his help with our calculations. Finally, we thank Hiroshi Imai, Bernd Sturmfels, and Akira Tajima for their support of this project.

REFERENCES

[1] N. Amenta and G.M. Ziegler, *Deformed products and maximal shadows of polytopes*, in Advances in Discrete and Computational Geometry, B. Chazelle, J.E. Goodman, and R. Pollack, eds., Contemp. Math. 223, AMS, Providence, RI, 1999, pp. 57–90.

[2] A. Barvinok and J. Pommersheim, *An algorithmic theory of lattice points in polyhedra*, in New Perspectives in Algebraic Combinatorics, Math. Sci. Res. Inst. Publ. 38, Cambridge University Press, Cambridge, UK, 1999.

[3] L. Billera, P. Filliman, and B. Sturmfels, *Constructions and complexity of secondary polytopes*, Adv. Math., 83 (1990), pp. 155–179.

[4] A. Below, U. Brehm, J.A. De Loera, and J. Richter-Gebert, *Minimal simplicial dissections and triangulations of convex 3-polytopes*, Discrete Comput. Geom., 24 (2000), pp. 35–48.

[5] W. Bruns, J. Gubeladse, and N.V. Trung, *Normal polytopes, triangulations and Koszul algebras* J. Reine Angew. Math., 485 (1997), pp. 123–160.

[6] H.S.M. Coxeter, *Regular Polytopes*, Dover, New York, 1973.

[7] G.B. Dantzig, A.J. Hoffman, and T.C. Hu, *Triangulations (tilings) and certain block triangular matrices*, Math. Programming, 31 (1985), pp. 1–14.

[8] J.A. De Loera, S. Hoşten, F. Santos, and B. Sturmfels, *The polytope of all triangulations of a point configuration*, Doc. Math., 1 (1996), pp. 103–119.

[9] H. Edelsbrunner, F.P. Preparata, and D.B. West, *Tetrahedrizing point sets in three dimensions*, J. Symbolic Comput., 10 (1990), pp. 335–347.

[10] R.T. Firla and G.M. Ziegler, *Hilbert bases, unimodular triangulations, and binary covers of rational polyhedral cones*, Discrete Comput. Geom., 21 (1999), pp. 205–216.

[11] B. Grünbaum, *Convex Polytopes*, Interscience, London, 1967.

[12] M. Haiman, *A simple and relatively efficient triangulation of the n-cube*, Discrete Comput. Geom., 6 (1991), pp. 287–289.

[13] B. Huber, J. Rambau, and F. Santos, *The Cayley trick, lifting subdivisions and the Bohne-Dress theorem on zonotopal tilings*, J. Eur. Math. Soc. (JEMS), 2 (2000), pp. 179–198.

[14] R.B. Hughes and M.R. Anderson, *Simplexity of the cube*, Discrete Math., 158 (1996), pp. 99–150.

[15] J.-M. Kantor, *Triangulations of integral polytopes and Ehrhart polynomials*, Beiträge Algebra Geom., 39 (1998), pp. 205–218.

[16] J. Lagarias and G.M. Ziegler, *Unimodular Triangulations*, manuscript, 1999.

[17] C.W. Lee, *Subdivisions and triangulations of polytopes*, in Handbook of Discrete and Computational Geometry, J.E. Goodman and J. O'Rourke, eds., CRC Press, Boca Raton, FL, 1997, pp. 271–290.

[18] P.S. Mara, *Triangulations for the cube*, J. Combin. Theory Ser. A, 20 (1976), pp. 170–177.

[19] J. Rambau, *TOPCOM: A Program for Computing All Triangulations of a Point Set*, ZIB-Berlin, 1999; also available online from http://www.zib.de/rambau/TOPCOM.html.

[20] J. Richter-Gebert, *Realization Spaces of Polytopes*, Lecture Notes in Math. 1643, Springer-Verlag, Berlin, 1996.

[21] G.L. Rothschild and E.G. Straus, *On triangulations of the convex hull of n points*, Combinatorica, 5 (1985), pp. 167–179.
[22] F. Santos, *A point configuration whose space of triangulations is disconnected*, J. Amer. Math. Soc., 13 (2000), pp. 611–637.
[23] F. Santos, *Triangulations of oriented matroids*, Mem. Amer. Math. Soc., to appear; also available online from http://www.matesco.unican.es/~santos/Articulos/index.html.
[24] A. Sebö, *Hilbert bases, Carathéodory's theorem and combinatorial optimization*, in Integer Programming and Combinatorial Optimization, R. Kannan and W. Pulleyblank, eds., Math. Programming Society, University of Waterloo Press, Waterloo, Ontario, Canada, 1990, pp. 431–456.
[25] A. Sebö, *An Introduction to Empty Lattice Simplices*, manuscript; also available online from http://cosmos.imag.fr/DMD/OPTICOMB/Membres/sebo/sebo.html.
[26] W.D. Smith, *A lower bound for the simplexity of the n-cube via hyperbolic volumes*, European J. Combin., 21 (2000), pp. 131–137.
[27] A. Tajima, *Optimality and integer programming formulations of triangulations in general dimension*, in Proceedings of 9th Annual International Symposium on Algorithms and Computation (ISAAC '98), K.-Y. Chwa and O.H. Ibarra, eds., Lecture Notes in Comput. Sci. 1533, Springer-Verlag, Berlin, pp. 377–386.
[28] A. Tajima, *Optimizing Geometric Triangulations by Using Integer Programming*, Ph.D. thesis, University of Tokyo, Tokyo, Japan 2000; also available online from http://www-imai.is.s.u-tokyo.ac.jp/~akira/papers/dissertation.pdf.
[29] F. Takeuchi and H. Imai, *Enumerating triangulations for products of two simplices and for arbitrary configurations of points*, in Proceedings of 3rd Annual International Conference on Computing and Combinatorics (COCOON '97), T. Jiang and D. T. Lee, eds., Lecture Notes in Comput. Sci. 1276, Springer-Verlag, Berlin, pp. 470–481.
[30] G.M. Ziegler, *Lectures on Polytopes*, Springer-Verlag, New York, 1995.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 162–169

ON THE DISTRIBUTION OF DIFFIE–HELLMAN TRIPLES WITH SPARSE EXPONENTS*

JOHN B. FRIEDLANDER† AND IGOR E. SHPARLINSKI‡

Abstract. Let g be a primitive root modulo a $(n+1)$-bit prime p. In this paper we prove the uniformity of distribution of the Diffie–Hellman triples (g^x, g^y, g^{xy}) as the exponents x and y run through the set of n-bit integers with precisely k nonzero bits in their bit representation provided that $k \geq 0.35n$. Such "sparse" exponents are of interest because for these the computation of g^x, g^y, g^{xy} is faster than for arbitrary x and y. In the latter case, that is, for arbitrary exponents, similar (albeit stronger) uniformity of distribution results have recently been obtained by R. Canetti, M. Larsen, D. Lieman, S. Konyagin [*Israel J. Math*, 120 (2000), pp. 23–46], and the authors.

Key words. Diffie–Hellman cryptosystem, sparse exponents, exponential sums

AMS subject classifications. 11T23, 11T71, 94A60

PII. S0895480199361740

1. Introduction and auxiliary results.

Let p be a prime and let $\mathbb{F}_p$ be a finite field of p elements which we identify with the set $\{0, \ldots, p-1\}$.

We fix a primitive root $g \in \mathbb{F}_p$ and consider *Diffie–Hellman triples* (g^x, g^y, g^{xy}); see [13, 16]. It has been shown in [2] and then improved in [1] that such triples are uniformly distributed when $x, y = 0, \ldots, p-2$. Such results, although they do not have any immediate security implications, are nevertheless very desirable. For example, the opposite statement would make possible a simple statistics-based attack on the Diffie–Hellman cryptosystems. On the other hand, studying the distribution of these triples is a very natural and attractive number theoretic question.

Here we consider the situation when the exponents x and y have a prescribed number k of nonzero bits in their bit representation. If k is small, then for such sparse exponents the computation of g^x, g^y, g^{xy} takes less time than for arbitrary x and y, and thus this choice has been considered in the literature. It is useful to recall that standard repeated squaring computation of u^e, with an integer exponent $e \geq 2$ over any ring, takes about $\log e + \nu(e)$ arithmetic operations where $\nu(e)$ is the number of nonzero bits in the bit representation of e and $\log z$ denotes, throughout, the binary logarithm of z; see section 1.3 of [3], section 4.3 of [4], or section 2.1 of [5]. It follows that the use of the aforementioned sparse x and y with $\nu(x) \sim \nu(y) \sim 0.35 \log p$ provides a 10% speed-up on average and a speed-up of more than 30% in the worst case.

Our results are based on some new bounds of exponential sums which in turn rely on bounds from [1].

We define the integer n by the inequalities

$$2^n \leq p - 1 \leq 2^{n+1} - 1$$

*Received by the editors September 24, 1999; accepted for publication (in revised form) December 12, 2000; published electronically February 23, 2001.
http://www.siam.org/journals/sidma/14-2/36174.html

†Department of Mathematics, University of Toronto, Toronto, Ontario M5S 3G3, Canada (frdlndr@math.toronto.edu). This author's work was supported in part by NSERC grant A5123 and by NEC Research Inc.

‡Department of Computing, Macquarie University, NSW 2109, Australia (igor@comp.mq.edu.au). This author's work was supported in part by ARC grant A69700294.

and denote by $\mathcal{W}_k$ the set of n-bit integers which have precisely k nonzero bits in their bit representation.

Finally we put $\mathbf{e}(z) = \exp(2\pi iz/p)$ and define the exponential sums

$$S_k(a,b,c) = \sum_{x\in\mathcal{W}_k}\sum_{y\in\mathcal{W}_k} \mathbf{e}\left(ag^x + bg^y + cg^{xy}\right).$$

We estimate these sums and then, using some standard arguments, derive the uniformity of distribution result for the triples (g^x, g^y, g^{xy}), $x, y \in \mathcal{W}_k$, provided that $k \ge 0.35n$.

Throughout the paper the implied constants in the symbols "O," "$\ll$," and "$\gg$" are absolute (we recall that $A \ll B$ and $B \gg A$ are equivalent to $A = O(B)$). We recall the well-known fact (see Problem 11.c in Chapter 2 of [17]) that the number $\tau(m)$ of integer divisors of $m \ge 1$ satisfies

$$\tau(m) \le m^{o(1)}. \tag{1}$$

The following statement has been proved in [1]; see the proof of Theorem 8 of that paper.

LEMMA 1.1. *Let $\lambda \in \mathbb{F}_p$ be of multiplicative order T. For any $a, b \in \mathbb{F}_p^*$, the bound*

$$\sum_{u\in\mathbb{Z}_T}\left|\sum_{v\in\mathbb{Z}_T}\mathbf{e}\left(a\lambda^v + b\lambda^{uv}\right)\right|^4 \ll pT^{11/3}$$

holds.

The following statement is well known and can be found in [9, 10, 14].

LEMMA 1.2. *Let $\lambda \in \mathbb{F}_p$ be of multiplicative order T. For any $a \in \mathbb{F}_p^*$ and any integer $H \le T$ the bound*

$$\left|\sum_{u=1}^{H}\mathbf{e}\left(a\lambda^u\right)\right| \ll p^{1/2}\log p$$

holds.

Several other related estimates are given in [8].

2. Exponential sums with sparse integers. We prove two bounds for the sums $S_k(a,b,c)$. The first one applies when $c \in \mathbb{F}_p^*$, the other, obtained by a different method, applies in the case $c = 0$.

For technical reasons we shall assume throughout the condition $k \le n/2$. Of course, since we are interested in choosing k as small as possible, this is the case of practical interest. Moreover, the case when $k \ge n/2$ is completely symmetric to the case $k \le n/2$ and can be dealt with along the same lines.

Let $H(\gamma)$ denote the *binary entropy function*

$$H(\gamma) = \begin{cases} -\gamma\log\gamma - (1-\gamma)\log(1-\gamma) & \text{if } 0 < \gamma < 1,\\ 0 & \text{otherwise.}\end{cases}$$

We define

$$G(\gamma) = \begin{cases} H(\gamma) & \text{if } \gamma < 1/2,\\ 1 & \text{if } \gamma \ge 1/2,\end{cases}$$

and

$$F(\alpha,\gamma) = 12H(\alpha) - 14/3 - 7(1-\gamma)G\left(\alpha/(1-\gamma)\right) - \gamma/3.$$

Finally, we put

$$E(\alpha) = \inf_{0\le\gamma<1} F(\alpha,\gamma).$$

Let $\beta = \alpha/(1-\gamma)$; thus the condition $0 \le \gamma < 1$ is equivalent to $\alpha \le \beta$ and

$$7(1-\gamma)G\left(\alpha/(1-\gamma)\right) + \gamma/3 = \alpha\left(7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3\right) + 1/3.$$

If $\beta \ge 1/2$, then

$$7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3 = 20\beta^{-1}/3$$

and therefore is decreasing there. For $\alpha \le \beta \le 1/2$ we compute the derivative

$$\begin{aligned}\frac{d}{d\beta}\left(7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3\right) &= \frac{d}{d\beta}\left(7\beta^{-1}H\left(\beta\right) - \beta^{-1}/3\right)\\ &= \frac{1}{3\beta^2}\left(21\log(1-\beta) + 1\right).\end{aligned}$$

Thus if $\alpha > 1 - 2^{-1/21}$ this derivative is negative and we see that

$$\begin{aligned}\sup_{\beta\ge\alpha}\left(7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3\right) &= \sup_{\beta\in[\alpha,1/2]}\left(7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3\right)\\ &= 7\alpha^{-1}H\left(\alpha\right) - \alpha^{-1}/3.\end{aligned}$$

If $\alpha \le 1 - 2^{-1/21}$, then

$$\sup_{\beta\ge\alpha}\left(7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3\right) = \sup_{\beta\in[\alpha,1/2]}\left(7\beta^{-1}G\left(\beta\right) - \beta^{-1}/3\right) = \vartheta_0,$$

where

$$\vartheta_0 = 7\left(1-2^{-1/21}\right)^{-1}H\left(1-2^{-1/21}\right) - \left(1-2^{-1/21}\right)^{-1}/3.$$

Therefore

$$E(\alpha) = \begin{cases} 5H(\alpha) - 14/3 & \text{if } 1/2 \ge \alpha > 1-2^{-1/21},\\ 12H(\alpha) - 5 - \vartheta_0\alpha & \text{if } 1-2^{-1/21} \ge \alpha > 0.\end{cases}$$

Now one easily verifies that $E(\alpha)$ is continuous and strictly increasing for $0 < \alpha \le 1/2$. Therefore we can define $\alpha_0 = 0.349\ldots$ as the unique root of the equation $E(\alpha) = 0$, $0 \le \alpha \le 1/2$.

THEOREM 2.1. *For any fixed $\alpha > \alpha_0$ there exists $\delta > 0$ such that for $n/2 \ge k \ge \alpha n$ the bound*

$$\max_{a,b\in\mathbb{F}_p;\, c\in\mathbb{F}_p^*} |S_k(a,b,c)| \ll |\mathcal{W}_k|^{2-\delta}$$

holds.

Proof. For a divisor $d|p-1$ we denote by $\mathcal{W}_k(d)$ the subset of $y \in \mathcal{W}_k$ with $\gcd(y, p-1) = d$. Then

$$|S_k(a,b,c)| \le \sum_{d|p-1} |\sigma_d|,$$

where

$$\sigma_d = \sum_{x \in \mathcal{W}_k} \sum_{y \in \mathcal{W}_k(d)} \mathbf{e}\left(ag^x + bg^y + cg^{xy}\right).$$

Using the Cauchy inequality, we derive

$$\begin{aligned}
|\sigma_d|^2 &\le |\mathcal{W}_k| \sum_{x \in \mathcal{W}_k} \left| \sum_{y \in \mathcal{W}_k(d)} \mathbf{e}\left(bg^y + cg^{xy}\right) \right|^2 \\
&\le |\mathcal{W}_k| \sum_{x=0}^{p-2} \left| \sum_{y \in \mathcal{W}_k(d)} \mathbf{e}\left(bg^y + cg^{xy}\right) \right|^2 \\
&= |\mathcal{W}_k| \sum_{y,z \in \mathcal{W}_k(d)} \mathbf{e}\left(bg^y - bg^z\right) \sum_{x=0}^{p-2} \mathbf{e}\left(c\left(g^{xy} - g^{xz}\right)\right).
\end{aligned}$$

By the Hölder inequality we have

$$\begin{aligned}
|\sigma_d|^8 &\le |\mathcal{W}_k|^4 |\mathcal{W}_k(d)|^6 \sum_{y,z \in \mathcal{W}_k(d)} \left| \sum_{x=0}^{p-2} \mathbf{e}\left(c\left(g^{xy} - g^{xz}\right)\right) \right|^4 \\
&\le |\mathcal{W}_k|^4 |\mathcal{W}_k(d)|^6 \sum_{y \in \mathcal{W}_k(d)} \sum_{u=0}^{(p-1)/d-1} \left| \sum_{x=0}^{p-2} \mathbf{e}\left(c\left(g^{xy} - g^{xud}\right)\right) \right|^4.
\end{aligned}$$

Because each element $y \in \mathcal{W}_k(d)$ can be represented in the form $y = dv$ with $\gcd(v, (p-1)/d) = 1$ and $g_d = g^d$ is of multiplicative order $(p-1)/d$, we see that the double sum over u and x does not depend on y. Therefore,

$$\begin{aligned}
|\sigma_d|^8 &\le |\mathcal{W}_k|^4 |\mathcal{W}_k(d)|^7 \sum_{u=0}^{(p-1)/d-1} \left| \sum_{x=0}^{p-2} \mathbf{e}\left(c\left(g_d^x - g_d^{xu}\right)\right) \right|^4 \\
&= |\mathcal{W}_k|^4 |\mathcal{W}_k(d)|^7 d^4 \sum_{u=0}^{(p-1)/d-1} \left| \sum_{v=0}^{(p-1)/d-1} \mathbf{e}\left(c\left(g_d^v - g_d^{vu}\right)\right) \right|^4.
\end{aligned}$$

By Lemma 1.1 we obtain

$$|\sigma_d|^8 \ll |\mathcal{W}_k|^4 |\mathcal{W}_k(d)|^7 p^{14/3} d^{1/3}. \tag{2}$$

To estimate $|\mathcal{W}_k(d)|$ we remark that if d is a divisor of $p-1$ in the range $2^l \le d \le 2^{l+1}-1$ and if $y \in \mathcal{W}_k(d)$, then those bits of y in the l rightmost positions are uniquely determined by the bits at the other (at most) $n-l$ positions. Therefore

$$|\mathcal{W}_k(d)| \le \sum_{j=0}^{k} \binom{n-l}{j}.$$

We recall the estimate

$$\binom{q}{s} \le \sum_{i=0}^{s} \binom{q}{i} \le 2^{qH(s/q)}, \tag{3}$$

which holds for any $s \le q/2$. Indeed, for $s < q/2$ it is Corollary 9 of section 10.11 of [11] while for $s = q/2$ we have $H(s/q) = H(1/2) = 1$ and the bound is obvious. Therefore,

$$|\mathcal{W}_k| \le 2^{nH(k/n)}$$

and also

$$|\mathcal{W}_k(d)| \le 2^{(n-l)G(k/(n-l))}.$$

Substituting the above bound in (2), we obtain

$$\begin{aligned} |\sigma_d|^8 &\le |\mathcal{W}_k|^4 2^{12nH(k/n)-nF(k/n,l/n)} = |\mathcal{W}_k|^{16} 2^{-nF(k/n,l/n)} \\ &\le |\mathcal{W}_k|^{16} 2^{-nE(k/n)+o(n)} \le |\mathcal{W}_k|^{16} 2^{-nE(\alpha)+o(n)}. \end{aligned}$$

Hence

$$|S_k(a,b,c)| \le |\mathcal{W}_k|^2 2^{-nE(\alpha)/8+o(n)} \tau(p-1).$$

Applying the bound (1), we derive the result. □

To estimate sums $S_k(a,b,c)$ with $c = 0$ we remark that

$$S_k(a,b,0) = T_k(a) T_k(b), \tag{4}$$

where

$$T_k(a) = \sum_{x \in \mathcal{W}_k} \mathbf{e}(ag^x).$$

Let us define the function

$$R(\beta,\rho) = \sup_{0 \le \lambda \le 1-\rho} \left\{ \rho H\left(\frac{\beta-\lambda}{\rho}\right) + 2(1-\rho) H\left(\frac{\lambda}{1-\rho}\right) \right\}.$$

We also put

$$Q(\beta) = \sup_{0<\rho<1} \min\left\{2H(\beta) - 1/2 - R(\beta,\rho), H(\beta) - \rho\right\}.$$

Using the same routine arguments as for the function $E(\alpha)$, one verifies that $Q(\beta)$ is a monotonically increasing function in the interval $[0, 1/2]$ and we define $\beta_0 = 0.202\ldots$ as the unique root of the equation $Q(\beta) = 0$, $0 \le \beta \le 1/2$.

THEOREM 2.2. *For any fixed $\beta > \beta_0$ there exists $\delta > 0$ such that for $n/2 \ge k \ge \beta n$ we have the bound*

$$\max_{a \in \mathbb{F}_p^*} |T_k(a)| \ll |\mathcal{W}_k|^{1-\delta}.$$

Proof. Select some $r \le n$ and denote by $\mathcal{U}_l$ the set of r-bit integers with $k-l$ nonzero bits in their bit representation and by $\mathcal{V}_l$ the set of $(n-r)$-bit integers with l nonzero bits in their bit representation. Obviously,

$$\sum_{l=0}^{n-r} |\,\mathcal{U}_l||\,\mathcal{V}_l| = |\,\mathcal{W}_k|.$$

We also have

$$T_k(a) = \sum_{l=0}^{n-r} \sum_{u \in \mathcal{U}_l} \sum_{v \in \mathcal{V}_l} \mathbf{e}\left(ag^{u+2^r v}\right).$$

Using the Cauchy inequality twice, we derive

$$\begin{aligned}
|T_k(a)|^2 &\le (n-r+1) \sum_{l=0}^{n-r} |\,\mathcal{U}_l| \sum_{u \in \mathcal{U}_l} \left| \sum_{v \in \mathcal{V}_l} \mathbf{e}\left(ag^{u+2^r v}\right) \right|^2 \\
&\le n \sum_{l=0}^{n-r} |\,\mathcal{U}_l| \sum_{u=0}^{2^r} \left| \sum_{v \in \mathcal{V}_l} \mathbf{e}\left(ag^{u+2^r v}\right) \right|^2 \\
&= n \sum_{l=0}^{n-r} |\,\mathcal{U}_l| \sum_{v_1, v_2 \in \mathcal{V}_l} \sum_{u=0}^{2^r} \mathbf{e}\left(ag^u \left(g^{2^r v_1} - g^{2^r v_2}\right)\right).
\end{aligned}$$

If $v_1 \ne v_2$, then, obviously, Lemma 1.2 applies to the inner sum. Otherwise we use the trivial bound, getting

$$\begin{aligned}
|T_k(a)|^2 &\ll n \sum_{l=0}^{n-r} |\,\mathcal{U}_l| \left(|\,\mathcal{V}_l|^2 2^{n/2} n + |\,\mathcal{V}_l| 2^r \right) \\
&= n^2 2^{n/2} \sum_{l=0}^{n-r} |\,\mathcal{U}_l||\,\mathcal{V}_l|^2 + n|\,\mathcal{W}_k| 2^r.
\end{aligned}$$

From (3) and the definition of $R(\beta, \gamma)$ we see that for $l = 0, \dots, r$

$$|\,\mathcal{U}_l||\,\mathcal{V}_l|^2 \le 2^{n/2 + R(k/n, r/n)}.$$

Therefore we derive

$$|T_k(a)|^2 \le n^3 2^{n/2 + R(k/n, r/n)} + n|\,\mathcal{W}_k| 2^r.$$

Because this holds for every r, then

$$|T_k(a)|^2 \le |\,\mathcal{W}_k|^2 2^{-nQ(k/n) + o(n)} \le |\,\mathcal{W}_k|^2 2^{-nQ(\beta) + o(n)},$$

provided that $k \ge \beta n$ for some $\beta > \beta_0$. □

Because $\beta_0 < \alpha_0$ from Theorems 2.1 and 2.2 we deduce the following theorem.

THEOREM 2.3. *For any fixed $\alpha > \alpha_0$ there exists $\delta > 0$ such that for $n/2 \ge k \ge \alpha n$ we have the bound*

$$\max_{\gcd(a,b,c,p)=1} |S_k(a,b,c)| \ll |\,\mathcal{W}_k|^{2-\delta}.$$

3. Distribution of the Diffie–Hellman triples with sparse exponents. Given a sequence of s-dimensional vectors $s_1, \ldots, s_M \in \mathbb{F}_p^s$, we define its *discrepancy* D as

$$D = \sup_{\mathcal{B} \subseteq [0,1]^s} \left| \frac{N(\mathcal{B})}{M} - |\mathcal{B}| \right|,$$

where $N(\mathcal{B})$ is the number of fractions s_ν/p, $\nu = 1, \ldots, M$, which hit the box $\mathcal{B} = [\alpha_1, \beta_1] \times \cdots \times [\alpha_s, \beta_s] \subseteq [0,1]^s$ of size

$$|\mathcal{B}| = \prod_{j=1}^{s} (\beta_j - \alpha_j).$$

In this paper we use this notion only with $s = 3$.

We denote by Δ_k the discrepancy of the triples (g^x, g^y, g^{xy}), $x, y \in \mathcal{W}_k$. Our bound of exponential sums leads to a similar upper bound for Δ_k. More precisely, Corollary 3.11 of [15] implies that

$$\Delta_k \ll |\mathcal{W}_k|^{-2} \max_{\gcd(a,b,c,p)=1} |S_k(a,b,c)| \log^3 p.$$

Combining this bound with Theorem 2.3, we derive the following result.

THEOREM 3.1. *For any fixed $\alpha > \alpha_0$ there exists $\delta > 0$ such that for $n/2 \ge k \ge \alpha n$ the bound*

$$\Delta_k \ll |\mathcal{W}_k|^{-\delta}$$

holds.

This bound implies that a positive proportion, say, 0.3δ, of the most significant bits of (g^x, g^y, g^{xy}) are independently and uniformly distributed when x and y run through the set $\mathcal{W}_k$. For arbitrary $x, y = 0, \ldots, p-2$ a similar result has been obtained in [2] and then improved in [1].

As has been done in [1, 2] in the case of arbitrary x and y, one can also derive the same result for the least significant bits.

4. Remarks. We note that for small values of d much more precise results are known about the sets $\mathcal{W}_k(d)$ (see [6, 12]), but unfortunately they cannot be used for our applications.

One can obtain a less accurate but simpler bound for the sums $T_k(a)$ by using the inequality $|\mathcal{V}_{k,l}| \le 2^{n-r}$ which implies

$$\sum_{l=0}^{n-r} |\mathcal{U}_{k,l}| |\mathcal{V}_{k,l}|^2 \le |\mathcal{W}_k| 2^{n-r}.$$

Thus taking $r = 3n/4$ one immediately obtains Theorem 2.2 with $\beta_0 = 0.214\ldots$ defined by the equation $H(\beta_0) = 3/4$. This is quite enough for our purposes, but the sums $T_k(a)$ are of independent interest so we present the more complicated but more precise estimate. The method used in studying these sums can be used to bound many other character sums over the elements of $\mathcal{W}_k$. For example one can study sums

$$T_k(f, h; \chi) = \sum_{x \in \mathcal{W}_k} \chi(f(x)) \mathbf{e}(h(x))$$

with rational functions $f(X), h(X) \in \mathbb{F}_p(X)$, where χ is a multiplicative character of $\mathbb{F}_p^*$. Of course, for this sum the Weil bound is to be used. In particular, one can show that under the conditions of Theorem 2.2 there are asymptotically $0.5|\,\mathcal{W}_k|$ quadratic nonresidues $x \in \mathcal{W}_k$.

Studying arithmetic properties of integers with given properties of their digits is a classical topic in number theory; see [6, 7, 12] and references therein. Nevertheless, results of this kind do not seem to be known.

It is a very interesting problem to obtain similar results for smaller values of k, say, for $k = o(n)$. We believe that it is quite unrealistic to hope to get such results if the primitive root g is fixed. On the other hand, we believe it could be possible to prove that such a result (for quite small values of k) holds for almost all primitive roots g.

Finally we remark that the same results hold for the set of integers with at most k nonzero bits.

REFERENCES

[1] R. Canetti, J. B. Friedlander, S. Konyagin, M. Larsen, D. Lieman, and I. E. Shparlinski, *On the statistical properties of Diffie–Hellman distributions*, Israel J. Math., 120 (2000), pp. 23–46.

[2] R. Canetti, J. B. Friedlander, and I. E. Shparlinski, *On certain exponential sums and the distribution of Diffie–Hellman triples*, J. London Math. Soc. (2), 59 (1999), pp. 799–812.

[3] H. Cohen, *A Course in Computational Algebraic Number Theory*, Springer-Verlag, Berlin, 1997.

[4] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, Cambridge, UK, 1999.

[5] D. M. Gordon, *A survey of fast exponentiation methods*, J. Algorithms, 27 (1998), pp. 129–146.

[6] S. Konyagin, *Arithmetic properties of integers with missing digits: Distribution in residue classes*, Period. Math. Hungar., to appear.

[7] S. Konyagin, Ch. Mauduit, and A. Sárközy, *On the number of prime factors of integers characterized by digit properties*, Period. Math. Hungar., 40 (2000), pp. 37–52.

[8] S. Konyagin and I. E. Shparlinski, *Character Sums with Exponential Functions and Their Applications*, Cambridge University Press, Cambridge, UK, 1999.

[9] N. M. Korobov, *On the distribution of digits in periodic fractions*, Mat. Sb., 89 (1972), pp. 654–670 (in Russian).

[10] N. M. Korobov, *Exponential Sums and their Applications*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[11] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, 1977.

[12] Ch. Mauduit and A. Sárközy, *On the arithmetic structure of the integers whose sum of digits is fixed*, Acta Arith., 81 (1997), pp. 145–173.

[13] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, FL, 1996.

[14] H. Niederreiter, *Quasi-Monte Carlo methods and pseudorandom numbers*, Bull. Amer. Math. Soc., 84 (1978), pp. 957–1041.

[15] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 63, SIAM, Philadelphia, 1992.

[16] D. R. Stinson, *Cryptography: Theory and Practice*, CRC Press, Boca Raton, FL, 1995.

[17] I. M. Vinogradov, *Elements of Number Theory*, Dover, New York, 1954.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 170–180

IMPROVING ON THE 1.5-APPROXIMATION OF A SMALLEST 2-EDGE CONNECTED SPANNING SUBGRAPH*

J. CHERIYAN†, A. SEBŐ‡, AND Z. SZIGETI§

Abstract. We give a $\frac{17}{12}$-approximation algorithm for the following NP-hard problem:

> Given a simple undirected graph, find a 2-edge connected spanning subgraph that has the minimum number of edges.

The best previous approximation guarantee was $\frac{3}{2}$. If the well-known $\frac{4}{3}$ conjecture for the metric traveling salesman problem holds, then the optimal value (minimum number of edges) is at most $\frac{4}{3}$ times the optimal value of a linear programming relaxation. Thus our main result gets halfway to this target.

Key words. approximation algorithms, edge connectivity, ear decomposition, graphs, joins, metric traveling salesman problem and 4/3 conjecture, NP-complete problems

AMS subject classifications. 05C40, 05C85, 68W25, 90C27, 90C59

PII. S0895480199362071

1. Introduction.

Given a simple undirected graph, consider the problem of finding a 2-edge connected spanning subgraph that has the minimum number of edges. The problem is NP-hard, via a reduction from the Hamiltonian cycle problem. A number of recent papers have focused on approximation algorithms for this and other related problems [3]. An *α-approximation algorithm* for a combinatorial optimization problem runs in polynomial time and delivers a solution whose value is always within the factor α of the optimum value. The quantity α is called the *approximation guarantee* of the algorithm. We use the abbreviation 2-ECSS to mean 2-edge connected spanning subgraph.

Here is an easy 2-approximation algorithm for the problem:

> Take an ear decomposition of the given graph (see section 2 for definitions), and discard all 1-ears (ears that consist of one edge). Then the resulting graph is 2-edge connected and has at most $2n-3$ edges, while the optimal subgraph has $\geq n$ edges, where n is the number of nodes.

Khuller and Vishkin [11] were the first to improve on the approximation guarantee of 2. They gave a simple and elegant algorithm based on depth-first search that achieves an approximation guarantee of 1.5. We improve Khuller and Vishkin's $\frac{18}{12}$-approximation guarantee to $\frac{17}{12}$. If the well-known $\frac{4}{3}$ conjecture for the metric traveling salesman

*Received by the editors September 25, 1999; accepted for publication (in revised form) November 2, 2000; published electronically February 23, 2001. A preliminary version of this paper appeared as *An improved approximation algorithm for minimum size 2-edge connected spanning subgraphs* in Proceedings of the Sixth International Integer Programming and Combinatorial Optimization Conference, Houston, TX, 1998, Lecture Notes in Comput. Sci. 1412, R.E. Bixby, E.A. Boyd, and R. Z. Rios-Mercado, eds., Springer-Verlag, Berlin, 1998, pp. 126–136.
http://www.siam.org/journals/sidma/14-2/36207.html

†Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1 (jcheriyan@math.uwaterloo.ca).

‡Departement de Mathematiques Discretes, CNRS, Laboratoire LEIBNIZ (CNRS, INPG, UJF)-IMAG, 46 Avenue Felix Viallet, 38031 Grenoble Cedex 1, France (Andras.Sebo@imag.fr).

§Equipe Combinatoire, Université Pierre et Marie Curie, 4 place Jussieu, Couloir 45-46 3e, 75252 Paris Cedex 5, France (Zoltan.Szigeti@ecp6.jussieu.fr).

problem (TSP) holds, then the optimal value (minimum number of edges) is at most $\frac{4}{3}$ times the optimal value of a linear programming relaxation; see Theorem 5.1. Thus our main result gets halfway to this target.

Let $G = (V, E)$ be the given simple undirected graph, and let n and m denote $|V|$ and $|E|$. Assume that G is 2-edge connected.

Our method is based on a matching-theory result of András Frank, namely, that there is a good characterization for the minimum number of even-length ears over all possible ear decompositions of a graph, and moreover, an ear decomposition achieving this minimum can be computed efficiently [5]. Recall that the 2-approximation heuristic starts with an arbitrary ear decomposition of G. Instead, if we start with an ear decomposition that maximizes the number of 1-ears, and if we discard all the 1-ears, then we will obtain the optimal solution. In fact, we start with an ear decomposition that maximizes the number of odd-length ears. Now, discarding all the 1-ears gives an approximation guarantee of 1.5 (see Proposition 3.4 below). To do better, we repeatedly apply "ear splicing" steps to the starting ear decomposition to obtain a final ear decomposition such that the number of odd-length ears is the same, and moreover, the internal nodes of distinct 3-ears are nonadjacent. We employ two lower bounds to show that discarding all the 1-ears from the final ear decomposition gives an approximation guarantee of $\frac{17}{12}$. The first lower bound is the "component lower bound" due to Garg, Santosh, and Singla [8, Lemma 4.1]; see Proposition 2.4 below. The second lower bound comes from the minimum number of even-length ears in an ear decomposition of G; see Proposition 3.3 below.

A preliminary version of this paper has appeared in [2]. Since this paper was submitted for journal publication, an improved approximation guarantee for the minimum-size 2-ECSS problem has been reported by Vempala and Vetta [15].

After developing some preliminaries in sections 2 and 3, we present our heuristic in section 4. Section 5.1 shows the relation of the well-known $\frac{4}{3}$ conjecture for the metric TSP to the problem of finding a $\frac{4}{3}$-approximation algorithm for a minimum-size 2-ECSS; see Theorem 5.1. Section 5.2 has two examples showing that our analysis of the heuristic is tight. Section 5.2 also compares the two lower bounds with the optimal value.

A useful assumption. For our heuristic to work, it is essential that the given graph be 2-node connected. Hence, in section 4 of the paper where our heuristic is presented, we will assume that the given graph G is 2-node connected. Otherwise, if G is not 2-node connected, we compute the blocks (i.e., the maximal 2-node connected subgraphs) of G, and apply the algorithm separately to each block. We compute a 2-ECSS for each block and output the union of the edge sets as the edge set of a 2-ECSS of G. The resulting graph has no cut edges since the subgraph found for each block has no cut edge. Moreover, if an approximation guarantee of α holds for each block, then it holds for G, because the optimal value for G equals the sum of the optimal values for the blocks.

2. Preliminaries. Except in section 5.1, all graphs are simple, that is, there are no loops or multiedges. A closed path means a cycle, and an open path means that all the nodes are distinct.

An *ear decomposition* of the graph G is a partition of the edge set into open or closed paths, $P_0 + P_1 + \cdots + P_k$, such that P_0 is the trivial path with one node, and each P_i $(1 \le i \le k)$ is a path that has both end nodes in $V_{i-1} = V(P_0) \cup V(P_1) \cup \cdots \cup V(P_{i-1})$ but has no internal nodes in V_{i-1}. A (closed or open) *ear* means one of the (closed or open) paths $P_1, \ldots, P_k$ in the ear decomposition; note that P_0 is not regarded as

an ear. By induction on the number of ears k, it is easily seen that for the number of edges in G,

$$|E| = |V| + k - 1. \tag{2.1}$$

For a nonnegative integer ℓ, an *ℓ-ear* means an ear that has ℓ edges. An ℓ-ear is called *even* if ℓ is an even number, otherwise, the ℓ-ear is called *odd*. An *open* ear decomposition $P_0 + P_1 + \cdots + P_k$ is one such that all the ears $P_2, \ldots, P_k$ are open. (The ear P_1 is always closed.)

PROPOSITION 2.1 (see Whitney [16]).

(i) *A graph is 2-edge connected iff it has an ear decomposition.*

(ii) *A graph is 2-node connected iff it has an open ear decomposition.*

An *odd* ear decomposition is one such that every ear has an odd number of edges. The graph G is called *factor-critical* if for every node $v \in V(G)$ there is a perfect matching in $G - v$. The next result gives another characterization of factor-critical graphs.

THEOREM 2.2 (see Lovász [12] and Lovász and Plummer [13, Theorem 5.5.1]). *A graph is factor-critical iff it has an odd ear decomposition.*

It follows that a factor-critical graph is necessarily 2-edge connected. An *open odd* ear decomposition $P_0 + P_1 + \cdots + P_k$ is an odd ear decomposition such that all the ears $P_2, \ldots, P_k$ are open.

THEOREM 2.3 (see Lovász and Plummer [13, Theorem 5.5.2]). *A 2-node connected factor-critical graph has an open odd ear decomposition. Given such a graph $G = (V, E)$, an open odd ear decomposition can be constructed in time $O(|V| \cdot |E|)$.*

Let $\varepsilon(G)$ denote the minimum number of edges in a 2-ECSS of G. For a graph H, let $c(H)$ denote the number of (connected) components of H. Garg, Santosh, and Singla [8, Lemma 4.1] use the following lower bound on $\varepsilon(G)$.

PROPOSITION 2.4. *Let $G = (V, E)$ be a 2-edge connected graph, and let S be a nonempty set of nodes such that the deletion of S results in a graph with $c = c(G - S)$ components. Then $\varepsilon(G) \geq |V| + c - |S|$.*

Proof. Focus on an arbitrary component D of $G - S$ and note that it contributes $\geq |V(D)| + 1$ edges to an optimal 2-ECSS, because every node in D contributes ≥ 2 edges, and at least two of these edges have exactly one end node in D. Summing over all components of $G - S$ gives the result. □

For the graph $G = (V, E)$, let $L_c^\star(G)$ denote $\max\{|V| + c(G - S) - |S| \ : \ \emptyset \neq S \subseteq V\}$; by Proposition 2.4, $\varepsilon(G) \geq L_c^\star(G)$.

For a set of nodes $S \subseteq V$ of a graph $G = (V, E)$, $\delta(S)$ denotes the set of edges that have one end node in S and one end node in $V - S$. For the singleton node set $\{v\}$, we use the notation $\delta(v)$. For a vector $x : E \to \mathbf{R}$ and an edge set $F \subseteq E$, $x(F)$ denotes $\sum_{e \in F} x_e$.

3. Frank's theorem and a new lower bound for ε . For a 2-edge connected graph $G = (V, E)$, let $\varphi(G)$ (or φ) denote the minimum number of even ears over all possible ear decompositions. For example, $\varphi(G) = 0$ if G is a factor-critical graph (e.g., G is an odd clique $K_{2\ell+1}$ or an odd cycle $C_{2\ell+1}$), $\varphi(G) = 1$ if G is an even clique $K_{2\ell}$ or an even cycle $C_{2\ell}$, and $\varphi(G) = \ell - 1$ if G is the complete bipartite graph $K_{2,\ell}$ ($\ell \geq 2$). Let $L_\varphi(G)$ denote $|V| + \varphi(G) - 1$.

A *join* of a graph G is an edge set $J \subseteq E(G)$ such that for (the edge set of) every cycle $Q \subseteq E(G)$ we have $|J \cap Q| \leq |Q|/2$. For example, any matching is a join. Let $\mu(G)$ denote the maximum size of a join of the graph G.

The proof of the next result appears in [5]; see Theorem 4.5 and section 2 of [5].

THEOREM 3.1 (see Frank [5]). *Let $G = (V, E)$ be a 2-edge connected graph. An ear decomposition $P_0 + P_1 + \cdots + P_k$ of G having $\varphi(G)$ even ears can be computed in time $O(|V| \cdot |E|)$. Moreover, $L_\varphi(G) = 2\mu(G)$.*

PROPOSITION 3.2. *For every 2-node connected graph G, there exists an* open *ear decomposition $P_0 + P_1 + \cdots + P_k$ that has $\varphi(G)$ even ears. Such an ear decomposition can be computed in time $O(|V| \cdot |E|)$.*

Proof. Apply Theorem 3.1 to construct an ear decomposition having $\varphi(G)$ even ears (the ears may be open or closed). Subdivide one edge in each even ear by adding one new node and one new edge. The resulting ear decomposition is odd. Hence, the resulting graph G' is factor-critical, and also, G' is 2-node connected since G is 2-node connected. Apply Theorem 2.3 to construct an open odd ear decomposition of G'. Finally, in the resulting ear decomposition, "undo" the $\varphi(G)$ edge subdivisions to obtain the desired ear decomposition $P_0 + P_1 + \cdots + P_k$ of G.

The running time for constructing $P_0 + P_1 + \cdots + P_k$ is $O(|V| \cdot |E|)$. Note that there are constructive proofs for both Theorems 3.1 and 2.3, and each construction can be implemented in time $O(|V| \cdot |E|)$. □

Frank's theorem gives the following lower bound on the minimum number of edges in a 2-ECSS.

PROPOSITION 3.3. *Let $G = (V, E)$ be a 2-edge connected graph. Then $\varepsilon(G) \geq L_\varphi(G) = 2\mu(G)$.*

Proof. Consider an arbitrary 2-ECSS $G' = (V, E')$ of G. Note that G' contains all nodes of G but there may be several edges in $E - E'$. If G' has an ear decomposition with fewer than $\varphi(G)$ even ears, then we can obtain an ear decomposition of G with fewer than $\varphi(G)$ even ears as follows: we start with the ear decomposition of G', and for each edge $vw \in E - E'$, we add the 1-ear v, w. This contradiction to the definition of $\varphi(G)$ shows that every ear decomposition of G' has $\geq \varphi(G)$ even ears. Let $P_0 + P_1 + \cdots + P_k$ be an ear decomposition of the 2-ECSS G', where $k \geq \varphi(G)$. By (2.1), $|E'| = |V| + k - 1 \geq |V| + \varphi(G) - 1$. The result follows. □

The next result is not useful for our main result, but we include it for completeness.

PROPOSITION 3.4. *Let $G = (V, E)$ be a 2-edge connected graph. Let $P_0 + P_1 + \cdots + P_k$ be an ear decomposition of G that has $\varphi(G)$ even ears, and let $G' = (V, E')$ be obtained by discarding all the 1-ears from $P_0 + P_1 + \cdots + P_k$. Then $|E'|/\varepsilon(G) \leq 1.5$.*

Proof. Let t be the number of internal nodes in the odd ears of $P_0 + P_1 + \cdots + P_k$. (Note that the node in P_0 is not counted by t.) Then, the number of edges contributed to E' by the odd ears is $\leq 3t/2$, and the number of edges contributed to E' by the even ears is $\leq \varphi + |V| - t - 1 = L_\varphi(G) - t$. By applying Proposition 3.3 (and the fact that $\varepsilon(G) \geq |V|$) we get

$$\begin{aligned} |E'|/\varepsilon(G) &\leq (t/2 + L_\varphi(G))/\max(|V|, L_\varphi(G)) \\ &\leq (t/2|V|) + L_\varphi(G)/L_\varphi(G) \\ &\leq 1.5. \end{aligned}$$ □

4. Approximating ε via Frank's theorem. For a graph H and an ear decomposition $P_0 + P_1 + \cdots + P_k$ of H, we call an ear P_i of length ≥ 2 *pendant* if none of the internal nodes of P_i is an end node of another ear P_j of length ≥ 2. In other words, if we discard all the 1-ears from the ear decomposition, then one of the remaining ears is called pendant if all its internal nodes have degree 2 in the resulting graph.

Let $G = (V, E)$ be the given graph, and let $\varphi = \varphi(G)$. Recall the assumption from section 1 that G is 2-node connected. By an *evenmin ear decomposition* of G, we mean an ear decomposition that has $\varphi(G)$ even ears. Our method starts with

an open evenmin ear decomposition $P_0 + P_1 + \cdots + P_k$ of G (see Proposition 3.2), i.e., for $2 \leq i \leq k$, every ear P_i has distinct end nodes, and the number of even ears is minimum possible. The method performs a sequence of "ear splicings" to obtain another (evenmin) ear decomposition $Q_0 + Q_1 + \cdots + Q_k$ (the ears Q_i may be either open or closed) such that the following holds.

Property (α).

(0) The number of even ears is the same in $P_0 + P_1 + \cdots + P_k$ and in $Q_0 + Q_1 + \cdots + Q_k$,
(1) every 3-ear Q_i is a pendant ear,
(2) for every pair of 3-ears Q_i and Q_j, there is no edge between an internal node of Q_i and an internal node of Q_j, and
(3) every 3-ear Q_i is open, where $Q_i \neq Q_1$.

See Figure 1 for an illustration of several cases in an "ear splicing" step.

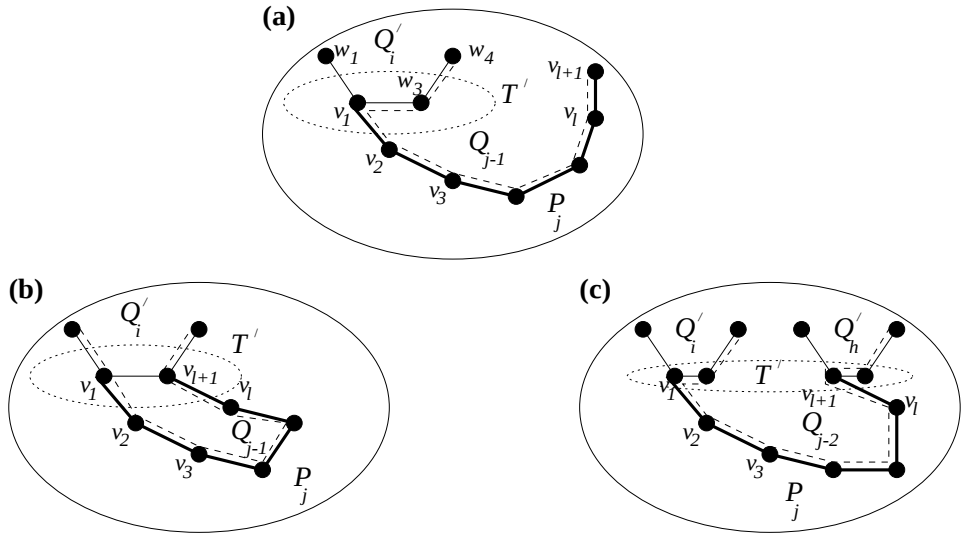


FIG. 1. *Illustration of the proof of Proposition* 4.1. (a), (b) *The first and second cases. Ears P_j and Q'_i are indicated by solid lines, and ear Q_{j-1} is indicated by dashed lines.* (c) *The third case. Ears P_j, Q'_i, Q'_h are indicated by solid lines, and ear Q_{j-2} is indicated by dashed lines.*

PROPOSITION 4.1. *Let $G = (V, E)$ be a 2-node connected graph. Let $P_0 + P_1 + \cdots + P_k$ be an open evenmin ear decomposition of G. There is a linear-time algorithm that, given $P_0 + P_1 + \cdots + P_k$, finds an ear decomposition $Q_0 + Q_1 + \cdots + Q_k$ satisfying property (α).*

Proof. The following procedure adds one ear P_j, $(j = 0, \ldots, k)$ at a time using ear splicing if necessary.

Suppose that $Q'_0 + \cdots + Q'_{j-1}$ have property (α). We will find in linear time in the length of P_j an ear-decomposition $Q_0 + \cdots + Q_j$ of $Q'_0 + \cdots + Q'_{j-1} + P_j$ that satisfies (α). Let T' denote here the set of internal nodes of the ears of length 3. If P_j has no endpoint in T', or has one endpoint in T' and the length of P_j is 1, we simply add it, that is, $Q_i := Q'_i$ $(i = 0, \ldots, j-1)$, $Q_j = P_j$.

If P_j has one endpoint in T' and has length bigger than 1, or has two endpoints in T' and these are in different 3-ears, then we add the appropriate subpaths of length 2 of these 3-ear(s) to P_j (Figure 1(a) and (c)). We get from the 3-ear(s) and P_j by this

ear-splicing an ear Q_j and 1-ear(s). Clearly, Q_j has the same parity as P_j. Property (α) is preserved, even if the two endpoints of Q_j are the same, since Q_j has length ≥ 4. Therefore, leaving the other ears as they are, the ear decomposition we have has property (α).

Finally if P_j has two endpoints in T' and both in the same ear (this is the only remaining case), then we add the first and last edge of the 3-ear to P_j to get Q_j, and the middle edge becomes a 1-ear (Figure 1(b)). The two new ears (Q_j and the 1-ear) should replace the 3-ear and P_j in the decomposition $Q'_0 + \cdots + Q'_{j-1} + P_j$, and property ($\alpha$) is again maintained. □

Remark. In the induction step, which applies for $j \geq 2$ (but not for $j = 1$), it is essential that the ear P_j is open, though Q'_i (and Q'_h) may be either open or closed. Note that Q_1 is not a 3-ear provided $|V| \neq 3$. Our main result (Theorem 4.3) does not use part (3) of property (α).

Our approximation algorithm for a minimum-size 2-ECSS computes the ear decomposition $Q_0+Q_1+\cdots+Q_k$ satisfying property (α), starting from an open evenmin ear decomposition $P_0 + P_1 + \cdots + P_k$. Then, the algorithm discards all the edges in 1-ears. Let the resulting graph be $G' = (V, E')$. G' is 2-edge connected by Proposition 2.1.

Let T denote the set of internal nodes of the 3-ears of $Q_0 + Q_1 + \cdots + Q_k$, and let $t = |T|$. (Note that the node in Q_0 is not counted by t.) Property (α) implies that in the subgraph of G induced by T, $G[T]$, every (connected) component has exactly two nodes. Consider the approximation guarantee for G', i.e., the quantity $|E'|/\varepsilon(G)$.

LEMMA 4.2. $\varepsilon(G) \geq L_c^\star(G) \geq 3t/2$.

Proof. Apply Proposition 2.4 with $S = V - T$ (so $|S| = n - t$) and $c = c(G - S) = t/2$ to get $\varepsilon(G) \geq n - (n - t) + (t/2)$. □

THEOREM 4.3. *Given a 2-edge connected graph $G = (V, E)$, the above algorithm finds a 2-ECSS $G' = (V, E')$ such that $|E'|/\varepsilon(G) \leq \frac{17}{12}$. The algorithm runs in time $O(|V| \cdot |E|)$.*

Proof. By the previous lemma and Proposition 3.3,

$$\varepsilon(G) \geq \max(3t/2, L_\varphi(G)). \tag{4.1}$$

We claim that

$$|E'| \leq \frac{t}{4} + \frac{5L_\varphi(G)}{4}. \tag{4.2}$$

To see this, let us denote by k, k_e, k_3, k_o the number of ears, the number of even ears, the number of 3-ears, and the number of odd ears of length > 3. Note that $k = k_e + k_3 + k_o, k_e = \varphi(G), k_3 = t/2$, and $k_o \leq (n - t - 1)/4$. Thus, by (2.1), $|E'| = n + k - 1 = n + k_e + k_3 + k_o - 1 \leq n + \varphi(G) + t/2 + (n - t - 1)/4 - 1 \leq t/4 + 5(n + \varphi(G) - 1)/4 = t/4 + 5L_\varphi(G)/4$.

The approximation guarantee follows since, by (4.1) and (4.2),

$$\frac{|E'|}{\varepsilon(G)} \leq \frac{t/4 + 5L_\varphi(G)/4}{\max(3t/2, L_\varphi(G))} \leq \frac{t}{4}\,\frac{2}{3t} + \frac{5L_\varphi(G)}{4L_\varphi(G)} = \frac{17}{12}. \qquad \square$$

The next result follows from the proof of Theorem 4.3.

COROLLARY 4.4. *For a 2-edge connected graph $G = (V, E)$,*

$$\varepsilon(G) \leq \frac{5}{4}\,L_\varphi + \frac{1}{6}L_c^\star \leq \frac{17}{12}\,\max(L_c^\star, L_\varphi).$$

5. Conclusions.

5.1. Lower bounds for ε and the relation to the TSP $frac43$ conjecture. This subsection has a comparison of several lower bounds for $\varepsilon(G)$; throughout, $G = (V, E)$ denotes an arbitrary 2-edge connected graph. The best of these lower bounds is given by a linear programming relaxation based on cut constraints. Moreover, we show that $\varepsilon(G)$ is at most $\frac{17}{12}$ times this lower bound. Theorem 5.1 below implies that if the well-known TSP $\frac{4}{3}$ conjecture is true, then we have $\frac{4}{3}$ rather than $\frac{17}{12}$ in the previous statement.

Recall that $L_\varphi(G) = |V| + \varphi(G) - 1 = 2\mu(G)$ is a lower bound on $\varepsilon(G)$, where $\mu(G)$ is the maximum size of a join of G; see Proposition 3.3.

Garg, Santosh, and Singla [8, Theorem 4.2] introduced another lower bound on $\varepsilon(G)$ that we denote by L_c. Let

$$L_c(G) = \max\left\{\sum_{i=1}^{\ell} c(G - S_i) \ : \ S_1, S_2, \ldots, S_\ell \text{ is a partition of } V, \text{ where } \ell \text{ is any integer} \geq 1\right\}.$$

(We remark that in the lower bound in [8, Theorem 4.2] $S_1, S_2, \ldots, S_\ell$ is a subpartition rather than a partition, but it can be seen that this lower bound equals L_c.) Clearly, $L_c \geq |V|$, by the partition of V into singleton sets. Notice that the lower bound in Proposition 2.4, $L_c^\star(G) = \max\{c(G-S)+|V-S| \ : \ \emptyset \neq S \subseteq V\}$, is $\leq L_c$; to see this, apply the definition of L_c with $S_1 = S$ and $S_2, \ldots, S_\ell$ being singleton sets of $V - S$.

Let $L_z(G)$ denote the optimal value of the following linear programming relaxation of the minimum-size 2-ECSS problem. There is one nonnegative variable x_e for each edge e in G, and the other constraints state that every (nontrivial) cut has x-weight at least two. Let $\mathbf{1}$ be a vector of "1"s with $|E|$ entries.

$$\begin{array}{lllll} L_z(G) = & \text{minimize} & \mathbf{1} \cdot x & & \\ & \text{subject to} & x(\delta(S)) \geq 2 & & \forall S \subset V,\ \emptyset \neq S \neq V, \\ & & x \geq 0, & & \\ & & x \in \mathbf{R}. & & \end{array}$$

Clearly, $L_z(G)$ is a lower bound on $\varepsilon(G)$ since the incidence vector of a minimum-size 2-ECSS satisfies all the constraints. We may have arbitrary coefficients $c : E \to \mathbf{R}$ in the objective function rather than unit coefficients, and then we will use $L_z(G, c)$ to denote the optimal value. Note that the optimal value of the linear program (LP) is computable in polynomial time, e.g., via the ellipsoid method.

Now consider the metric TSP. Let $G' = K_n$ be a complete graph and let $c' : E(G') \to \mathbf{R}$ assign metric costs to the edges (so for every triple of nodes i, j, k we have $c'(ij) \leq c'(ik) + c'(kj)$). Let $tsp(G', c')$ denote the minimum cost of a Hamiltonian cycle (or TSP tour) of G', c'. Clearly, the above linear program, but with objective vector c' instead of $\mathbf{1}$, is a relaxation of the TSP, so $L_z(G', c') \leq tsp(G', c')$.

The $\frac{4}{3}$ conjecture for the TSP is the following: if c' is a metric, then $tsp(G', c') \leq \frac{4}{3} L_z(G', c')$.

Remark. It is known that $tsp(G', c') \leq 1.5\, L_z(G', c')$; see [4],[9],[17]. The conjecture actually refers to the optimal value of the linear programming relaxation that has the additional constraints $x(\delta(v)) = 2$ for each node v; however, if the edge costs are metric, then the addition of the new constraints does not change the optimal value; see [14],[9].

THEOREM 5.1. *Let $G = (V, E)$ be a 2-edge connected graph. Then*

$$L_\varphi = 2\mu \le L_c \le L_z \le \varepsilon \le \frac{17}{12} L_c \le \frac{17}{12} L_z.$$

Moreover, if the $\frac{4}{3}$ conjecture for the metric TSP holds, then

$$\varepsilon \le \frac{4}{3} L_z.$$

Proof. To prove the first statement, we will derive the first two inequalities.

- $(2\mu \le L_c)$ Let J be a join of G with $|J| = \mu$. By [6, Theorem 8′] there exists a partition $V_1, \ldots, V_r$ $(r \ge 1)$ of V such that $2|J| \le \sum_{i=1}^{r} c(G - V_i)$. Therefore,

$$2\mu(G) \le \max\left\{ \sum_{i=1}^{\ell} c(G - S_i) \ : \ S_1, S_2, \ldots, S_\ell \text{ is a partition of } V \right\} = L_c.$$

Note that the maximization is over ℓ as well as the sets $S_1, S_2, \ldots, S_\ell$ in the partition.

- $(L_c \le L_z)$ Let $S_1, S_2, \ldots, S_\ell$ denote the optimal partition in the definition of L_c, so $L_c = \sum_{i=1}^{\ell} c(G - S_i)$. We sum up the following constraints (inequalities) from the linear program defining L_z: $x(\delta(V(D))) \ge 2$ for each component D of $G - S_i$, for each $i = 1, \ldots, \ell$. Let the resulting inequality be $\sum_{e \in E} a_e x_e \ge b_0$. Clearly, $b_0 = 2\sum_{i=1}^{\ell} c(G - S_i)$. Moreover, note that every coefficient a_{vw} $(vw \in E)$ is ≤ 2. To see this, we consider two cases: v, w are in different sets, say, $v \in S_i$, $w \in S_j$ $(i \ne j)$, or v, w are in the same set S_i. Consider the first case in detail; the inequality for the component of $G - S_i$ containing w contributes x_{vw}, and similarly for the component of $G - S_j$ containing v, so $a_{vw} = 2$. In the second case, $a_{vw} = 0$. Hence,

$$2x(E) \ge \sum_{e \in E} a_e x_e \ge b_0 = 2\sum_{i=1}^{\ell} c(G - S_i) = 2L_c,$$

where $x : E \to \mathbf{R}$ is any feasible solution to the linear program. If x is an optimal solution of the linear program, then we get $2L_z = 2x(E) \ge 2L_c$. This proves the second inequality in the theorem. Moreover, by Corollary 4.4 and the fact that $L_c^\star \le L_c$, we have $\varepsilon(G) \le \frac{17}{12} \max(L_c^\star, L_\varphi) \le \frac{17}{12} L_c \le \frac{17}{12} L_z$. Hence, the first statement in the theorem follows.

Focus on the second statement in the theorem. The multiedge (or uncapacitated) version of our minimum-size 2-ECSS problem is the following: Given $G = (V, E)$ as above, compute $\widetilde{\varepsilon}(G)$, the minimum size (counting multiplicities) of a 2-edge connected spanning submultigraph $H = (V, F)$, where F is a multiset of edges of E. (To give an analogy, if we take $\varepsilon(G)$ to correspond to the f-factor problem, then $\widetilde{\varepsilon}(G)$ corresponds to the f-matching problem.)

FACT 5.2. *If G is a 2-edge connected graph, then $\widetilde{\varepsilon}(G) = \varepsilon(G)$.*

Proof. Let $H = (V, F)$ give the optimal solution for $\widetilde{\varepsilon}(G)$. If H uses two copies of an edge vw, then we can replace one of the copies by some other edge of G in the cut given by $H - \{vw, vw\}$. In other words, if S is the node set of one of the two components of $H - \{vw, vw\}$, then we replace one copy of vw by some edge from $\delta_G(S) - \{vw\}$. □

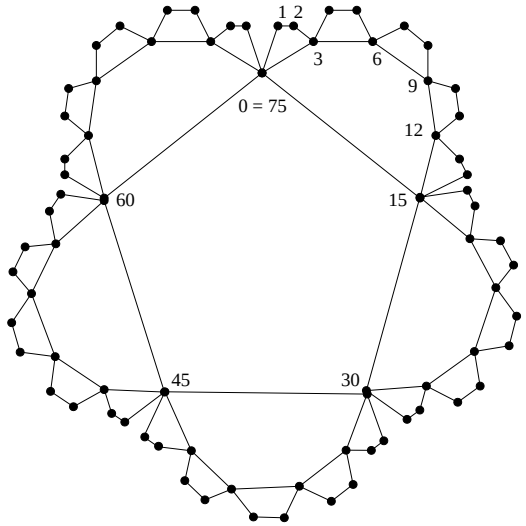
Remark. The above is a lucky fact. It fails to generalize both for minimum-cost (rather than minimum-size) 2-ECSS and for minimum-size k-ECSS, $k \ge 3$.

Given an n-node graph $G = (V, E)$ together with edge costs c (possibly c assigns unit costs), define its *metric completion* G', c' to be the complete graph $K_n = G'$ with c'_{vw} $(\forall\, v, w \in V)$ equal to the minimum-cost of a v-w path in G, c.

FACT 5.3. *Let G be a 2-edge connected graph, and let c assign unit costs to the edges. The minimum cost of the TSP on the metric completion of G, c satisfies $tsp(G', c') \geq \widetilde{\varepsilon}(G) = \varepsilon(G)$.*

Proof. Let T be an optimal solution to the TSP. We replace each edge $vw \in E(T) - E(G)$ by the edges of a minimum-cost v-w path in G, c. The resulting multigraph H is obviously 2-edge connected and has $tsp(G', c') = c(H) \geq \widetilde{\varepsilon}(G)$. ∎

The previous two facts show that $\varepsilon(G) \leq tsp(G', c')$. Moreover, note that for the metric completion G', c', $L_z(G, \mathbf{1})$ equals $L_z(G', c')$, since every feasible solution of the LP on G', c' gives a feasible solution of the LP on G, $\mathbf{1}$ of the same objective value and vice versa. Hence, if the TSP $\frac{4}{3}$ conjecture holds, then we have $\varepsilon(G) \leq tsp(G', c') \leq \frac{4}{3}\, L_z(G', c') = \frac{4}{3}\, L_z$. ∎



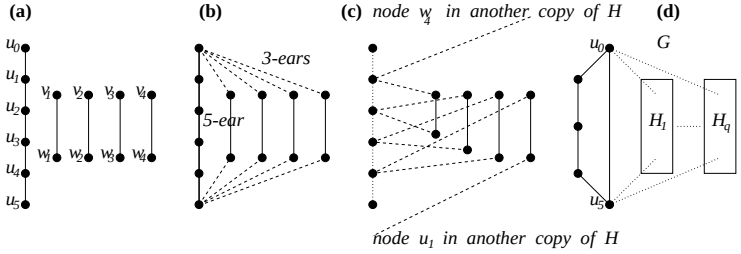


FIG. 2. *Tight examples for our $\frac{17}{12}$-approximation algorithm for minimum-size 2-ECSS. Top: The first example, with $q = 2$. Some of the node labels $0, 1, 2, \ldots, 3 \times 5^q - 1$ are indicated. Bottom: The second example.* (a) *The graph H;* (b) *"covering" the nodes of H by a 5-ear and four 3-ears;* (c) *"covering" the nodes of $H - \{u_0, u_5\}$ by a subpath of a Hamiltonian cycle;* (d) *the graph G.*

5.2. Tight examples. Our analysis of the heuristic is (asymptotically) tight. We give two example graphs. Each is an n-node Hamiltonian graph $G = (V, E)$, where

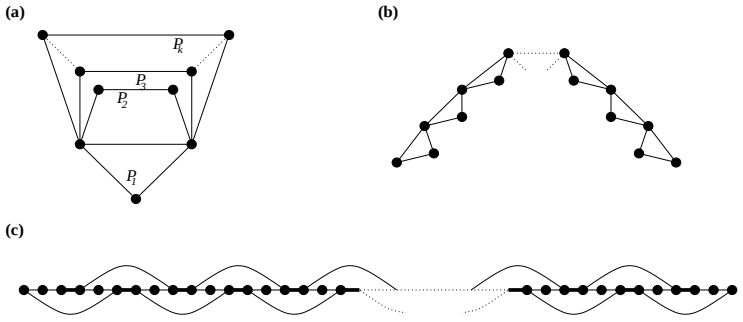


FIG. 3. *Comparing the lower bounds $L_c^\star$ and L_φ with ε. The graphs achieve the following ratios.* (a) $\varepsilon/L_\varphi \geq 1.5 - \Theta(1)/n$. (b) $\varepsilon/L_c^\star \geq 1.5 - \Theta(1)/n$. (c) $\varepsilon/L_c^\star \geq 4/3 - \Theta(1)/n$. *To get a graph with* $\varepsilon/\max(L_c^\star, L_\varphi) \geq 5/4 - \Theta(1)/n$, *subdivide every "thick edge" (3rd edge in path). The resulting graph G has $L_c^\star \leq n+1$ (G has a Hamiltonian path), $L_\varphi \leq n$ (G is factor-critical), and* $\varepsilon = |E(G)| = (5n-7)/4$.

the heuristic (in the worst case) finds a 2-ECSS $G' = (V, E')$ with $17n/12 - \Theta(1)$ edges.

Here is the first example graph, $G = (V, E)$ (see Figure 2 (top)). The number of nodes is $n = 3 \times 5^q$, and $V = \{0, 1, 2, \ldots, 3 \times 5^q - 1\}$. The "first node" 0 will also be denoted 3×5^q. The edge set E consists of (the edge set of) a Hamiltonian cycle together with (the edge sets of) "shortcut cycles" of lengths $n/3, n/(3 \times 5), n/(3 \times 5^2), \ldots, 5$. In detail, $E = \{i(i+1) : 0 \leq i \leq n-1\} \cup \{(3 \times 5^j \times i)(3 \times 5^j \times (i+1)) : 0 \leq j \leq q-1,\ 0 \leq i \leq 5^{q-j} - 1\}$. Note that $|E| = 3 \times 5^q + 5^q + 5^{q-1} + \cdots + 5 = (17 \times 5^q - 5)/4$. In the worst case, the heuristic initially finds 5-ears, and finally finds 3-ears, and so the 2-ECSS (V, E') found by the heuristic has all the edges of G. Hence, we have $|E'|/\varepsilon(G) = |E|/n = 17/12 - 1/(12 \times 5^{q-1})$.

The second example graph, G, (see Figure 2 (bottom)) is constructed by "joining" many copies of the following graph H: H consists of a 5-edge path $u_0, u_1, u_2, u_3, u_4, u_5$, and four disjoint edges $v_1w_1, v_2w_2, v_3w_3, v_4w_4$. We take q copies of H and identify the node u_0 in all copies and identify the node u_5 in all copies. Then we add all possible edges u_iv_j, and all possible edges u_iw_j, i.e., we add the edge set of a complete bipartite graph on all the u-nodes and all the v-nodes, and we add the edge set of another complete bipartite graph on all the u-nodes and all the w-nodes. Finally, we add three more nodes u_1', u_2', u_3' and five more edges to obtain a 5-edge cycle $u_0, u_1', u_2', u_3', u_5, u_0$. Clearly, $\varepsilon(G) = n = 12q + 5$. If the heuristic starts with the closed 5-ear u_0,u_1',u_2',u_3',u_5,u_0, and then finds the 5-ears u_0,u_1,u_2,u_3,u_4,u_5 in all the copies of H, and finally finds the 3-ears $u_0v_jw_ju_5$ $(1 \leq j \leq 4)$ in all the copies of H, then we have $|E'| = 17q + 5$.

How do the lower bounds in Proposition 2.4 (namely, $L_c^\star$) and in Proposition 3.3 (namely, L_φ) compare with ε? Let n denote the number of nodes in the graph. There is a 2-node connected graph such that $\varepsilon/L_\varphi \geq 1.5 - \Theta(1)/n$ (see Figure 3(a)). Therefore the upper bound $|E'| \leq 1.5 \max(L_\varphi, n)$ of Proposition 3.4 is tight. There is another 2-edge connected (but not 2-node connected) graph such that $\varepsilon/L_c^\star \geq 1.5 - \Theta(1)/n$ and $\varepsilon/L_\varphi \geq 1.5 - \Theta(1)/n$ (see Figure 3(b)). Huh [10] uses the proof of Theorem 3.1 of Garg, Santosh, and Singla [8] to show that $\varepsilon \leq 1.5\, L_c^\star$. Among 2-node connected graphs, we have a graph with $\varepsilon/L_c^\star \geq 4/3 - \Theta(1)/n$, but we do not know whether there exist graphs that give higher ratios (see Figure 3(c)). There is a 2-node connected graph such that $\varepsilon/\max(L_c^\star, L_\varphi) \geq 5/4 - \Theta(1)/n$, but we do not know whether there exist graphs that give higher ratios (see Figure 3(c)).

Acknowledgment. We thank the referees for their comments.

REFERENCES

[1] R. Carr and R. Ravi, *A new bound for the* 2*-edge connected subgraph problem*, in Proceedings of the Sixth International Integer Programming and Combinatorial Optimization Conference, Houston, Texas, 1998, Lecture Notes in Comput. Sci. 1412, R.E. Bixby, E.A. Boyd, R.Z. Rios-Mercado, eds., Springer-Verlag, Berlin, pp.112–125.

[2] J. Cheriyan, A. Sebő, and Z. Szigeti, *An improved approximation algorithm for minimum size* 2*-edge connected spanning subgraphs*, in Proceedings of the Sixth International Integer Programming and Combinatorial Optimization Conference, Houston, Texas, 1998, Lecture Notes in Comput. Sci. 1412, R.E. Bixby, E.A. Boyd, R.Z. Rios-Mercado, eds., Springer-Verlag, Berlin, pp. 126–136.

[3] J. Cheriyan and R. Thurimella, *Approximating minimum-size* k*-connected spanning subgraphs via matching*, in Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, IEEE, Los Alamitos, CA, 1996, pp. 292–301; SIAM J. Sci. Comput., 30 (2000), pp. 528–560.

[4] N. Christofides, *Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem*, Technical report, G.S.I.A., Carnegie-Mellon University, Pittsburgh, PA, 1976.

[5] A. Frank, *Conservative weightings and ear-decompositions of graphs*, Combinatorica, 13 (1993), pp. 65–81.

[6] A. Frank, A. Sebő, and E. Tardos, *Covering directed and odd cuts*, Math. Programming Stud., 22 (1984), pp. 99–112.

[7] G. L. Frederickson and J. Ja'Ja', *On the relationship between the biconnectivity augmentation and traveling salesman problems*, Theoret. Comput. Sci., 19 (1982), pp. 189–201.

[8] N. Garg, V. S. Santosh, and A. Singla, *Improved approximation algorithms for biconnected subgraphs via better lower bounding techniques*, in Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, Austin, TX, 1993, ACM, New York, pp. 103–111.

[9] M. X. Goemans and D. J. Bertsimas, *Survivable networks, linear programming relaxations and the parsimonious property*, Math. Program., 60 (1993), pp. 143–166.

[10] T. Huh, *On* 2*-edge connected spanning subgraphs*, M. Math essay, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada, 1998.

[11] S. Khuller and U. Vishkin, *Biconnectivity approximations and graph carvings*, J. ACM, 41 (1994), pp. 214–235. A preliminary version appears in Proceeding of the 24th Annual ACM Symposium on Theory of Computing, ACM, New York, 1992, pp. 759–770.

[12] L. Lovász, *A note on factor-critical graphs*, Studia Sci. Math. Hungar., 7 (1972), pp. 279–280.

[13] L. Lovász and M. D. Plummer, *Matching Theory*, Akadémiai Kiadó, Budapest, 1986.

[14] C. L. Monma, B. S. Munson, and W. R. Pulleyblank, *Minimum-weight two-connected spanning networks*, Math. Program., 46 (1990), pp. 153–171.

[15] S. Vempala and A. Vetta, *Factor* 4/3 *approximations for minimum* 2*-connected subgraphs*, in Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 2000, Springer, Berlin, 2000, pp. 262–273.

[16] H. Whitney, *Nonseparable and planar graphs*, Trans. Amer. Math. Soc., 34 (1932), pp. 339–362.

[17] L. A. Wolsey, *Heuristic analysis, linear programming and branch and bound*, Math. Program. Stud., 13 (1980), pp. 121–134.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 181–192

COMPACT REPRESENTATIONS OF THE INTERSECTION STRUCTURE OF FAMILIES OF FINITE SETS*

JÁNOS KÖRNER† AND ANGELO MONTI†

Abstract. The Nešetřil–Pultr dimension of the Kneser graph is interpreted as the shortest length of strings over an infinite alphabet representing the vertices of the graph so that the absence of coincidences in the codewords of a pair of vertices is equivalent to adjacency, i.e., to the two underlying sets being disjoint. We study analogous but more demanding representations in case the alphabet size may be limited and yet the full intersection has to be determined from the coincidences. Our results introduce a connection between extremal set theory and zero-error problems in multiterminal source coding in the Shannon sense.

Key words. dimension, Kneser graph, coloring

AMS subject classifications. 05C62, 05D05, 68P30, 05C15, 94A24

PII. S0895480198348343

1. Introduction.

Let $[n]$ denote a generic set (alphabet) of n elements and let $\binom{[n]}{k}$ stand for the family of all the k-element subsets of an n-set. Given two of these sometimes we are only interested to know whether or not they intersect. This can be decided on the basis of a compact representation of the sets in the following sense. We denote by $\mathtt{N}$ the set of positive integers. Given the sequences $\mathbf{x} \in \mathtt{N}^t$ and $\mathbf{y} \in \mathtt{N}^t$ let $I(\mathbf{x}, \mathbf{y})$ denote the set of those indices i for which the ith coordinates x_i and y_i of the two sequences are equal. A function $f : \binom{[n]}{k} \to \mathtt{N}^t$ is called a Nešetřil–Pultr representation of the family $\binom{[n]}{k}$ if for any $A \in \binom{[n]}{k}$, $B \in \binom{[n]}{k}$,

$$A \cap B = \emptyset \Leftrightarrow I(f(A), f(B)) = \emptyset.$$

The minimum value $t = t(n, k)$ for which $\binom{[n]}{k}$ has a Nešetřil–Pultr representation is called the Nešetřil–Pultr dimension of the family $\binom{[n]}{k}$. This quantity seems to be very hard to determine. The elements of $\binom{[n]}{k}$ can be considered as the vertices of a graph $G_{\binom{[n]}{k}}$ in which they are adjacent if the two sets in question are disjoint. Such a graph is called a Kneser graph and the above minimum is usually referred to as its Nešetřil–Pultr (or Prague) dimension [11], [12]. (Needless to say that this concept of dimension is defined for arbitrary graphs.) Lovász, Nešetřil, and Pultr [10] have shown that

$$t(2n, n) = \left\lceil \log \binom{2n}{n} \right\rceil.$$

(Note that here and throughout the paper all logarithms are binary.) However, not much more is known about this quantity and it seems to us that $t(\alpha n, n)$ is unknown even for $\alpha = 3$. The asymptotics of $t(n, k)$ is heavily dependent on the relative order of magnitude of k and n. The most investigated case is when k is fixed and only n

*Received by the editors December 2, 1998; accepted for publication December 19, 2000; published electronically March 15, 2001.
http://www.siam.org/journals/sidma/14-2/34834.html
†Department of Computer Science, "La Sapienza" University of Rome, via Salaria 113, 00198 Rome, Italy (korner@dsi.uniroma1.it, monti@dsi.uniroma1.it).

goes to infinity. In particular, Poljak, Pultr, and Rödl [12], [13] (cf. also Poljak and Tuza [14]) have shown that

$$1 \leq \liminf_{n\to\infty} \frac{t(n,k)}{\log\log n} \leq \limsup_{n\to\infty} \frac{t(n,k)}{\log\log n} \leq \frac{k(k-1)}{\log k}.$$

As it was noticed already in [12], the upper bound can be improved for $k = 2$ in a rather elementary manner and as we shall soon see the asymptotics of $t(n,2)$ is known. Combining the representation idea of [12] with the construction of qualitatively independent partitions of Gargano, Körner, and Vaccaro [3] (cf. also [7] and [4]) the upper bound becomes

$$\limsup_{n\to\infty} \frac{t(n,k)}{\log\log n} \leq \frac{k}{2},$$

and this better upper bound has been conjectured tight in [6]. (Note that the lower and upper bounds coincide for $k = 2$.)

Our paper grew out of a failed tentative to prove the conjecture. It seems to us that a novel proof technique would be needed to get a lower bound depending on k. Our analysis has brought about a wealth of new, analogous problems and some nontrivial results. We expect to return to all of these questions in later work.

Since the representation in [12] and even its modified version [3] use a Nešetřil–Pultr representation of $\binom{[n]}{k}$ from which actually the reconstruction of the whole intersection of the set pair is possible, it is natural to ask what happens if we directly ask for such a reconstruction. More precisely, we shall say that a pair of functions

$$f : \binom{[n]}{k} \to \mathtt{N}^t,$$
$$\phi : \{0,1\}^t \to 2^{[n]}$$

is a Bohemian representation of $\binom{[n]}{k}$ if for every pair of distinct sets $A \in \binom{[n]}{k}$, $B \in \binom{[n]}{k}$,

$$\phi(\mathbf{i}(f(A), f(B))) = A \cap B, \tag{1}$$

where $\mathbf{i}(f(A), f(B))$ is the t-length binary characteristic vector of the set $I(f(A), f(B))$. We shall call the number t the length of the representation. Let $T(n,k)$ denote the smallest integer t for which $\binom{[n]}{k}$ has a Bohemian representation. We call $T(n,k)$ the Bohemian dimension of $\binom{[n]}{k}$. Clearly, its determination is a completely new question even though it is not unrelated to the original one. In particular, the new concept has no immediate extension to arbitrary graphs. On the other hand, while the problem of determining the Prague dimension of $\binom{[n]}{k}$ becomes trivial when $k > \frac{n}{2}$, not so for our new dimension. In general, the Bohemian dimension is exponentially larger than the Prague dimension. In a subsequent section we shall show that the concept of Bohemian dimension has a meaningful and perhaps even interesting generalization to arbitrary graphs and hypergraphs.

A more detailed analysis of our problems inevitably leads to the question of how large a subset of $\mathtt{N}$ is effectively needed for an optimal Bohemian representation of $\binom{[n]}{k}$. The analogous question is less interesting in the case of a Prague representation of the Kneser graph $G_{\binom{[n]}{k}}$ since if $f : \binom{[n]}{k} \to \mathtt{N}^t$ is such a representation and $f_i : \binom{[n]}{k} \to \mathtt{N}$

denotes the ith coordinate of its value, then, as it is easily seen, each f_i must be a vertex-coloring of $G_{\binom{[n]}{k}}$ whence $|f_i(\binom{[n]}{k})| \geq n - 2k + 2$ by Lovász' celebrated result on the chromatic number of the Kneser graph [9], [1]. (Actually, one immediately sees that $|f_i(\binom{[n]}{k})| \geq \frac{n}{k}$.) Our present case is different and we will show that Bohemian representation remains possible even for a binary alphabet. More precisely, for any fixed value $b \geq 2$ let us call the pair of functions (f, ϕ) with

$$f : \binom{[n]}{k} \to [b]^t,$$
$$\phi : \{0,1\}^t \to 2^{[n]}$$

a b-limited Bohemian representation of $\binom{[n]}{k}$ if for every pair of distinct sets $A \in \binom{[n]}{k}$, $B \in \binom{[n]}{k}$,

$$\phi(\mathbf{i}(f(A), f(B))) = A \cap B. \tag{2}$$

Further, let $T_b(n,k)$ denote the smallest integer t for which $\binom{[n]}{k}$ has a b-limited Bohemian representation. Clearly, $b \leq c$ implies $T_b(n,k) \geq T_c(n,k) \geq T(n,k)$. The main objective of this paper is to present asymptotic information on all of these quantities for fixed k. In particular, we shall show that all of them grow logarithmically with n.

2. Bohemian dimension. We concentrate our attention on various dimension-type invariants of $\binom{[n]}{2}$. At first glance, this family has little interest, for its Prague dimension is rather easy to determine. In fact, as it was observed in [12] we have

$$\log\log n \leq t(n,2) \leq \lceil \log\log n \rceil. \tag{3}$$

Yet, we have not been able to determine even the Bohemian dimension $T(n,2)$. The following bounds show, nevertheless, that there is an exponential gap between $t(n,2)$ and $T(n,2)$.

THEOREM 1.

$$\frac{3}{\log 5} \leq \liminf_{n\to\infty} \frac{T(n,2)}{\log n} \leq \limsup_{n\to\infty} \frac{T(n,2)}{\log n} \leq 2.$$

Proof. In order to prove the lower bound, consider a Bohemian representation (f, ϕ) of length t. With the help of this Bohemian representation we obtain a mapping

$$F : \binom{[n]}{3} \to \binom{\{0,1\}^t}{3}$$

as follows by setting

$$F(\{x,y,z\}) = \{\mathbf{i}(f(\{x,y\}), f(\{y,z\})), \mathbf{i}(f(\{y,z\}), f(\{z,x\})), \mathbf{i}(f(\{z,x\}), f(\{x,y\}))\}.$$

Denoting

$$\{\phi(\mathbf{i}(f(\{x,y\}), f(\{y,z\}))), \phi(\mathbf{i}(f(\{y,z\}), f(\{z,x\}))), \phi(\mathbf{i}(f(\{z,x\}), f(\{x,y\})))\}$$

by $\Phi(\{x,y,z\})$ we recall that by our hypothesis the latter is the identity mapping on $\binom{[n]}{3}$. On the other hand, it should be clear that Φ is a function of F and therefore

$$\binom{n}{3} = \left|\binom{[n]}{3}\right| = \left|\Phi\left(\binom{[n]}{3}\right)\right| \leq \left|F\left(\binom{[n]}{3}\right)\right|. \tag{4}$$

Given a set $\{x, y, z\}$ the value of $F(\{x, y, z\})$ is an unordered triple of three binary sequences of length t, but it can be thought of, equivalently, as a unique supersequence of length t every supercoordinate of which consists of 3 bits. Thus in every supercoordinate we might see at most eight different triples of bits. However, we would like to argue that only five of these values can actually show up in each supercoordinate. In order to understand this, it suffices to realize that for any $j = 1, \ldots, t$ the jth coordinate of $\mathbf{i}(f(\{x, y\}), f(\{y, z\}))$ equals one iff the corresponding coordinates f_j satisfy

$$f_j(\{x, y\}) = f_j(\{y, z\}),$$

and since if two of these relations occur, then the third also has to hold, it follows that among the three-bit supercoordinates we shall never see those in which the number of bits equal to one is precisely two. This leaves us five choices, and hence

$$\left| F\left(\binom{[n]}{3}\right)\right| \leq 5^t.$$

Comparing this to (4) we see that

$$\binom{n}{3} \leq 5^t,$$

proving the desired lower bound.

Next, in order to establish the promised upper bound on $T(n, 2)$ we shall exhibit a Bohemian representation of $\binom{[n]}{2}$ of length

$$t(n) = \lceil \log \log n \rceil + 2\lceil \log n \rceil.$$

Consider the Prague representation of minimum length $t_1(n)$ of $\binom{[n]}{2}$

$$f_1 : \binom{[n]}{2} \to \mathtt{N}^{t_1(n)}$$

and recall that by the definition of such a representation there exists a function $\phi_1 : \{0, 1\}^{t_1(n)} \to \{0, 1\}$ for which

$$\phi_1(\mathbf{i}(f_1(A), f_1(B))) = 0 \Leftrightarrow A \cap B = \emptyset. \tag{5}$$

Further, since its length is minimum, (3) implies

$$t_1(n) \leq \lceil \log \log n \rceil. \tag{6}$$

In order to proceed, we now introduce some notation. Considering n as fixed, we write $l = l(n) = \lceil \log n \rceil$ and define for every element $m \in [n]$ its binary representation by an arbitrary injection

$$\lambda : [n] \to \{0, 1\}^l.$$

Also, let $\Lambda(m)$ denote the set of coordinates in which the binary vector $\lambda(m)$ equals one. Further, we shall associate with every set $A \in \binom{[n]}{2}$ a private name by another arbitrary injection

$$\nu : \binom{[n]}{2} \to \{\mathtt{N} - [n]\}.$$

Now we are ready to construct our Bohemian representation of $\binom{[n]}{2}$ in the form of the juxtaposition of three mappings $f_h, h = 1, 2, 3$, of which one has been defined above for $h = 1$ and now we complete the picture by introducing the additional two;

$$f_h : \binom{[n]}{2} \to \mathtt{N}^{t_h(n)}, \quad h = 2, 3.$$

First set $h = 2$ and consider a set $A = \{a, b\} \in \binom{[n]}{2}$. We shall denote by $f_2(A, i)$ the ith coordinate of $f_2(A)$ and define it by setting

$$\begin{aligned} f_2(A,i) &= \ a \quad && \text{if} \quad i \in \Lambda(a) - \Lambda(b), \\ f_2(A,i) &= \ b \quad && \text{if} \quad i \in \Lambda(b) - \Lambda(a), \\ f_2(A,i) &= \nu(A) && \text{else.} \end{aligned}$$

Notice that with this definition we have

$$t_2(n) = \lceil \log n \rceil. \tag{7}$$

Next, let $w \in \mathtt{N}$ be an element present neither in $[n]$ nor in $\nu(\binom{[n]}{2})$. We define the coordinates $f_3(A, i)$ of $f_3(A)$ as follows:

$$\begin{aligned} f_3(A,i) &= \ w \quad && \text{if} \quad i \notin \Lambda(a) \cup \Lambda(b), \\ f_3(A,i) &= \nu(A) && \text{else.} \end{aligned}$$

As before, we once again have

$$t_3(n) = \lceil \log n \rceil. \tag{8}$$

We claim that defining $f(A)$ by juxtaposition as

$$f(A) = f_1(A), f_2(A), f_3(A)$$

gives rise to a function

$$f : \binom{[n]}{2} \to \mathtt{N}^{t_1(n)+t_2(n)+t_3(n)}$$

to which we can find another function $\phi : \{0,1\}^{t_1(n)+t_2(n)+t_3(n)} \to 2^{[n]}$ such that the pair (f, ϕ) is a Bohemian representation of $\binom{[n]}{2}$. By the relations (6)–(8) we see that the length of this alleged representation is

$$t(n) = \lceil \log \log n \rceil + 2\lceil \log n \rceil,$$

as claimed. It remains to be seen that a corresponding function ϕ can be found so that (f, ϕ) yields the requested Bohemian representation. To this end, it is sufficient to show that for any two sets $A \in \binom{[n]}{2}$, $B \in \binom{[n]}{2}$ with $A \neq B$ we can reconstruct $A \cap B$ from the $t(n) = t_1(n) + t_2(n) + t_3(n)$-length binary sequence $\mathbf{i}(f(A), f(B))$. If $A \cap B = \emptyset$, then, as observed in (5), we will already realize this upon inspection of the first $t_1(n)$ coordinates of $\mathbf{i}(f(A), f(B))$. Suppose therefore that $A \cap B = \{x\}$. Let us say that $A = \{a, x\}$ and $B = \{b, x\}$ for some $a \neq b$ from $[n]$. Then notice that $\mathbf{i}(f_2(\{a, x\}), f_2(\{b, x\}))$ is the l-length binary characteristic vector $\chi((\Lambda(x) - \Lambda(a)) \cap (\Lambda(x) - \Lambda(b)))$ of the set $[\Lambda(x) - \Lambda(a)] \cap [\Lambda(x) - \Lambda(b)]$. Similarly, by its definition,

$\mathbf{i}(f_3(\{a,x\}), f_3(\{b,x\}))$ is the l-length binary characteristic vector $\chi(\overline{\Lambda(a)\cap\Lambda(x)}\cap\overline{\Lambda(b)\cap\Lambda(x)})$ of the complement of the set $[\Lambda(a)\cap\Lambda(x)]\cup[\Lambda(b)\cap\Lambda(x)]$. Hence knowing the value of both $\mathbf{i}(f_2(\{a,x\}), f_2(\{b,x\}))$ and $\mathbf{i}(f_3(\{a,x\}), f_3(\{b,x\}))$ means knowing both of the sets $[\Lambda(x)-\Lambda(a)]\cap[\Lambda(x)-\Lambda(b)]$ and $[\Lambda(a)\cap\Lambda(b)]\cup[\Lambda(b)\cap\Lambda(x)]$. Now the union of these two is precisely $\Lambda(x)$ and this allows us to complete the construction of a function ϕ as requested. □

As an immediate consequence of the above result, in particular, its upper bound part, we can give a weak although not entirely trivial limitation for the general case.

PROPOSITION 1.

$$k-1 \le \liminf_{n\to\infty} \frac{T(n,k)}{\log n} \le \limsup_{n\to\infty} \frac{T(n,k)}{\log n} \le k(k-1).$$

Proof. The lower bound is an immediate consequence of the fact that if (f,ϕ) form a Bohemian representation of length t of $\binom{[n]}{k}$, then the set

$$\phi\left(\mathbf{i}\left(f\left(\binom{[n]}{k}\right), f\left(\binom{[n]}{k}\right)\right)\right)$$

must contain $\binom{[n]}{k-1}$ and therefore

$$\binom{n}{k-1} \le |\phi\left(\mathbf{i}\left(f\left(\binom{[n]}{k}\right), f\left(\binom{[n]}{k}\right)\right)\right)| \le 2^t.$$

To prove the upper bound, let us fix a Bohemian representation (f,ϕ) of minimum length $T(n,2)$ of $\binom{[n]}{2}$. Further, for an arbitrary set $A\in\binom{[n]}{k}$ and $\{i,j\}\in\binom{[n]}{2}$ let $A(i,j)$ denote the set consisting of the ith and the jth elements of A in the natural ordering of $\mathtt{N}$. We construct a Bohemian representation (F,Φ) of $\binom{[n]}{k}$ by juxtaposition. More precisely, we define the value $F(A)$ of any $A\in\binom{[n]}{k}$ by juxtaposing the f-images of the two-element subsets of A in some fixed order, independent of A. Thus we write

$$F(A) = f(A(1,2)), f(A(1,3)), \ldots, f(A(k-1,k)).$$

The construction works because we can obtain the intersection of any two k-element sets as the union of the intersections of their respective two-element subsets;

$$A\cap B = \bigcup_{\{i,j\}\in\binom{[k]}{2}} [A(i,j)\cap B(i,j)].$$

With the function Φ defined in a similar piecewise fashion we are therefore enabled to represent the intersections of the two-element subsets at length $\lceil\log n\rceil$ by the function $f(i,j)$ and its counterpart ϕ. To conclude, notice that the overall length of the representation (F,Φ) is $2\binom{k}{2}\lceil\log n\rceil$ and this proves the upper bound. □

3. Binary representations. We have mentioned that in the case of a Prague representation of $\binom{[n]}{2}$ the alphabet size must necessarily grow to infinity with the size of the ground set; not so in our case. We shall illustrate this difference for the simplest situation of representing $\binom{[n]}{2}$ by a binary alphabet. We will show, however, that as one might expect, there is a penalty to be paid in the sense that even the best binary representation must be longer than its unlimited alphabet size counterpart.

We introduce the shorthand notation $T(n) = T_2(n,2)$. Our main result is the following theorem.

THEOREM 2.

$$3 \le \liminf_{n\to\infty} \frac{T(n)}{\log n} \le \limsup_{n\to\infty} \frac{T(n)}{\log n} \le 6.$$

Proof. To prove the lower bound consider a 2-limited Bohemian representation (f, ϕ) of length t of $\binom{[n]}{2}$,

$$f : \binom{[n]}{2} \to \{0,1\}^t, \qquad \phi : \{0,1\}^t \to 2^{[n]}.$$

By definition, we must have

$$\phi(\mathbf{i}(f(A), f(B))) = A \cap B.$$

Let us fix an arbitrary $a \in [n]$. We claim that the values

$$\mathbf{i}(f(\{a,x\}), f(\{y,z\}))$$

must all be different provided that the sets $\{x,y,z\} \in \binom{[n]-a}{3}$ are also. (More could be said, but this much will prove sufficient.) To verify the previous statement, suppose to the contrary that

$$\mathbf{i}(f(\{a,x\}), f(\{y,z\})) = \mathbf{i}(f(\{a,x'\}), f(\{y',z'\})) \quad \text{for some} \quad \{x,y,z\} \ne \{x',y',z'\},$$

where we have $\{x,y,z\}$ and $\{x',y',z'\} \in \binom{[n]-a}{3}$. Noticing that for any two binary sequences $\mathbf{x}$ and $\mathbf{y}$ of length t the value of $\mathbf{i}(\mathbf{x},\mathbf{y})$ is just the binary complement of their sum $\mathbf{x} \oplus \mathbf{y}$ in vector addition modulo 2, one immediately sees that the previous equality implies

$$\mathbf{i}(f(\{a,x\}), f(\{a,x'\})) = \mathbf{i}(f(\{y,z)\}, f(\{y',z'\})).$$

On the other hand, we know that

$$\phi(\mathbf{i}(f(\{a,x\}), f(\{a,x'\}))) = \{a,x\} \cap \{a,x'\} = \{a\}$$

while

$$\phi(\mathbf{i}(f(\{y,z\}), f(\{y',z'\}))) = \{y,z\} \cap \{y',z'\}$$

and since the sets on the right-hand side of the last two relations are different, we get a contradiction. We conclude that

$$\binom{n-1}{3} \le \left| \left\{ \mathbf{i}(f(\{a,x\}), f(\{y,z\}))\,;\, \{x,y,z\} \in \binom{[n]-a}{3} \right\} \right| \le 2^t.$$

Comparing the two ends of this chain of inequalities and taking logarithms we get the desired lower bound.

To prove the upper bound we will find a Bohemian representation by a routine random choice argument. Let us consider the family $\mathcal{G}$ of all the functions $f : \binom{[n]}{2} \to \{0,1\}^t$ and let Γ be a random variable with uniform distribution on $\mathcal{G}$. Then, clearly, for any ordered pair of distinct and fixed two-element subsets A and B of $[n]$ and for any fixed binary string $\mathbf{x} \in \{0,1\}^t$ we have

$$\mathtt{Pr}\{\mathbf{i}(\Gamma(A), \Gamma(B)) = \mathbf{x}\} = 2^{-t}.$$

Actually, we can even say that given any pair of different set pairs $(A,B) \in \binom{[n]}{2} \times \binom{[n]}{2}$ and $(A',B') \in \binom{[n]}{2} \times \binom{[n]}{2}$ satisfying

$$A \neq B,\ A' \neq B' \quad \text{and} \quad \{A,B\} \neq \{A',B'\}$$

and any pair of (not necessarily distinct) sequences $(\mathbf{x},\mathbf{y}) \in \{0,1\}^t \times \{0,1\}^t$ we have

$$\mathtt{Pr}\{\mathbf{i}(\Gamma(A),\Gamma(B)) = \mathbf{x},\ \mathbf{i}(\Gamma(A'),\Gamma(B')) = \mathbf{y}\} = 2^{-2t}.$$

Hence it follows that

$$\mathtt{Pr}\{\mathbf{i}(\Gamma(A),\Gamma(B)) = \mathbf{i}(\Gamma(A'),\Gamma(B'))\} = \sum_{\mathbf{x}\in\{0,1\}^t} 2^{-2t} = 2^{-t}. \tag{9}$$

Next notice that we don't always have to avoid the equality

$$\mathbf{i}(\Gamma(A),\Gamma(B)) = \mathbf{i}(\Gamma(A'),\Gamma(B')). \tag{10}$$

In fact, this equality causes us no problem if $|A\cup B\cup A'\cup B'| = 8$ for this means that both intersections $A\cap B$ and $A'\cap B'$ are empty and therefore equal. On the other hand, we will have to avoid a situation leading to (10) in all the other cases. This means that the total number of configurations of pairs of pairs from $[(A,B),(A',B')] \in \binom{[n]}{2}^4$ which we have to keep under control corresponds to those pairs of pairs of sets for which we have

$$|A\cup B\cup A'\cup B'| \leq 7, \tag{11}$$

and this amounts to a total of at most n^7 configurations from $\binom{[n]}{2}^4$. For an arbitrary function $f : \binom{[n]}{2} \to \{0,1\}^t$ now let $B(f)$ denote the set of pairs of pairs $[(A,B),(A',B')] \in \binom{[n]}{2}^4$ satisfying both (11) and (10). For the expectation of $|B(f)|$ we have

$$\mathtt{E}|B(\Gamma)| = \sum_{[(A,B),(A',B')]\in\binom{[n]}{2}^4} \mathtt{Pr}\{[(A,B),(A',B')] \in B(\Gamma)\} \leq n^7\cdot 2^{-t}, \tag{12}$$

where the last inequality is a consequence of (9). Now choose t so as to satisfy $n^7\cdot 2^{-t} \leq \frac{n}{2}$. To this end, we specify

$$t = t(n) = 6\lceil \log n\rceil + 1. \tag{13}$$

With this choice (12) implies that

$$\mathtt{E}|B(\Gamma)| \leq \frac{n}{2},$$

whence we can conclude that there exists a function $f : \binom{[n]}{2} \to \{0,1\}^t$ for which $|B(f)| \leq \frac{n}{2}$. Let us now omit from our ground set $[n]$ for each of the configurations $[(A,B),(A',B')] \in B(f)$ an arbitrary element from the union of the underlying set $A\cup B\cup A'\cup B'$. The remaining ground set will have at least $\frac{n}{2}$ elements and without loss of generality we can suppose that it is just $[m]$ for $m = \lceil \frac{n}{2}\rceil$. Restricting our function f to this new ground set will result in its having no bad configurations. Thus our f gives rise to a Bohemian representation of $\binom{m}{2}$ of length $t(m) = 6\lceil \log m\rceil + 7$, concluding the proof. □

Remark (added later). The upper bound in this theorem can be lowered from 6 to 4 if instead of the above random choice argument one uses a well-known result of Lindström [8].

4. A wealth of dimensions. So far we have considered only the Bohemian representation of the family of all the k-sets of an $[n]$-set. The reason for this limitation is that only in this case (related to Kneser graphs) do we have results of some significance. It should be clear, however, that our concepts extend to arbitrary set families (i.e., hypergraphs in the sense of Berge [1]). As a consequence of this extension we will be able to define the Bohemian dimension of arbitrary graphs. This and similar new concepts of dimension are the topic of this section. We will describe definitions and open problems, thereby offering a framework for further research.

A hypergraph is a family of subsets of a set but for most purposes what matters is only what we would like to call its intersection structure. A (finite) hypergraph $H = (V(H), E(H))$ is an ordered couple of finite sets satisfying the relation $E(H) \subseteq 2^{V(H)}$. The elements of $V(H)$ are called vertices, those of $E(H)$ are called hyperedges. The intersection structure of H is a function $\sigma_H : \binom{E(H)}{2} \to 2^{V(H)}$ for which

$$\sigma_H(\{A, B\}) = A \cap B \quad \text{for every} \quad \{A, B\} \in \binom{E(H)}{2}. \tag{14}$$

If we determine the intersection structure of a hypergraph, then we will not necessarily know it completely, since the intersection structure gives us no information on those vertices in $V(H)$ that belong to a single hyperedge. Actually, what we will not know is just how many of them the various hyperedges contain, and this is usually quite irrelevant.

Definition 1. *A pair of functions*

$$f : E(H) \to \mathtt{N}^t, \qquad \phi : \{0,1\}^t \to 2^{V(H)}$$

constitute a Bohemian representation of the hypergraph $H = (V(H), E(H))$ if

$$\phi(\mathbf{i}[f(A), f(B)]) = \sigma_H(\{A, B\}) \qquad \textit{for every} \qquad \{A, B\} \in \binom{E(H)}{2}.$$

The integer t is called the length of the representation. The smallest integer t for which H has a Bohemian representation is called its dimension.

Analogously, we can define the b-limited Bohemian dimension of a hypergraph; we omit the details.

In this language the problem treated in section 2 regards the Bohemian dimension of the complete k-uniform hypergraph on n vertices. Next we would like to explain that in our view there is a strong connection between the concept of Bohemian and the Nešetřil–Pultr dimension of graphs and hypergraphs in the sense that both of them are analogous special instances of a more general concept. For the sake of simplicity of notation let us suppose for the rest of our discussion that all our hypergraphs have a vertex set contained in $\mathtt{N}$.

Definition 2. *Given an arbitrary function $\tau : 2^{\mathtt{N}} \to \mathtt{R}$ a pair of functions*

$$f : E(H) \to \mathtt{N}^t, \qquad \phi : \{0,1\}^t \to \mathtt{R}$$

is called a τ-partial Bohemian representation of the hypergraph H if

$$\phi(\mathbf{i}[f(A), f(B)]) = \tau(A \cap B) \qquad \textit{for every} \qquad \{A, B\} \in \binom{E(H)}{2}.$$

Once again, the integer t is called the length of the representation. The smallest integer t for which H has a τ-partial Bohemian representation is called its τ-partial dimension.

If τ is the identity function, the last definition gives back the previous one. Changing the definition of the function τ we obtain various new characteristics interpretable as partial dimension. Other natural choices are the cardinality function τ_1 defined by

$$\tau_1(A) = |A|$$

and the positivity function τ_2 defined by setting

$$\begin{aligned} \tau_2(A) = 0 \quad &\text{if} \quad |A| = 0, \\ 1 \quad &\text{if} \quad |A| > 0. \end{aligned}$$

Clearly, the corresponding dimensions of the same hypergraph are decreasing in the order of introduction of these concepts. Their determination represents a lot of new problems in this area.

There are many ways to arrive at further generalizations and variations of our problems. One of these is a consequence of replacing the function $\mathbf{i}$ by a different coincidence function, $\iota : \mathtt{N} \times \mathtt{N} \to \mathtt{N} \cup \{0\}$, defined by setting

$$\begin{aligned} \iota(a,b) = a \quad &\text{if} \quad a = b, \\ 0 \quad &\text{if} \quad a \neq b. \end{aligned}$$

This function represents a majority vote on the two values in the same coordinate position. Further, let $\mathbf{c}$ denote the usual extension of ι to t-length sequences by juxtaposition, i.e., let us have

$$\mathbf{c}([\mathbf{x}, \mathbf{y}]) = \iota(x_1, y_1) \dots \iota(x_t, y_t).$$

We conclude this section by defining the corresponding dimension, to be called the Malá Strana dimension of a hypergraph $H = (V(H), E(H))$.

Definition 3. *A pair of functions*

$$f : E(H) \to \mathtt{N}^t, \qquad \phi : (\mathtt{N} \cup \{0\})^t \to 2^{V(H)}$$

constitute a Malá Strana representation of the hypergraph $H = (V(H), E(H))$ if

$$\phi(\mathbf{c}[f(A), f(B)]) = A \cap B \qquad \textit{for every} \qquad \{A, B\} \in \binom{E(H)}{2}.$$

The integer t is called the length of the representation. The smallest integer t for which H has a Malá Strana representation is called its Malá Strana dimension.

A rapid analysis of the construction idea of [12] can convince the reader that in view of the already cited result of [3] we have the following proposition.

Proposition 2. *The Malá Strana dimension $M(n,k)$ of the complete k-uniform hypergraph on n vertices satisfies*

$$\limsup_{n\to\infty} \frac{M(n,k)}{\log\log n} \leq \frac{k}{2}.$$

We omit the proof since it is an easy exercise for anyone familiar with the above cited results of [12] and [3]. Compared to Bohemian dimension of the same hypergraph, our last observation is telling us that Malá Strana dimension is on the same side of the exponential gap with the Prague dimension of these hypergraphs. Unfortunately, we don't know any nontrivial lower bound for the Malá Strana dimension of

complete uniform hypergraphs. With respect to the Prague representation we have two different conditions affecting the numerical result in opposite directions. On the one hand, the majority function **c** gives more information on the comparison of the values of f but on the other hand, in the definition of Malá Strana representation, we drop the restriction that disjoint sets should be represented by f-vectors with no coincidence in their coordinates.

5. Dimension and information theory. For those familiar with the Shannon theory of information and especially with coding theorems for multiterminal sources all the above might ring a bell. All our models are very much like zero-error problems in multiuser information theory. Considering Bohemian representations, one sees that the functions (f, ϕ) are a coder-decoder pair, the comparator function **i** is playing the role of a multiaccess channel among other things. Although our present problems are not zero-error versions of coding problems in information theory, there is still a conceptual connection as it should be. Here and there one is dealing with compact representations of a given structure in a prescribed form. Here and there one expects that the theoretical limit for the "size" of such a representation is a "measure of information content." Since this paper is meant to be self-contained from the point of view of readers interested in combinatorics, we will not elaborate on the connections, but those interested can use the book [2] or the survey article [6] and the references therein. On the other hand, we can consider our models as part of the effort of analyzing functional complexity in the spirit of Shannon's seminal paper [15]. A more precise connection is offered through the concept of relative capacity capturing the extent to which the powers of a graph can mimic those of another one. This notion, a true generalization of Shannon's capacity [16] of graphs was introduced in [5] and we shall return to its relevance for the Prague dimension elsewhere.

6. A tribute. This paper is a late tribute to the living memory of Svata Poljak whose beautiful invention of combining constructions of perfect hashing and qualitatively independent partitions into Nešetřil–Pultr representations (as described in [12]) motivated our research.

REFERENCES

[1] C. Berge, *Hypergraphes, Combinatoire des ensembles finis*, Gauthier–Villars, Paris, 1987.

[2] I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*, Academic Press, New York, 1982 and Akadémiai Kiadó, Budapest, 1981.

[3] L. Gargano, J. Körner, and U. Vaccaro, *Sperner capacities*, Graphs Combin., 9 (1993), pp. 31–46.

[4] L. Gargano, J. Körner, and U. Vaccaro, *Capacities: From information theory to extremal set theory*, J. Combin. Theory Ser. A, 68 (1994), pp. 296–316.

[5] J. Körner and K. Marton, *Relative Shannon capacity of graphs*, in Proceedings of the IEEE Symposium on Information Theory, Ann Arbor, 1986.

[6] J. Körner and A. Orlitsky, *Zero-error information theory. Information theory*: 1948–1998, IEEE Trans. Inform. Theory, 44 (1998), pp. 2207–2229.

[7] J. Körner and G. Simonyi, *A Sperner-type theorem and qualitative independence*, J. Combin. Theory Ser. A, 59 (1992), pp. 90–103.

[8] B. Lindström, *Determination of two vectors from the sum*, J. Combin. Theory Ser. A, 6 (1969), pp. 402–407.

[9] L. Lovász, *Kneser's conjecture, chromatic number and homotopy*, J. Combin. Theory Ser. A, 25 (1978), pp. 319–324.

[10] L. Lovász, J. Nešetřil, and A. Pultr, *On a product dimension of graphs*, J. Combin. Theory Ser. B, 29 (1980), pp. 47–67.

[11] J. Nešetřil and A. Pultr, *Product and other representations of graphs and related characteristics*, in Algebraic Methods in Graph Theory, Colloq. Math. Soc. János Bolyai 25, North-Holland, Amsterdam, 1981, pp. 571–598.
[12] S. Poljak, A. Pultr, and V. Rödl, *On the dimension of Kneser graphs*, in Algebraic Methods in Graph Theory, Colloq. Math. Soc. János Bolyai 25, North-Holland, Amsterdam, 1981, pp. 631–646.
[13] S. Poljak, A. Pultr, and V. Rödl, *On qualitatively independent partitions and related problems*, Discrete Appl. Math., 6 (1983), pp. 193–205.
[14] S. Poljak and Z. Tuza, *On the maximum number of qualitatively independent partitions*, J. Combin. Theory Ser. A, 51 (1989), pp. 111–116.
[15] C. E. Shannon, *The synthesis of two–terminal switching circuits*, Bell System Tech. J., 28 (1949), pp. 59–98.
[16] C. E. Shannon, *The zero–error capacity of a noisy channel*, IRE Trans. Inform. Theory, 2 (1956), pp. 8–19.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 193–206

SORTING STRINGS BY REVERSALS AND BY TRANSPOSITIONS*

DAVID A. CHRISTIE† AND ROBERT W. IRVING†

Abstract. The problems of sorting by reversals and sorting by transpositions have been studied because of their applications to genome comparison. Prior studies of both problems have assumed that the sequences to be compared (or sorted) contain no duplicates, but there is a natural generalization in which the sequences are allowed to contain repeated characters. In this paper we study primarily the versions of these problems in which the strings to be compared are drawn from a binary alphabet. We obtain upper and lower bounds for reversal and transposition distance and show that the problem of finding reversal distance between binary strings, and therefore between strings over an arbitrary fixed-size alphabet, is NP-hard.

Key words. strings, sorting, genome comparison, reversals, transpositions, NP-complete problems

AMS subject classifications. 68R05, 68R15, 68Q17, 92D15

PII. S0895480197331995

1. Introduction.

1.1. Reversals and transpositions over permutations.

The reversal distance and the transposition distance between two permutations (and the related problems of sorting by reversals and sorting by transpositions) are used to estimate the number of global mutations between genomes and can be used by molecular biologists to infer evolutionary and functional relationships between genomes. A *reversal* (or inversion) involves reversing the order of elements in a substring of the permutation. More formally, the reversal $\rho(i,j)$ $(1 \le i < j \le n)$ transforms the permutation π of $\{1,2,\ldots,n\}$ into π', where

$$\pi'(k) = \begin{cases} \pi(i+j-k) & \text{if } i \le k \le j, \\ \pi(k) & \text{otherwise.} \end{cases}$$

A *transposition* involves swapping two adjacent substrings of the permutation. More formally, the transposition $\tau(i,j,k)$ $(1 \le i < j < k \le n+1)$ transforms the permutation π of $\{1,2,\ldots,n\}$ into π', where

$$\pi'(m) = \begin{cases} \pi(m+j-i) & \text{if } i \le m < i+k-j, \\ \pi(m-k+j) & \text{if } i+k-j \le m < k, \\ \pi(m) & \text{otherwise.} \end{cases}$$

Sorting by reversals is the problem of finding the minimum number $d_r(\pi)$ of reversals needed to transform a given permutation π into the identity permutation $\imath$. *Sorting by transpositions* is the analogous problem of determining $d_t(\pi)$, the minimum number of transpositions needed to transform π into $\imath$. The functions $d_r(\pi)$ and $d_t(\pi)$ are known as the *reversal distance* and *transposition distance*, respectively, of π.

In the context of sorting by reversals, Kececioglu and Sankoff [14] introduced the concept of a breakpoint. A permutation π of $\{1,2,\ldots,n\}$ has a *breakpoint* at position

*Received by the editors December 31, 1997; accepted for publication (in revised form) December 21, 2000; published electronically March 15, 2001.
http://www.siam.org/journals/sidma/14-2/33199.html
†Department of Computing Science, University of Glasgow, Glasgow G12 8RZ, Scotland, UK (christie@dcs.gla.ac.uk, rwi@dcs.gla.ac.uk).

i if $|\pi(i) - \pi(i-1)| \neq 1$. (Special elements $\pi(0) = 0$ and $\pi(n+1) = n+1$ are added so that breakpoints at the ends of the permutation are included.) Consideration of breakpoints leads to simple lower and upper bounds for reversal distance [14]. The *reversal diameter* of the symmetric group S_n is the maximum value of $d_r(\pi)$ over all permutations of length n. Bafna and Pevzner [1] have proved that the reversal diameter of S_n is $n-1$ and is achieved by only two permutations of length n.

In the transposition case, Bafna and Pevzner [2] defined breakpoints slightly differently. Here, π has a breakpoint at position i if $\pi(i) - \pi(i-1) \neq 1$, and once again, consideration of breakpoints leads to simple lower and upper bounds for transposition distance [2]. The *transposition diameter* of S_n is the maximum value of $d_t(\pi)$ over all permutations of length n. The transposition diameter of S_n has not been resolved, but Bafna and Pevzner [2] have shown that it lies somewhere between $n/2 + 1$ and $3n/4$.

Caprara [4] (see also [5]) has shown that sorting by reversals is NP-hard. Earlier, Kececioglu and Sankoff [14] had found a simple 2-approximation algorithm and Bafna and Pevzner [1] a 7/4-approximation algorithm. Christie [7] has obtained a 3/2-approximation algorithm, which is the best currently known for this problem.

There is a variation of sorting by reversals in which each element of the permutation is given a sign "+" or "−" that is flipped when the element is involved in a reversal. Perhaps surprisingly, this version of the problem is solvable in polynomial time, as was proved by Hannenhalli and Pevzner [11] (see also [12]). Their algorithm to find signed reversal distance has been improved and simplified by Berman and Hannenhalli [3] and also by Kaplan, Shamir, and Tarjan [13].

Sorting by transpositions is less well understood than sorting by reversals, and in particular, the complexity of sorting by transpositions remains open. However, Bafna and Pevzner [2] have described a 3/2-approximation algorithm for the problem.

1.2. Reversals and transpositions over strings. In the context of genome comparisons, duplicate genes can occur, so that the permutation model is not always the appropriate one. In this paper, we define reversal distance and transposition distance on strings and investigate these new problems, which are of interest in their own right, focusing primarily on the case of a binary alphabet. However, some of the bounds that we establish can be extended to arbitrary fixed-size alphabets, and our main NP-hardness result—for reversal distance over a binary alphabet—immediately implies NP-hardness of the corresponding problem over an arbitrary fixed-size alphabet.

For permutations, transforming π into ρ is equivalent to transforming $\rho^{-1} \cdot \pi$ into ι. However, there is no direct analogue of this result for strings. Therefore in this context the problems are expressed in terms of the distance between two strings.

For strings S and T, the *reversal distance* $d_r(S,T)$ between S and T is the minimum number of reversals required to transform S into T; and the *transposition distance* $d_t(S,T)$ is the minimum number of transpositions required to transform S into T. It is impossible to insert or delete characters using reversals or transpositions, so in each case T must be a rearrangement of S. We say that S and T are *related* if T is a rearrangement of S.

Note that the sorting problem on permutations is a special case of the distance problem on strings. Therefore, for reversals and transpositions, the distance problem on strings is at least as hard as the sorting problem on permutations. Thus, finding the reversal distance between strings is NP-hard since sorting permutations by reversals is NP-hard. However, if the strings are drawn from a fixed-size alphabet, then minimally

sorting a permutation is no longer a special case of finding the distance between two strings, and the hardness of sorting permutations by reversals does not imply a corresponding result for sorting strings by reversals in this case.

The remainder of the paper is organized as follows. Section 2 introduces appropriate terminology and notation. Section 3 is devoted to reversals and section 4 to transpositions. In these two sections lower bounds, upper bounds, and diameter results are described for each problem, respectively. In section 5 it is proved that the generalized version of sorting by reversals is NP-hard, even when the strings are drawn from a binary alphabet. In the final section we compare and contrast the results obtained for binary strings with the known results for sorting problems over permutations.

2. Terminology and notation. We denote the ith symbol of a string S by $S(i)$. A reversal on a string will be represented by enclosing in brackets the substring to be reversed. For example, $0[1010110]1 = 001101011$. A similar notation, in which two adjacent substrings are bracketed, will be used to describe transpositions—for instance, $0[10][1011]01 = 010111001$.

Let 0^k represent a string of zeros of length k, 1^k a string of ones of length k, and, in general, let S^k (or $(S)^k$) represent the string obtained by concatenating k copies of S for any string S. We use $S \cdot T$, or simply ST, to denote the concatenation of strings S and T, and we define a string concatenation operation $(\sum)$ that can be used in a similar way to summation. For example, $\sum_{i=1}^{3}(0^i1) = 010010001$. We also use the standard notation S^+ to represent the concatenation of one or more copies of S and S^* the concatenation of zero or more copies of S.

Let $\mathcal{B}_n$ be the set of binary strings of length n. For a particular transformation, the *diameter* of $\mathcal{B}_n$ is the maximum distance between any two related binary strings of length n. Define $E_k = 0^k1^k$ and $C_k = (10)^k$. For example, $E_4 = 00001111$ and $C_4 = 10101010$. These strings are particularly useful for establishing diameter results in later sections.

Let $\overline{S}$ represent the string derived from S by switching ones and zeros, and let S^R represent the string S in reverse order. Therefore, for example, if $S = 0100110001$, then $\overline{S} = 1011001110$, $S^R = 1000110010$, and $\overline{S}^R = 0111001101$.

Strings X and Y are *isomorphic* to strings S and T if

(i) $X = S$ and $Y = T$, or $X = T$ and $Y = S$, or
(ii) $X = \overline{S}$ and $Y = \overline{T}$, or $X = \overline{T}$ and $Y = \overline{S}$, or
(iii) $X = S^R$ and $Y = T^R$, or $X = T^R$ and $Y = S^R$, or
(iv) $X = \overline{S}^R$ and $Y = \overline{T}^R$, or $X = \overline{T}^R$ and $Y = \overline{S}^R$.

In other words, the pairs $\{X, Y\}$ and $\{S, T\}$ are isomorphic if one pair can be obtained from the other by a fixed permutation of the alphabet, followed by an optional complete reversal of both strings in the pair. This version of the definition applies equally to strings over an arbitrary alphabet. Obviously, if X and Y are isomorphic to S and T, then $d_r(X, Y) = d_r(S, T)$ and $d_t(X, Y) = d_t(S, T)$.

Define $lcp(S, T)$ and $lcs(S, T)$ to be the lengths of the longest common prefix and the longest common suffix, respectively, of S and T.

A *block of zeros* is a maximal length substring that consists only of the character 0. A *block of ones* is defined similarly. Let $b(S)$ denote the total number of blocks in S and $z(S)$ denote the number of blocks of zeros in S. Therefore, for example, $b(001110101) = 6$ and $z(001110101) = 3$.

Finally, we represent by $\ldots$ an arbitrary substring of length ≥ 0. For example,

if S has prefix "01," a substring "00," and suffix "11," then we could write $S = 01\ldots00\ldots11$.

3. Reversal distance between binary strings. In this section, we describe a lower bound and an upper bound for reversal distance between binary strings. These bounds are then used to determine the reversal diameter of $\mathcal{B}_n$ and also to identify some strings that achieve this reversal diameter. A restricted version of the problem, that is in some sense analogous to sorting permutations by reversals, is shown to be solvable in polynomial time. However, in section 5 the general problem of determining reversal distance between two (binary) strings is shown to be NP-hard.

3.1. A lower bound. We first adapt the concept of breakpoint from permutation sorting problems for use in the context of string sorting. This new kind of breakpoint is then used to establish a lower bound for reversal distance. Recall that, for permutations, two elements form a breakpoint if they are adjacent in π but not adjacent in the identity permutation. Substrings of length two represent adjacencies in strings S and T, so our definition of breakpoints on strings will be based on these substrings.

If S contains more "00" substrings than T, then each extra "00" must be broken, by a reversal, at some time in the transformation from S into T. Each extra "00" in S is an example of a *reversal breakpoint*. An obvious difference between breakpoints on strings and on permutations is that, on strings, the specific location of a breakpoint may not necessarily be identified. For instance, if S contains three "00" substrings and T contains only two "00" substrings, then one of the "00" substrings in S is a breakpoint, but no particular "00" substring of S is identified as the breakpoint. Breakpoints also occur for "01," "10," and "11" substrings as well. However, because reversals can convert "01" substrings into "10" substrings and vice versa, these substrings must be counted together when considering reversal breakpoints.

Breakpoints can also be contributed from the beginning and end of the strings. For example, if $S(1) \neq T(1)$, then position one contributes a breakpoint. In order to deal with these breakpoints, S and T are extended by adding special characters α at the beginning and ω at the end of both strings. These breakpoints can then be counted by comparing the number of occurrences of the substrings "$\alpha 0$," "$\alpha 1$," "0ω," and "1ω" in both strings. Adding α and ω to S and T is similar to adding 0 and $n+1$ to π when dealing with permutations.

The number of times the substring "ab" occurs in S, i.e., the frequency count for "ab" in S, is denoted by $f_{ab}(S)$, where $a, b \in \{\alpha, 0, 1, \omega\}$. We also assume, for convenience, $\alpha < 0 < 1 < \omega$.

We now define the number of reversal breakpoints between S and T, $b_r(S,T)$ to be

$$b_r(S,T) = \sum_{\alpha \le a < b \le \omega} \delta(f_{ab}(S) + f_{ba}(S) - f_{ab}(T) - f_{ba}(T)) + \sum_{0 \le a \le 1} \delta(f_{aa}(S) - f_{aa}(T)),$$

where

$$\delta(x) = x \text{ if } x > 0 \text{ and } 0 \text{ otherwise.}$$

Clearly, if $S = T$, then $b_r(S,T) = 0$. However, it is possible to have $b_r(S,T) = 0$, even when $S \neq T$, for example, if $S = 100101$ and $T = 101001$.

Note that, although the definition of the number of breakpoints is not symmetric with respect to the two strings S and T, it is easy to see that $b_r(S,T) = b_r(T,S)$.

We now derive a lower bound for reversal distance based on these breakpoints.

LEMMA 3.1. *Suppose that S' is obtained from S by a single reversal. Then*

$$b_r(S', T) \geq b_r(S, T) - 2.$$

Proof. A reversal on S cuts two substrings of length two in the extended version of S. Hence the number of breakpoints can be reduced by at most two as a result of such a reversal, and the result follows. □

This lemma can be used to easily deduce the following lower bound for reversal distance.

THEOREM 3.2. *Let S and T be related binary strings. Then*

$$d_r(S, T) \geq \lceil b_r(S, T)/2 \rceil.$$

It is easy to find examples for which this lower bound is not tight; e.g., if $S = 0011000111$ and $T = 1110011000$, then $b_r(S, T) = 2$ and $d_r(S, T) = 2 \neq \lceil b_r(S, T)/2 \rceil$.

The definition of a reversal breakpoint for binary strings, together with the bound of Theorem 3.2, can be generalized in a straightforward way to strings over larger alphabets.

3.2. An upper bound. In this section we derive a simple upper bound on reversal distance for binary strings.

LEMMA 3.3. *Let S and T be related strings of length n such that $S \neq T$. Then it is possible either*

(a) *to apply a reversal on S resulting in the string S', such that $lcp(S', T) + lcs(S', T) \geq lcp(S, T) + lcs(S, T) + 2$, or*

(b) *to apply a reversal on T resulting in the string T' such that $lcp(S, T') + lcs(S, T') \geq lcp(S, T) + lcs(S, T) + 2$.*

Proof. Without loss of generality, it can be assumed that $S(1) = 0$, since otherwise we could consider $\overline{S}$ and $\overline{T}$ and apply the resulting reversal to S or T. Further, it can be assumed that $S(1) \neq T(1)$, and $S(n) \neq T(n)$, because otherwise we could reduce n by removing any common prefix or suffix from both strings.

The reversal applied to S or T depends on S and T. We describe seven cases and show a reversal with the required property in each case. (Each case is considered only if S and T fail to meet the conditions of any of the earlier cases, and the cases are thereby mutually exclusive.)

Case (i). $S(n) = 1$: then $S = 0 \ldots 1$ and $T = 1 \ldots 0$, so take $S' = [0 \ldots 1]$.

Case (ii). $T(2) = 0$: then $S = 0 \ldots 0$ and $T = 10 \ldots 1$, so take $S' = [0^+1] \ldots 0$.

Case (iii). $T(n-1) = 0$: then $S = 0 \ldots 0$ and $T = 11 \ldots 01$, so, by considering S^R and T^R, this case can be dealt with in a similar way to Case (ii).

Case (iv). $f_{11}(S) > 0$: then $S = 0 \ldots 11 \ldots 0$ and $T = 11 \ldots 11$, so take $S' = [0^+(10^+)^*11] \ldots 0$.

Case (v). $S(2) = 1$: then $S = 01 \ldots 0$ and $T = 11 \ldots 11$, so, by considering $\overline{T}$ and $\overline{S}$, this case can be dealt with in a similar way to Case (ii).

Case (vi). $S(n-1) = 1$: then $S = 00 \ldots 10$ and $T = 11 \ldots 11$, so, by considering $\overline{T}^R$ and $\overline{S}^R$, this case can be dealt with in a similar way to Case (ii).

Case (vii). $f_{00}(T) > 0$: then $S = 00 \ldots 00$ and $T = 11 \ldots 00 \ldots 11$, so, by considering $\overline{T}$ and $\overline{S}$, this case can be dealt with in a similar way to Case (iv).

This completes the proof because Cases (i)–(iv) can fail to apply only if S contains more zeros than ones, whereas Cases (i) and (v)–(vii) can fail to apply only if T contains more ones than zeros. □

THEOREM 3.4. *Let S and T be related binary strings of length n. Then*

$$d_r(S,T) \leq \lfloor n/2 \rfloor.$$

Proof. Lemma 3.3 describes a way to increase the combined length of the common prefix and suffix of S and T by at least two using a single reversal. Therefore a sequence of $\lfloor n/2 \rfloor$ such reversals will be enough to transform S into T. □

For example, if $S = 010101010$ and $T = 110000011$, then applying reversals as described in the proof of Lemma 3.3 results in the following sequence of strings:

$$\begin{array}{cccccccccc}
0 & 1 & [\ 0 & 1\ ] & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & [\ 1 & 0 & 1 & 0\ ] \\
0 & 1 & 1 & 0 & 0 & 0 & [\ 1 & 0\ ] & 1 \\
[\ 0 & 1 & 1\ ] & 0 & 0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{array}$$

Note that the first reversal found by the proof of Lemma 3.3 is the one that reverses the first three symbols of T, and so it is the last reversal shown in the illustration.

3.3. Reversal diameter of $\mathcal{B}_n$. The *reversal diameter*, $D_r(n)$, of $\mathcal{B}_n$ is defined to be the maximum value of $d_r(S,T)$ over all related binary strings S and T of length n. More formally

$$D_r(n) = \max\{d_r(S,T) : S,\ T \text{ are related binary strings of length } n\}.$$

LEMMA 3.5. *$\forall k \geq 1$, $d_r(E_k, C_k) = k$ and $d_r(0 \cdot E_k, 0 \cdot C_k) = k$.*

Proof. This follows at once by application of Theorems 3.2 and 3.4 to these strings. □

THEOREM 3.6. *$\forall n \geq 1$, $D_r(n) = \lfloor n/2 \rfloor$.*

Proof. This is an immediate consequence of Theorem 3.4 and Lemma 3.5. □

THEOREM 3.7. *Let S and T be related binary strings of length $2n \geq 6$. Then $d_r(S,T) = n$ if and only if S and T are isomorphic to C_n and E_n.*

Proof. We prove this theorem by induction. The base case is when $n = 3$. Then, by complete search, it may be verified that $d_r(S,T) = 3$ if and only if S and T are isomorphic to E_3 and C_3. Now suppose that the theorem holds when $n \leq k$. Let S and T be strings of length $2k+2$ such that $d_r(S,T) = k+1$. We show that S and T are isomorphic to C_{k+1} and E_{k+1}.

By Lemma 3.3, we can apply a reversal to S or T that increases the combined length of the common prefix and suffix by at least two. Without loss of generality, we can relabel S and T so that the reversal found in the proof of Lemma 3.3 is applied to S resulting in the string S'. Furthermore, we can assume, without loss of generality, that $S(1) = 0$.

It must be that $lcp(S',T) + lcs(S',T) = 2$ and $d_r(S',T) = k$, since any alternative would contradict $d_r(S,T) = k+1$. Let S'_e and T_e be the strings S' and T excluding any common prefix and suffix. By the induction hypothesis, S'_e and T_e must be isomorphic to E_k and C_k. Therefore, since $\overline{E_k} = {E_k}^R$ and $\overline{C_k} = {C_k}^R$, either (a) $S'_e = E_k$ and $T_e = C_k$, or (b) $S'_e = C_k$ and $T_e = E_k$, or (c) $S'_e = {E_k}^R$ and $T_e = {C_k}^R$, or (d) $S'_e = {C_k}^R$ and $T_e = {E_k}^R$.

By the proof of Lemma 3.3 there are essentially three ways that the reversal can be applied to S, as typified by Cases (i), (ii), and (iv) in that proof. We take each

of these three cases in turn and show that for the four possible values of S'_e and T_e, $d_r(S,T) = k+1$ if and only if S and T are isomorphic to E_{k+1} and C_{k+1}.

Case (i). $S = 0\ldots1$, $T = 1\ldots0$. In this case the whole of S is reversed. We show that cases (a) and (b) for S'_e and T_e lead to contradictions, whereas cases (c) and (d) establish the induction step.

(a) $S = 0\cdot {E_k}^R\cdot 1 = 0\cdot 1^k\cdot 0^k\cdot 1$ and $T = 1\cdot C_k\cdot 0 = 1\cdot (10)^k\cdot 0$. Suppose that instead of applying the reversal of Lemma 3.3 we apply the reversal $[011]1^{k-2}\cdot 0^k\cdot 1$ to obtain S''. This reversal extends the common prefix by three characters, so $d_r(S'',T) < k$. Therefore $d_r(S,T) < k+1$, a contradiction.

(b) $S = 0\cdot {C_k}^R\cdot 1 = 0\cdot (01)^k\cdot 1$ and $T = 1\cdot E_k\cdot 0 = 1\cdot 0^k\cdot 1^k\cdot 0$. These strings are isomorphic to the strings in (a), so $d_r(S,T) < k+1$, a contradiction.

(c) $S = 0\cdot E_k\cdot 1 = E_{k+1}$ and $T = 1\cdot {C_k}^R\cdot 0 = C_{k+1}$.

(d) $S = 0\cdot C_k\cdot 1 = {C_{k+1}}^R$ and $T = 1\cdot {E_k}^R\cdot 0 = {E_{k+1}}^R$.

Case (ii). $S = 0\ldots0$ and $T = 10\ldots1$. In this case the reversal results in a string S' that has prefix "10." Therefore S'_e and T_e must be suffixes of S' and T. Now since T ends with a 1, only cases (b) and (c) need to be considered. Both cases lead to a contradiction.

(b) $S' = 10\cdot C_k = 10\cdot (10)^k$ and $T = 10\cdot E_k = 10\cdot 0^k\cdot 1^k$. The reversal on S ends with the first "1" in S, so $S = 01\cdot (10)^k$. Then the reversal $[01101]0\cdot (10)^{k-2}$ produces string S'' that has $d_r(S'',T) < k$ by the induction hypothesis. Therefore $d_r(S,T) < k+1$, a contradiction.

(c) $S' = 10\cdot {E_k}^R = 10\cdot 1^k\cdot 0^k$ and $T = 10\cdot {C_k}^R = 10\cdot (01)^k$. Therefore $S = 01\cdot 1^k\cdot 0^k$, because the reversal on S ends with the first "1." Then the reversal $01\cdot 1^{k-1}[10^k]$ results in a string S'' that has $d_r(S'',T) < k$ by the induction hypothesis. Therefore $d_r(S,T) < k+1$, a contradiction.

Case (iv). $S = 0\ldots11\ldots0$ and $T = 11\ldots11$. In this case the reversal is applied to S to obtain a string S' that has prefix 11. Therefore S'_e and T_e must be suffixes of S' and T. Now, since T ends with "11" only case (b) need be considered. However, in fact, even this case cannot occur.

(b) $S' = 11\cdot C_k = 11\cdot (10)^k$ and $T = 11\cdot E_k = 11\cdot 0^k\cdot 1^k$. However, then the reversal on S could not have moved the first "11" substring in S. Therefore this case cannot occur.

Therefore $d_r(S,T) = k+1$ if and only if S and T are isomorphic to E_{k+1} and C_{k+1}. Therefore, by induction, we have proved the theorem. □

Theorem 3.7 describes the strings of length n that achieve the reversal diameter when n is even. When n is odd, significantly more pairs of strings achieve the reversal diameter.

We note in passing that there appears to be no simple analogue of Lemma 3.3 in the case of alphabet size > 2 and therefore no easy generalization of Theorem 3.6.

3.4. Sorting by reversals. Let S_ι denote the string that is related to S and consists only of a block of zeros, followed by a block of ones. For example, if $S = 01100110$, then $S_\iota = 00001111$. Then determining $d_r(S,S_\iota)$ is an analogue of determining the reversal distance of a permutation. We show that $d_r(S,S_\iota)$ can be determined in polynomial time.

Recall that $z(S)$ denotes the number of blocks of zeros contained in S. Obviously, $z(S_\iota) = 1$ (unless S does not contain any zeros). The following lemma can be verified easily.

LEMMA 3.8. *Let S' be a string obtained from S by a single reversal. Then*

$$z(S') \geq z(S) - 1.$$

With this lemma we can determine $d_r(S, S_i)$.

THEOREM 3.9. *For any binary string S,*

$$d_r(S, S_i) = \begin{cases} z(S) - 1 & \text{if } S(1) = 0, \\ z(S) & \text{otherwise.} \end{cases}$$

Proof. By Lemma 3.8, $d_r(S, S_i) \geq z(S) - 1$. If $S(1) = 0$, then $z(S) - 1$ reversals of the form $0^+[1^+0^+]1\ldots$ or $0^+[1^+0^+]$ transform S into S_i. If $S(1) = 1$, then an extra reversal is required because it is impossible to change the first symbol to a 0 and also reduce the value of z. This bound can be achieved by performing a reversal $[1\ldots0]1\ldots$ or $[1\ldots0]$ before performing $z(S) - 1$ reversals as described for the case $S(1) = 0$. □

The distance described in Theorem 3.9 can be calculated easily in polynomial time. In section 5 it is shown that, in general, determining $d_r(S, T)$ is NP-hard.

4. Transposition distance between binary strings. In this section, we present an upper bound and a lower bound for transposition distance between binary strings. These bounds are used to determine the transposition diameter, and identify some strings that achieve the diameter. A restricted version of the problem, that is analogous to the problem of sorting by transpositions, is shown to be solvable in polynomial time.

4.1. A lower bound. Again, it is the appropriate concept of a breakpoint that is used to obtain a lower bound for transposition distance.

Transposition breakpoints are defined in a similar way to reversal breakpoints. For example, if S contains more "11" substrings than T, then each extra "11" substring contributes a breakpoint. However, a crucial difference between reversal breakpoints and transposition breakpoints is that "01" and "10" substrings are counted separately, since a transposition cannot transform one to the other. As before, we prepend α and append ω to each string.

The number of *transposition breakpoints* is therefore

$$b_t(S, T) = \sum_{a,b \in A} \delta(f_{ab}(S) - f_{ab}(T)),$$

where $A = \{\alpha, 0, 1, \omega\}$ and, as before,

$$\delta(x) = x \text{ if } x > 0 \text{ and } 0 \text{ otherwise.}$$

Clearly, if $S = T$, then $b_t(S, T) = 0$. However, it is possible that $b_t(S, T) = 0$, even when $S \neq T$, for example, if $S = 101001$ and $T = 100101$.

LEMMA 4.1. *Suppose that S' is obtained from S by a single transposition. Then*

$$b_t(S', T) \geq \begin{cases} b_t(S, T) - 3 & \text{if } S(1) \neq S'(1) \text{ and } S(n) \neq S'(n), \\ b_t(S, T) - 2 & \text{otherwise.} \end{cases}$$

Proof. The transposition must have the form $\ldots a[b\ldots c][d\ldots e]f\ldots$, where $a \in \{\alpha, 0, 1\}$, $b, c, d, e \in \{0, 1\}$, and $f \in \{0, 1, \omega\}$. The transposition results in the string $\ldots a[d\ldots e][b\ldots c]f\ldots$. Now let us suppose that the none of the substrings "ab," "cd,"

or "ef" is the same as any of the substrings "ad," "eb," or "cf." Then "cd"≠"cf," so $d \neq f$. Similarly, $c \neq a$, $b \neq f$, and $e \neq a$. Now suppose that $a \neq \alpha$. Then $c = e$. However, then "ef"="cf", a contradiction. Similarly if $f \neq \omega$, then "ab"="ad." Therefore, if $a \neq \alpha$ or $f \neq \omega$, then at least one of the substrings "ab," "cd," or "ef" is the same as one of the substrings "ad," "eb," or "cf."

If a transposition moves the first and last symbol of S, then at most three substrings of length two may change as a result of the transposition. Given the definition of $b_t(S', T)$, this means that $b_t(S', T) \geq b_t(S, T) - 3$. However, if a transposition does not move the first and last symbols of S, then at most two substrings of length two may change as a result of the transposition. In such cases $b_t(S', T) \geq b_t(S, T) - 2$. □

THEOREM 4.2. *Let S and T be related binary strings of length n. Then*

$$d_t(S,T) \geq \begin{cases} \lceil b_t(S,T)/2 \rceil & \text{if } S(1) = T(1), \text{ or } S(n) = T(n), \\ \lceil (b_t(S,T) - 1)/2 \rceil & \text{otherwise.} \end{cases}$$

Proof. A sequence of transpositions that transforms S into T can contain at most one transposition that reduces the number of breakpoints by 3. Such a transposition is only possible if $S(1) \neq T(1)$ and $S(n) \neq T(n)$. Every other transposition can reduce the number of breakpoints by at most 2. The theorem follows easily from these observations. □

As before, it is easy to find examples for which this lower bound is not exact; for example, if $S = 011100110001$ and $T = 100011001110$, then the bound is 1, but $d_t(S, T) = 2$.

Again, the definition of a transposition breakpoint for binary strings together with the bound of Theorem 4.2 can be directly extended to strings over larger alphabets. For alphabets of size > 2, however, each transposition can reduce the number of breakpoints by 3. Hence the extended version of Theorem 4.2 is as follows.

THEOREM 4.3. *Let S and T be related strings, of length n, over an alphabet of size > 2. Then*

$$d_t(S,T) \geq \lceil b_t(S,T)/3 \rceil.$$

4.2. An upper bound. In this section a simple upper bound is derived for transposition distance.

LEMMA 4.4. *Let S and T be related strings of length n such that $S \neq T$. Then it is possible either*

(a) *to apply a transposition to S resulting in a string S' such that $lcp(S', T) + lcs(S', T) \geq lcp(S, T) + lcs(S, T) + 2$ or*

(b) *to apply a transposition to T resulting in a string T' such that $lcp(S, T') + lcs(S, T') \geq lcp(S, T) + lcs(S, T) + 2$.*

Proof. Without loss of generality, it can be assumed that $S(1) = 0$, $S(1) \neq T(1)$, and $S(n) \neq T(n)$. The transposition that is applied to S or T depends on S and T. Three cases are described, and for each case a transposition is shown with the required property. (Again, each case is considered only if S and T fail to meet the conditions of any earlier case, making the cases mutually exclusive.)

Case (i). $S(n) = 1$: then $S = 0 \ldots 1$ and $T = 1 \ldots 0$, so take $S' = [0^+][1 \ldots]$.

Case (ii). $f_{11}(S) > 0$: then $S = 0 \ldots 11 \ldots 0$ and $T = 1 \ldots 1$, so take $S' = [0^+(10^+)^*1][1 \ldots 0]$.

Case (iii). $f_{11}(S) = 0$ and $f_{00}(T) > 0$: then $S = 0 \ldots 0$ and $T = 1 \ldots 00 \ldots 1$, so, by considering $\overline{T}$ and $\overline{S}$, this case is similar to Case (ii).

This completes the proof because Cases (i) and (ii) can fail to apply only if S contains more zeros than ones, whereas Cases (i) and (iii) can fail to apply only if T contains more ones than zeros. □

THEOREM 4.5. *Let S and T be related binary strings of length n. Then*

$$d_t(S,T) \leq \lfloor n/2 \rfloor.$$

Proof. Lemma 4.4 describes a way to increase the combined length of the common prefix and suffix of S and T by at least two using a single transposition. Therefore a sequence of $\lfloor n/2 \rfloor$ such transpositions will be enough to transform S into T. □

4.3. Transposition diameter of $\mathcal{B}_n$. The *transposition diameter*, $D_t(n)$, of $\mathcal{B}_n$ is the maximum value of $d_t(S,T)$ taken over all related binary strings of length n. More formally

$$D_t(n) = \max\{d_t(S,T) : \text{S, T are related binary strings of length } n\}.$$

LEMMA 4.6. *$\forall k \geq 1$, $d_t(E_k, C_k) = k$ and $d_t(0 \cdot E_k, 0 \cdot C_k) = k$.*

Proof. Both cases follow at once by application of Theorems 4.2 and 4.5. □

THEOREM 4.7. *$\forall n \geq 1$, $D_t(n) = \lfloor n/2 \rfloor$.*

Proof. This is an immediate consequence of Theorem 4.5 and Lemma 4.6. □

THEOREM 4.8. *Let S and T be related binary strings of length $2n \geq 4$. Then $d_t(S,T) = n$ if and only if S and T are isomorphic to C_n and E_n.*

Proof. We prove this theorem by induction. The base case is when $n = 2$. Then, by complete search, it may be verified that $d_t(S,T) = 2$ if and only if S and T are isomorphic to E_2 and C_2. Now suppose that the theorem holds when $n \leq k$. Let S and T be strings of length $2k+2$ such that $d_t(S,T) = k+1$. We show that S and T are isomorphic to C_{k+1} and E_{k+1}.

By Lemma 4.4 we can apply a transposition to S or T that increases the combined length of the common prefix and suffix by at least two. Without loss of generality, we can relabel S and T so that the transposition found in the proof of Lemma 4.4 is applied to S resulting in the string S'. Furthermore, we can assume, without loss of generality, that $S(1) = 0$.

It must be that $lcp(S',T) + lcs(S',T) = 2$ and $d_t(S',T) = k$, since any alternative would contradict $d_t(S,T) = k+1$. Let S'_e and T_e be the strings S' and T excluding any common prefix and suffix. By the induction hypothesis, S'_e and T_e must be isomorphic to E_k and C_k. Therefore, either (a) $S'_e = E_k$ and $T_e = C_k$, or (b) $S'_e = C_k$ and $T_e = E_k$, or (c) $S'_e = {E_k}^R$ and $T_e = {C_k}^R$, or (d) $S'_e = {C_k}^R$ and $T_e = {E_k}^R$.

By the proof of Lemma 4.4, there are essentially two ways that the transposition can be applied to S, as typified by Cases (i) and (ii) in that proof. We take each case in turn and show that for the four possible values of S'_e and T_e, $d_t(S,T) = k+1$ if and only if S and T are isomorphic to E_{k+1} and C_{k+1}.

Case (i). $S = 0 \ldots 1$, $T = 1 \ldots 0$. In this case the transposition moves the block of zeros at the front of S to the end. We show that cases (c) and (d) for S'_e and T_e establish the induction step, whereas cases (a) and (b) lead to contradictions.

(a) $S' = 1 \cdot E_k \cdot 0 = 1 \cdot 0^k \cdot 1^k \cdot 0$ and $T = 1 \cdot C_k \cdot 0 = 1 \cdot (10)^k \cdot 0$. Therefore $S = 01 \cdot 0^k \cdot 1^k$. However, then the transposition $0[100][0^{k-2} \cdot 1^k]$ produces a string S'' such that $d_t(S'',T) < k$ by the induction hypothesis. Therefore $d_t(S,T) < k+1$, giving a contradiction.

(b) $S' = 1 \cdot C_k \cdot 0 = 1 \cdot (10)^k \cdot 0$ and $T = 1 \cdot E_k \cdot 0 = 1 \cdot 0^k \cdot 1^k \cdot 0$. Then $S = 001 \cdot (10)^{k-1} \cdot 1$. However, then the transposition $[001 \cdot (10)^{k-1}][1]$ produces a string

S'' such that $d_t(S'', T) < k$ by the induction hypothesis. Therefore $d_t(S, T) < k + 1$, giving a contradiction.

(c) $S' = 1 \cdot {E_k}^R \cdot 0 = {E_{k+1}}^R$ and $T = 1 \cdot {C_k}^R \cdot 0 = C_{k+1}$. Therefore $S = E_{k+1}$.

(d) $S' = 1 \cdot {C_k}^R \cdot 0 = C_{k+1}$ and $T = 1 \cdot {E_k}^R \cdot 0 = {E_{k+1}}^R$. Therefore $S = {C_{k+1}}^R$.

Case (ii). $S = 0 \ldots 11 \ldots 0$, $T = 1 \ldots 1$. In this case the transposition applied to S to obtain S' is of the form $[0^+(10^+)^*1][1 \ldots 0]$. Note that this transposition splits the first occurrence of "11" in S. Note also that S' must contain "00" and end with "01," so only Case (c) needs to be considered. However, this cases leads to contradiction.

(c) $S' = 1 \cdot {E_k}^R \cdot 1 = 1 \cdot 1^k \cdot 0^k \cdot 1$ and $T = 1 \cdot {C_k}^R \cdot 1 = 1 \cdot (01)^k \cdot 1$. Then $S = 0^+ \cdot 1^{k+2} \cdot 0^+$. However, the transposition $0^+ \cdot 1^k[11][0^+]$ produces a string S'' such that $d_t(S'', T) < k$ by the induction hypothesis. Therefore $d_t(S, T) < k + 1$, giving a contradiction.

Therefore $d_t(S, T) = k + 1$ if and only if S and T are isomorphic to E_{k+1} and C_{k+1}. Therefore by induction the theorem is true. □

Again, we note that there appears to be no direct analogue of Lemma 4.4 when the alphabet size is > 2 and therefore no easy generalization of Theorem 4.7.

4.4. Sorting by transpositions. We show that $d_t(S, S_\iota)$ can be determined in polynomial time. The following lemma can be verified easily.

LEMMA 4.9. *Let S' be a string obtained from S by a single transposition. Then*

$$z(S') \geq z(S) - 1.$$

With this lemma we can determine $d_t(S, S_\iota)$.

THEOREM 4.10. *For any binary string S,*

$$d_t(S, S_\iota) = \begin{cases} z(S) - 1 & \text{if } S(1) = 0, \\ z(S) & \text{otherwise.} \end{cases}$$

Proof. By Lemma 4.9, $d_t(S, S_\iota) \geq z(S) - 1$. If $S(1) = 0$, then $z(S) - 1$ transpositions of the form $0^+[1^+][0^+]1 \ldots$ or $0^+[1^+][0^+]$ transform S into S_ι. If $S(1) = 1$, an extra transposition is required, because it is impossible to change the letter at the front of the string to 0 with a transposition and also reduce the value of z. The bound in this case can be achieved by performing the transposition $[1^+][0^+]1 \ldots$ or $[1^+][0^+]$, followed by the sequence of transpositions used when $S(1) = 0$. □

The distance described in Theorem 4.10 can be calculated easily in polynomial time. The question of whether, in general, the transposition distance between any two strings can be calculated in polynomial time remains open.

5. NP-completeness of reversal distance. In this section, we prove that the general problem of finding the reversal distance between two related strings is NP-hard, even if the strings are drawn from a binary alphabet. We begin with a definition of the reversal distance problem as a decision problem (RD):

RD

Instance: Related strings S and T of length n, over an alphabet of size $t \geq 2$, and a bound $d \in Z^+$.

Question: Is $d_r(S, T) \leq d$?

The proof consists of a pseudopolynomial transformation from 3-Partition to RD. The definition of 3-Partition is as follows:

3-*Partition*

Instance: A set A of $3m$ elements, a bound $B \in Z^+$, and a size

$s(a) \in Z^+$ for each $a \in A$ such that $B/4 < s(a) < B/2$ and such that $\sum_{a \in A} s(a) = mB$.
Question: Can A be partitioned into m disjoint sets $A_1, A_2, \ldots, A_m$, such that, for $1 \leq i \leq m$, $\sum_{a \in A_i} s(a) = B$? (Note that each A_i must contain exactly three elements from A.)

Garey and Johnson [9] (see also Chapter 4.2 of [10]) have proved that 3-Partition is NP-complete (in the strong sense). Therefore a pseudopolynomial transformation from 3-Partition to RD is enough to prove that RD is NP-complete.

THEOREM 5.1. *RD is NP-complete, even when $t = 2$.*

Proof. RD is in NP because, given a sequence of reversals, it can easily be checked in polynomial time that the sequence transforms S into T and has length at most d. We now describe the pseudopolynomial transformation from 3-Partition to RD.

Let I be an instance of 3-Partition. From this instance construct an instance, I', of RD with $S = (\sum_{i=1}^{3m-1} 0^{s(a_i)}1^3) \cdot 0^{s(a_{3m})}$, $T = (0^B1)^m \cdot 1^{8m-3}$, and $d = 3m - 1$. Therefore the blocks of zeros in S represent elements of A and the lengths of the blocks represent the sizes of the elements.

We first show that $d_r(S, T) \geq 3m - 1$.

Let U be a string that is related to S and T. Recall that $z(U)$ is the number of blocks of zeros in the string U. Define $o(U)$ as the number of blocks of ones of length one in U. Then the function $f(U) = z(U) - o(U) - 1$ can be viewed as a kind of distance function between strings U and T, since $f(T) = 0$. Furthermore, $f(S) = 3m - 1$. We show that, if U' is obtained from U by applying a single reversal, then $f(U') \geq f(U) - 1$.

Suppose that ρ is a reversal that transforms U into U' with $f(U') < f(U)$. Then ρ must reduce the number of blocks of zeros or increase the number of blocks of ones of length one.

If ρ reduces the number blocks of zeros, then it must have the form $\ldots 1[0 \ldots 1]0 \ldots$ or $\ldots 0[1 \ldots 0]1 \ldots$, and therefore $f(U) - f(U') = 1$. Therefore it is impossible for ρ to increase the number of blocks of ones of length one as well as reduce the number of blocks of zeros.

If ρ increases the number of blocks of ones of length one, then it must be of the form $\ldots 01[10 \ldots 0]0 \ldots$, or $\ldots 1[10 \ldots 11]0 \ldots$, or $\ldots 01[1 \ldots 0]11 \ldots$, or the mirror image of one of these three reversals. (Note the reversals $\ldots 01[11 \ldots 0]0 \ldots$ and $\ldots 11[10 \ldots 0]0 \ldots$ do not reduce the value of f.) In each case $f(U) - f(U') = 1$. Therefore $d_r(S, T) \geq 3m - 1$.

Note that the first reversal in the previous paragraph is special because it increases the number of blocks of length one by two but also increases the number of blocks of zeros. We call this kind of reversal a *bad reversal*.

We now show that the given transformation from 3-Partition to an instance of RD is a pseudopolynomial transformation. To do so we have to verify four standard properties of a pseudopolynomial transformation [10, p. 101]. We verify these four properties in turn.

For property (a), we have to show that I is a yes instance of 3-Partition if and only if $d_r(S, T) \leq 3m - 1$.

We have already shown that $d_r(S, T) \geq 3m - 1$. We now show that if $d_r(S, T) = 3m-1$, then no minimal length sequence of reversals that transforms S into T contains a bad reversal.

Suppose that $d_r(S, T) = 3m-1$. Every reversal in a minimal length sequence that transforms S into T must reduce the value of f by one. For a reversal to be bad the

string must contain 0110 as a substring. However, S contains no such substring, and no reversal that reduces the value of f by one can create such a substring. Therefore if $d_r(S,T) = 3m-1$, then no minimal length sequence of reversals that transforms S into T contains a bad reversal.

This means that if $d_r(S,T) = 3m-1$, each block of zeros in T is constructed from three blocks of zeros in S. It follows that I is a yes instance of 3-Partition if $d_r(S,T) = 3m-1$.

Now we show that if I is a yes instance of 3-Partition, then $d_r(S,T) \leq 3m-1$. Since I is a yes instance, we can partition A into m disjoint sets $A_1, \ldots, A_m$, each of which contains three elements and sums to B. For each subset A_i in turn, we can use two reversals of the form $\ldots 0[1\ldots 0]a\ldots$, where $a \in \{1, \omega\}$ (where ω is the special character used to denote the end of the string) to merge the three blocks of zeros representing the elements of A_i into a single block of zeros of length B without affecting any other block of zeros. (Note that the reversals shown do not move the block of zeros at the front of the string.) Then we can use $m-1$ reversals of the form $\ldots 01[11\ldots 0]1\ldots$ to create blocks of ones of length one separating the blocks of zeros. This sequence of reversals has length $3m-1$, so $d_r(S,T) \leq 3m-1$. This establishes the required property (a).

To prove properties (b), (c), and (d) we need Length and Max functions for 3-Partition and RD. For 3-Partition, reasonable definitions are $\text{Length}(I) = |A| + \sum_{a\in A}\lceil \log_2 s(a)\rceil$ and $\text{Max}(I) = \max\{s(a) : a \in A\}$. For RD, reasonable definitions are $\text{Length}'(I') = 2n + \lceil \log_2 d\rceil$ and $\text{Max}'(I') = 1$ (since RD is not a number problem). Note that $n = \sum_{a\in A} s(a) + |A| - 3$. Given these functions, the required properties can be proved quite easily.

Therefore the transformation is a pseudopolynomial transformation and therefore RD is NP-complete. □

The transformation just described was obtained after several other simpler transformations had been shown to fail. An example is a potential transformation from sorting by reversals to *RD*. Given a permutation π, define strings $S = (\sum_{i=1}^{n-1} 0^{\pi(i)}1)\cdot 0^{\pi(n)}$ and $T = (\sum_{i=1}^{n-1} 0^i 1)\cdot 0^n$. One might conjecture that determining the value of $d_r(S,T)$ must also determine the value of $d_r(\pi)$. However, if $\pi = 3142$, then $d_r(\pi) = 3$, but $S = 0001010000100$, $T = 0100100010000$, and $d_r(S,T) = 2$. Therefore this transformation does not work.

6. Conclusion. In this paper we have shown that, just as sorting permutations by reversals is NP-hard, so also is finding the reversal distance between two strings, even when the strings are drawn from a binary alphabet. We have derived lower and upper bounds for the reversal distance between binary strings and used these to find the reversal diameter of $\mathcal{B}_n$.

The complexity of finding the transposition distance between two strings remains open, just as the complexity of sorting permutations by transpositions is open. We have also derived lower and upper bounds for the transposition distance between binary strings and used these to find the transposition diameter of $\mathcal{B}_n$. This contrasts with the problem of transposition diameter for permutations, which is unresolved.

In [6], Christie introduces the problem of sorting by block-interchanges. A block-interchange is similar to a transposition, except that the substrings that are swapped need not be adjacent. Christie proved that this problem could be solved in polynomial time. When extended to strings, however, it can be shown [8] in a manner similar to that used in the proof of Theorem 5.1 that the block-interchange distance problem is NP-hard, even when the strings are drawn from a binary alphabet.

REFERENCES

[1] V. Bafna and P. A. Pevzner, *Genome rearrangements and sorting by reversals*, SIAM J. Comput., 25 (1996), pp. 272–289.

[2] V. Bafna and P. A. Pevzner, *Sorting by transpositions*, SIAM J. Discrete Math., 11 (1998), pp. 224–240.

[3] P. Berman and S. Hannenhalli, *Fast sorting by reversals*, in Combinatorial Pattern Matching, Lecture Notes in Comput. Sci. 1075, Springer, Berlin, 1996, pp. 168–185.

[4] A. Caprara, *Sorting by reversals is difficult*, in Proceedings of the First International Conference on Computational Molecular Biology (RECOMB'97), ACM Press, Santa Fe, NM, 1997, pp. 75–83.

[5] A. Caprara, *Sorting permutations by reversals and Eulerian cycle decompositions*, SIAM J. Discrete Math., 12 (1999), pp. 91–110.

[6] D. A. Christie, *Sorting permutations by block-interchanges*, Inform. Process. Lett., 60 (1996), pp. 165–169.

[7] D. A. Christie, *A 3/2-approximation algorithm for sorting by reversals*, in Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, 1998, pp. 244–252.

[8] D. A. Christie, *Genome Rearrangement Problems*, Ph.D. thesis, Department of Computing Science, University of Glasgow, Glasgow, Scotland, 1998.

[9] M. R. Garey and D. S. Johnson, *Complexity results for multiprocessor scheduling under resource constraints*, SIAM J. Comput., 4 (1975), pp. 397–411.

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

[11] S. Hannenhalli and P. A. Pevzner, *Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals)*, in Proceedings of the 27th Annual ACM Symposium on Theory of Computing, Las Vegas, 1995, pp. 178–189.

[12] S. Hannenhalli and P. A. Pevzner, *Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals*, J. ACM, 46 (1999), pp. 1–27.

[13] H. Kaplan, R. Shamir, and R. E. Tarjan, *Faster and simpler algorithm for sorting signed permutations by reversals*, in Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, 1997, pp. 344–351.

[14] J. Kececioglu and D. Sankoff, *Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement*, Algorithmica, 13 (1995), pp. 180–210.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 207–222

THE WAKEUP PROBLEM IN SYNCHRONOUS BROADCAST SYSTEMS*

LESZEK GĄSIENIEC†, ANDRZEJ PELC‡, AND DAVID PELEG§

Abstract. This paper studies the differences between two levels of synchronization in a distributed broadcast system (or a multiple-access channel). In the *globally synchronous* model, all processors have access to a global clock. In the *locally synchronous* model, processors have local clocks ticking at the same rate, but each clock starts individually when the processor wakes up.

We consider the fundamental problem of waking up all n processors of a completely connected broadcast system. Some processors wake up spontaneously, while others have to be woken up. Only awake processors can send messages; a sleeping processor is woken up upon hearing a message. The processors hear a message in a given round if and only if exactly one processor sends a message in that round. Our goal is to wake up all processors as fast as possible in the worst case, assuming an adversary controls which processors wake up and when. We analyze the problem in both the globally synchronous and locally synchronous models with or without the assumption that n is known to the processors. We propose randomized and deterministic algorithms for the problem, as well as lower bounds in some of the cases. These bounds establish a gap between the globally synchronous and locally synchronous models.

Key words. broadcast, clock, synchronous, wakeup

AMS subject classifications. 68W15, 68W20

PII. S0895480100376022

1. Introduction.

1.1. The problem. This paper focuses on the effects of the level of synchronization required in *broadcast* systems (or *multiple-access channels*) such as the Ethernet. The communication system is assumed to be synchronous, namely, processors send messages in rounds. As the communication channel is shared by all processors, messages might collide. It is assumed that the processors succeed in hearing a message in a given round if and only if exactly one processor sends a message in that round; if more than one processor, or none of them, sends a message in that round, then nobody hears anything. Hence the communication model is equivalent to the radio model; cf. [1, 2, 5, 6, 7, 8, 9, 10, 14, 18, 19, 20, 22] in a complete graph without collision detection. As pointed out in [3], which studied the relationships between radio networks with and without collision detection, the absence of collision detection characterizes noisy networks since the noise does not allow processors to distinguish no transmission from multitransmission.

*Received by the editors August 1, 2000; accepted for publication (in revised form) January 19, 2001; published electronically April 3, 2001. A preliminary version of this paper appeared in *Proceedings of the* 19*th Annual ACM Symposium on Principles of Distributed Computing*, Portland, OR, 2000, pp. 113–121.

http://www.siam.org/journals/sidma/14-2/37602.html

†Department of Computer Science, The University of Liverpool, Liverpool L69 7ZF, United Kingdom (leszek@csc.liv.ac.uk).

‡Département d'Informatique, Université du Québec à Hull, Hull, Québec J8X 3X7, Canada (Andrzej_Pelc@uqah.uquebec.ca). The research of this author was supported in part by NSERC grant OGP 0008136. This research was done during the author's stay at the University of Liverpool.

§Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehovot 76100, Israel (peleg@wisdom.weizmann.ac.il). The research of this author was supported in part by grants from the Israel Ministry of Science and Art.

Evidently, the possibility of collisions makes broadcast systems harder to coordinate and control than standard point-to-point message passing systems, and performing even simple tasks poses serious difficulties. A central problem in such systems is therefore to establish a pattern of access to the shared communication media that will allow messages to go through with as little interruption as possible, i.e., avoid (or efficiently resolve) message collisions.

This problem is somewhat easier if the processors are required to be constantly attentive to the communication channel. This enables the system to use, for instance, round-robin based access protocols, as well as other schemes based on the processors being fully synchronized at all times. However, it is often desirable to allow a processor to stop being "alert" on the communication channel, whenever there is no traffic currently being transmitted on the channel, and the processor itself does not wish to send a message. From the point of view of the communication system, such a processor switches from "alert" to "sleeping," until such time as its participation is required again. Clearly, allowing processors to "sleep" entails a certain loss of synchronization, which must be regained when the processors become alert again and wish to communicate. This loss of synchronization and its effects are at the focus of the current study.

Specifically, in this paper we consider the fundamental problem of waking up all of n processors, numbered $1, \ldots, n$, in a completely connected broadcast system. Some processors wake up spontaneously, in different rounds, while the others have to be woken up. Only awake processors can send messages. A sleeping processor wakes up upon hearing a message. This will happen on the first "successful" round, namely, the first round when exactly one processor sends a message.

We consider two settings of measuring time. In the first setting, termed the *globally synchronous* model, all processors have access to a global clock showing the current round number. The global clock is always available, and when a processor wakes up it can immediately see the current round number. The clock thus counts round numbers globally for all processors, starting in round 1. In the second setting, termed the *locally synchronous* model, each processor has a local clock. All local clocks tick at the same rate, one tick per round. However, no global count is available, and the local clock of processor i starts counting rounds in the round in which processor i wakes up. Moreover, in each of the above settings, we distinguish the situation when the size n of the system is known to all processors and the situation when n is unknown.

Our goal is to construct algorithms for waking up all processors as fast as possible via a multiple-access channel. We focus on the worst case, when an adversary controls which processors wake up spontaneously and when. The time complexity of the wakeup process is measured by the number of rounds elapsing from the time the first processor wakes up (spontaneously) and the time all processors are woken up (i.e., the first successful round).

During the execution, each of the processors is *active* (sends a message) in some rounds and *idle* in the others. The rounds in which a processor i is active are decided by a local protocol Π_i. The protocol Π_i can be thought of as generating a binary *activation sequence* α_i, designating the *activation times* of processor i. Specifically, if the sequence contains 1 in its tth position, i.e., $\alpha_i[t] = 1$, then processor i is required to be active on the tth round after it wakes up; conversely, $\alpha_i[t] = 0$ means i must remain silent.

The protocols Π_i used by the processors may assume any one of a number of

forms. In the current paper we distinguish between the following types of algorithms. The simplest and most rigid type of protocol is what is hereafter referred to as a *fixed schedule*. It is specified by a *predefined* (and sufficiently long) activation sequence α_i for each processor i. The sequence α_i is constructed for each processor i in advance by a preprocessing algorithm Π^{pre} and is stored at its local memory. Once the processor i wakes up, it starts following its activation sequence α_i without deviation, regardless of any other parameters or inputs it may have. We refer to the set of sequences $A = \{\alpha_1, \ldots, \alpha_n\}$ of the fixed schedule as the *activation set* of the system. Observe that using a fixed schedule makes the adversary rather powerful (in a manner of speaking), as it can use this knowledge in order to decide on its waking strategy.

Alternatively, the protocol Π_i may be an *online* distributed protocol, which is invoked locally by processor i upon waking up and starts generating the activation sequence α_i online. Such a protocol may be either deterministic or randomized.

Note that in the locally synchronous model there is no difference between a fixed schedule and a *deterministic* online algorithm; in both, the activation sequence α_i used by processor i is unique in all executions. This is no longer the case in the globally synchronous model, where in different executions, processor i may wake up on different rounds and may use the global round number as input to the algorithm Π_i, thus generating different sequences. Nevertheless, it is clear that even in the globally synchronous model, the adversary has complete knowledge of the activation sequences, as it controls the spontaneous wakeup time of the processors. The situation is radically different once we consider randomized protocols, as in this setting the adversary is prevented from knowing the activation sequences in advance.

1.2. Related work. Multiple-access channels, including systems such as the Aloha multiaccess system, the local area Ethernet network, multitapped buses, satellite communication systems, and packet radio networks, have been studied extensively in the literature (see [4, 23] and the references therein). Some of these models (in particular the Ethernet) assume an intermediate model of collision detection, which is not discussed here, in which the transmitting processor detects the fact that its message has collided. This feature naturally simplifies the wakeup problem.

Collision detection and resolution, as well as access management algorithms for multiple-access channels, were studied mainly in the queueing theory model, i.e., assuming a probability distribution on the arrival rate of messages at the different processors; cf. [4, 17, 16, 15]. Also, the wakeup problem was not considered as such in these contexts, although the complications that arise are similar. (The wakeup problem *has* been studied in a number of other contexts within the area of distributed systems; see, for example, [11, 13, 21], but the issues and techniques are naturally different, given the radically different communication model.)

The broadcast operation in multihop radio networks was studied in [1, 2, 5, 6, 9, 10]. The model used in those papers is based on one of two assumptions, namely, either there is a single *source* initiating the process, or all processors wake up spontaneously at time 0. Hence the starting point for the broadcast problem assumes that the wakeup problem, dealt with here, has already been solved. Nevertheless, there are strong links between the models.

Our model is closely related to certain restricted forms of concurrent write PRAM models. See, for example, [12].

To the best of our knowledge, the current paper is the first attempt to provide worst-case time bounds (against an adversary) for the wakeup problem in the synchronous setting.

1.3. Our results. We begin by showing that in the globally synchronous model, where all processors have access to a global clock, optimal deterministic wakeup time is exactly n in the worst case if the size n is known to all processors. We also show a randomized online algorithm that operates in time $O(\log n \cdot \log(1/\varepsilon))$ and succeeds in waking up the system with probability at least $1-\varepsilon$ under the same assumptions. In the case of unknown n we construct a deterministic wakeup algorithm working in worst-case time $4n$.

Under the locally synchronous model with known n, we construct a randomized online algorithm that operates in time $O(n \log(1/\varepsilon))$ and succeeds in waking up the system with probability at least $1-\varepsilon$. On the other hand, we construct a $O(n^2 \log n)$ deterministic wakeup algorithm. We also show that even when n is known, every deterministic wakeup algorithm requires worst-case time at least $(1+\varepsilon)n$ for some $\varepsilon > 0$. This establishes a gap in efficiency between the locally and globally synchronous models. Finally, still in the locally synchronous model with known n, we prove (nonconstructively) the existence of a fixed schedule with wakeup time $O(n \log^2 n)$.

Under the weakest assumptions, namely, in the locally synchronous model without the knowledge of n, we present two wakeup algorithms. The first is a randomized online algorithm which succeeds in waking up the system in time $O(n^2 \log(1/\varepsilon))$ with probability at least $1-\varepsilon$; the second is a deterministic wakeup algorithm working in time $O(n^4 \log^5 n)$.

2. The globally synchronous model. In this section we consider the globally synchronous model, where a global clock is available to all processors, and every round has a global number which is known to all currently awake processors.

2.1. Known system size. We first consider the case when the number of processors, n, is known to all of them. In this simplest case we have tight upper and lower bounds on the time required for waking up the system deterministically.

THEOREM 2.1. *If a global clock is available, the processors are labeled* $\{1, \ldots, n\}$ *and the number* n *of processors is known to all of them, then there exists a deterministic online algorithm for waking up the system in time* n *in the worst case.*

Proof. Every processor sends a message only once after waking up in the earliest round whose number (modulo n) is equal to its label minus 1. Thus in each round at most one processor sends a message. Clearly every processor sends a message at most n rounds after waking up. □

THEOREM 2.2. *The worst-case minimum time to wake up an* n*-processor system by either a deterministic online algorithm or a fixed schedule is at least* n*, even if a global clock is available and the number* n *of processors is known to all of them.*

Proof. In order to prove the lower bound, consider an algorithm that guarantees wakeup time $k < n$ in the worst case. We show that this leads to a contradiction by presenting an adversary that wakes up a certain nonempty subset of processors in round 1 and prevents any processor from being the only sender of a message in any round until round k.

The adversary constructs a sequence $R_0, \ldots, R_k$ of sets of integers as follows. Let $R_0 = \{1, \ldots, n\}$. Suppose that R_j is already constructed. Let S_j be the set of those integers $i \in R_j$ for which there exists a round $r \leq k$ such that i is the only processor in R_j with the following property: it sends a message in round r if it wakes up in round 1. Let $R_{j+1} = R_j \setminus S_j$.

It follows from the construction that the union of all sets S_j, $j < k$, has size at most k. Since $k < n$, it follows that R_k is nonempty. The adversary wakes up all

processors from the set R_k in round 1. By definition of R_k, in any round $r \in \{1, \ldots, k\}$ either none or at least two processors send messages. □

We next show that the problem can be solved by a polylogarithmic time *randomized* online algorithm.

Algorithm Repeated-Decay. The algorithm makes use of a variant of the procedure *Decay* presented in [2] for performing broadcast. The procedure assumes that all processors wake up at time 0, and some $1 \leq d \leq n$ processors contend on finding a free time slot and broadcasting their message. Each of the d contending processors repeatedly performs the following, up to a maximum of $k = 2\lceil \log n \rceil$ rounds. In each round it broadcasts a wakeup message and then flips a fair coin. If the outcome of the coinflip is 0, then it quits the procedure.

To adapt the procedure to our setting, we partition the time axis into consecutive blocks of k rounds each and repeatedly execute procedure Decay in each block of rounds. A processor waking up spontaneously on some round t must remain silent until the end of the current block and may start participating in procedure Decay only at the beginning of the next block. Hence all currently awake processors execute procedure Decay in an aligned manner.

THEOREM 2.3. *If a global clock is available and the number n of processors is known to all of them, then the randomized algorithm Repeated-*Decay *succeeds in waking up the system in time $O(\log n \cdot \log(1/\varepsilon))$ with probability at least $1 - \varepsilon$.*

Proof. It is shown in [2] that in a single invocation of procedure Decay, with probability at least 1/2, there will be a successful round in which exactly one of the contending processors will broadcast. (Intuitively, the reason can be thought of as follows. Roughly half of the contenders quit after each round. Therefore there will likely be one or two final rounds in the phase, roughly after $\log d$ rounds, in which the number of participating contenders is small, say, one or two. On those rounds, there's a good chance of success.)

Consequently, the probability that k repeated invocations of the procedure will fail to wake up the system is at most $1/2^k$. It follows that within $\log(1/\varepsilon)$ time blocks from the time the first processor woke up spontaneously, the system will be woken up with probability at least $1 - 1/2^{\log(1/\varepsilon)} = 1 - \varepsilon$. □

2.2. Unknown system size. If the number of processors is not known to any of them but the global clock is available, it is still possible to wake up the system by a deterministic online algorithm in linear time.

Algorithm Interleave. For any positive integer i, partition the set of all rounds into segments $R_1^i, R_2^i, \ldots$ of length 2^i, starting from round 1, i.e., $R_j^i = \{(j-1)2^i + 1, \ldots, j \cdot 2^i\}$. Consider the following schedule. Nodes 1 and 2 send messages, respectively, in the first and second round of each segment R_j^1 of length 2. Nodes 3 and 4 send messages, respectively, in the first and second round of each odd segment of length 2, i.e., R_j^1 for odd j. Nodes 5, 6, 7, and 8 send messages, respectively, in the first, second, third, and fourth round of odd segments of length 4, i.e., R_j^2 for odd j. In general, for any $i > 0$, processors $2^i + 1, \ldots, 2^{i+1}$ send messages in consecutive rounds of odd segments of length 2^j, i.e., R_j^i for odd j.

Figure 2.1 illustrates the schedule. Note that in the set $S_j = \{i_1 < i_2 < \cdots\}$ of nodes transmitting on any given round j, the node numbers grow at least exponentially, i.e., $i_{l+1} \geq 2i_l$ for every $l \geq 1$.

THEOREM 2.4. *For an n-processor system, if a global clock is available, then the deterministic online algorithm Interleave succeeds in waking up the system in time at*

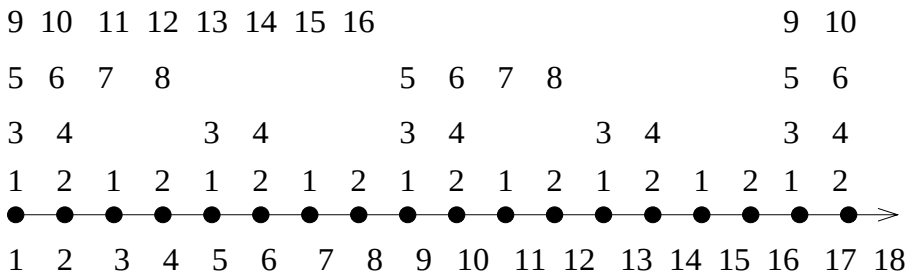


FIG. 2.1. *The schedule used by algorithm Interleave. Each node i is listed on all rounds in which it transmits.*

most $4n$, even when the number n of processors is not known to any of them.

Proof. Suppose that $2^k < n \leq 2^{k+1}$, and let r' be the round when the first processor is woken up. Let $r \leq r' + 2^{k+1}$ be the first round in which any processor sends a message. Let $S \subseteq \{1, \ldots, n\}$ be the set of all processors woken up by the adversary. We will show that some processor in S is the only one to send a message in a round $j \leq r + 2^{k+1} < r' + 4n$.

Let x_1 be the largest processor sending a message in round r. If no other processor sends a message in round r, we are done. Otherwise, let $x_2 < x_1$ be the largest such processor different from x_1. We have $x_2 \leq 2^k$. Consequently, in some round $r_2 \leq r + 2^k$, x_2 is the largest processor that sends a message. If no other processor sends a message in round r_2, we are done. Otherwise, let $x_3 < x_2$ be the largest such processor different from x_2. We have $x_3 \leq 2^{k-1}$. Consequently, in some round $r_3 \leq r + 2^k + 2^{k-1}$, x_3 is the largest processor that sends a message. Using this reasoning inductively, we conclude that there is a round $j \leq r + 2^{k+1} < r' + 4n$ in which exactly one processor in S sends a message. (Indeed, if processor 1 or processor 2 is the largest one to send a message in a round, it is also unique.) □

3. The locally synchronous model with known n. In this section we consider the locally synchronous model, where only local (equal rate) clocks are available at processors, and the local clock of each processor starts measuring time on the round when the processor wakes up. We assume that the size n of the system is known to all processors. In section 3.1 we present a randomized algorithm for the problem. We then turn to fixed schedules. Following section 3.2, which provides some necessary terminology, in section 3.3 we present a deterministic wakeup algorithm, section 3.4 describes a randomized schedule construction algorithm, and section 3.5 establishes a lower bound on wakeup time with a fixed schedule.

3.1. Randomized online algorithm. Consider the following straightforward randomized algorithm.

Algorithm Rand-Try. Upon waking up spontaneously, each processor performs the following in each round. It randomly sets a bit

$$b \leftarrow_R \begin{cases} 1 & \text{with probability } 1/n, \\ 0 & \text{with probability } 1 - 1/n. \end{cases}$$

If the outcome is $b = 1$, then it broadcasts a wakeup message.

THEOREM 3.1. *If the number n of processors is known to all of them, then the randomized algorithm Rand-Try succeeds in waking up the system in time $O(n \log(1/\varepsilon))$*

with probability at least $1-\varepsilon$.

Proof. Let $\mathcal{W}(t)$ denote the event of a successful wakeup in a round t in which k processors are awake. The success probability of this event is

$$\mathbb{P}(\mathcal{W}(t)) \;=\; \left(1-\frac{1}{n}\right)^{k-1}\cdot\frac{1}{n}\cdot k \;\geq\; \left(1-\frac{1}{n}\right)^{n}\cdot\frac{1}{n} \;\geq\; \frac{1}{2en}\,.$$

Hence the probability that none of the first $\lceil 2en\ln(1/\varepsilon)\rceil$ rounds succeed is at most

$$\mathbb{P}\left(\bigcap_{r\in R}\overline{\mathcal{W}(t)}\right) \;\leq\; \left(1-\frac{1}{2en}\right)^{2en\ln(1/\varepsilon)} \;\leq\; (1/e)^{\ln(1/\varepsilon)} \;=\; \varepsilon.$$

Hence after $O(n\log(1/\varepsilon))$ from the time the first processor woke up spontaneously, the system will be woken up with probability at least $1-\varepsilon$. ∎

3.2. Fixed schedules. Throughout the remainder of this section, we concentrate on fixed schedules (including deterministic algorithms as a special case). Let us begin by introducing some necessary terminology concerning activation sequences and executions. To begin with, note that each processor i may wake up and start its activation sequence α_i at a different time p_i (controlled by the adversary). As we start counting time from the round in which the first processor wakes up, at least one processor i must have $p_i = 0$. A sequence α_i is said to be *aligned* if it starts at time $p_i = 0$.

The resulting set of *start-time* assignments for the processors is denoted $P = \{(i,p_i) \mid 1\leq i\leq n\}$. Given such a set P, it is possible to view each sequence α_i as shifted to the right by p_i positions, and padded by zeros at the left, thus yielding some *actual activation sequence* α_i^P which is the actual sequence governing the actions of processor i in the execution corresponding to P. Hence when we talk hereafter about bit position t of the sequence α_i under the shift pattern P, we actually look at $\alpha_i^P[t]$, the tth bit of the sequence α_i^P, or equivalently at $\alpha_i[t-p_i]$, the $(t-p_i)$th bit of the original sequence α_i (if $t<p_i$, then this bit is zero by default).

The set P is henceforth referred to as the *shift pattern* of the execution. Later on we will also consider *partial* shift patterns, specifying the start-times for only some of the activation sequences, i.e., $P=\{(i_j,p_{i_j}) \mid 1\leq j\leq k\}$ for some $k<n$. For an activation set A and a shift pattern P, the set of shifted activation sequences resulting from applying P to A is denoted by $A^P=\{\alpha_i^P \mid \alpha_i\in A\}$. Recall that every shifted activation set A^P must contain at least one aligned sequence, as every shift pattern P contains at least one pair (i,p_i) with $p_i=0$. The original activation set (or equivalently, the set A^P resulting from selecting the starting time $p_i=0$ for all sequences) is referred to as the *aligned* activation set.

Definition 3.2. *The bit position $t\geq 0$ is* covered *by the shifted activation set A^P if there is exactly one sequence $\alpha_i\in A$ satisfying $\alpha_i^P[t]=1$, and the rest have $\alpha_j^P[t]=0$. Position t is* blocked *by A^P if it is not covered by it. It is* filled *by A^P if there is* at least *one sequence $\alpha_i\in A$ such that $\alpha_i^P[t]=1$.*

Two sequences α_i^P and α_j^P in the shifted activation set A^P collide *in position t if $\alpha_i^P[t]=\alpha_j^P[t]=1$.*

For integers $k\geq m\geq 0$, the shift pattern P is said to be $[m,k]$-blocking *for A if the shifted activation set A^P blocks every bit position $m\leq t\leq k$.*

For an activation set A, let $W(A)$ denote the worst-case wakeup time of A, namely, the minimal integer k such that there does not exist a $[0,k]$-blocking shift pattern P for A.

Note that an algorithm for the wakeup problem does not need to generate activation sequences of infinite length. Assuming it generates the sequences of an activation set A with $W(A) < \infty$, one bit at a time, it suffices to generate the first $W(A)$ bits.

Since the adversary controls the times when the processors wake up (i.e., it controls the shift pattern P), our problem of minimizing the wakeup time of the system is equivalent to constructing an activation set A of n binary sequences α_i of minimal wakeup time $W(A)$. Without loss of generality, all sequences start with bit 1, i.e., $\alpha_i[1] = 1$ for all $i = 1, \ldots, n$. (This is because there is a one-to-one correspondence between sequences of leading zeros and late wakeup times, and the latter are under the control of the adversary. Hence $W(A)$ cannot be improved by, say, padding each sequence $\alpha_i \in A$ with z_i leading zeros, since the adversary can always nullify the effect of the leading zeros, and mimic the worst shift pattern P for A on the modified activation set, simply by making every processor i wake up z_i rounds earlier than in P.)

3.3. A deterministic online algorithm.

Algorithm Prime-Steps. Let q_i, for $i = 1, \ldots, n$, be the ith prime number larger than n. We define a set of sequences $A = \{\alpha_i : i = 1, \ldots, n\}$ of lengths $m_i = nq_i + 1$ as follows. The sequence α_i, describing the behavior of processor i, has bit 1 on positions kq_i, for natural $k = 0, 1, \ldots, n$, and bit 0 on all other positions.

For example, suppose that $n = 4$. Then we take $q_1 = 5$, $q_2 = 7$, $q_3 = 11$, and $q_4 = 13$, and hence the (periodic) sequences of rounds $\bar{t}_i$ in which node i transmits (after waking up) are

$$\bar{t}_1 = (0, 5, 10, 15, 20),$$
$$\bar{t}_2 = (0, 7, 14, 21, 28),$$
$$\bar{t}_3 = (0, 11, 22, 33, 44),$$
$$\bar{t}_4 = (0, 13, 26, 39, 52).$$

The choice of algorithm Prime-Steps ensures that the sequences assigned to different processors collide rather infrequently, as detailed in the proof of the following theorem.

THEOREM 3.3. *Algorithm Prime-Steps wakes up a system of n processors in time $O(n^2 \log n)$.*

Proof. Since q_i is of size $O(n \log n)$, for all $i = 1, 2, \ldots, n$, we have $m_i \in O(n^2 \log n)$. Fix a shift pattern P. It suffices to show that there exists a bit position covered by A^P. Let i be the processor that wakes up first, i.e., $p_i = 0$. If bit position 0 is not covered, then some other sequence $\alpha^P_{j_1}$ necessarily has 1 in position 0. It follows that sequences α^P_i and $\alpha^P_{j_1}$ do not collide in positions $t > 0$. (Since q_i and q_{j_1} are primes, α^P_i and $\alpha^P_{j_1}$ collide as rarely as $q_i q_{j_1}$ which is larger than m_i and m_{j_1}.) If bit position q_i is not covered, it means that some sequence $\alpha^P_{j_2}$, different from α^P_i and $\alpha^P_{j_1}$, has 1 in position q_i. Again, sequences α^P_i and $\alpha^P_{j_2}$ do not collide in positions $t > q_i$. Consequently, in order to guarantee that bit positions kq_i, for $i = 0, 1, 2, \ldots$ are not covered, the adversary must generate a collision with a different sequence α_{j_k} for each bit position kq_i. Since there are only $n - 1$ sequences different from α_i, it follows that one of the bit positions kq_i, for $i = 0, 1, \ldots, n-1$, must be covered. Since $nq_i < m_i$, the theorem follows. ∎

3.4. The existence of a short fixed schedule. We now show that there exists a fixed schedule guaranteeing much faster wakeup times than those given by the deterministic online algorithm Prime-Steps. The proof is nonconstructive; we describe a randomized (Monte-Carlo) preprocessing algorithm Π^{random}, which randomly selects an activation set A for n processors, with the property that $W(A) = O(n \log^2 n)$ with probability at least $1 - \frac{1}{n}$. This implies that there must *exist* a fixed schedule with wakeup time $O(n \log^2 n)$, as the nonexistence of such a schedule would force the success probability of algorithm Π^{random} to be zero. However, we know of no efficient (deterministic or randomized) algorithmic way for ascertaining the wakeup time $W(A)$ of any given activation set A, whether constructed by the preprocessing algorithm Π^{random} or produced by any other means. Subsequently, we do not have a way of transforming algorithm Π^{random} into a polynomial expected time Las Vegas algorithm for constructing fixed schedules.

Set[1] $m = cn \log n \ln n$, for $c = 33$.

Preprocessing algorithm Π^{random}. Construct the sequences α_i of the activation set A by randomly setting each bit $\alpha_i[t]$, for $1 \le i \le n$ and $0 \le t \le m$, fixing

$$\alpha_i[t] \leftarrow_R \begin{cases} 1 & \text{with probability } 1/n, \\ 0 & \text{with probability } 1 - 1/n. \end{cases}$$

Analysis. For every $0 \le \ell \le \log n$, let $T_\ell = cn \ln n \cdot \ell$.

For every $0 \le t \le m$, let $w(t)$ denote the number of processors already awake by time t, $w(t) = |\{i \mid p_i \le t\}|$. Note that $w(t)$ is nondecreasing in the range $0 \le t \le m$.

DEFINITION 3.4. *P is said to be* ℓ-regular *if $w(T_\ell) \ge 2^\ell$ and $w(T_{\ell+1}) < 2^{\ell+1}$. Let $I(P)$ denote the smallest index such that P is* ℓ-regular,

$$I(P) \;=\; \min\{\ell \mid P \text{ is } \ell\text{-regular}\} \; .$$

Note. For every P, there exists some $0 \le \ell \le \log n$ such that P is ℓ-regular; hence $0 \le I(P) \le \log n$.

Partition the shift patterns into classes $\mathcal{P}_0, \ldots, \mathcal{P}_{\log n}$, such that $\mathcal{P}_\ell$ contains all shift patterns P for which $I(P) = \ell$,

$$\mathcal{P}_\ell = \{P \mid I(P) = \ell\} \; .$$

Consider a shift pattern $P \in \mathcal{P}_\ell$. Let $Cov(P,t)$ be the event that bit position t for $T_\ell \le t \le m$ is covered by P (i.e., it has exactly one 1 and $w(t) - 1$ zeros). Then

$$\begin{aligned} \mathbb{P}(Cov(P,t)) &\ge w(t) \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{w(t)-1} \\ &\ge 2^\ell \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n} \;\approx\; \frac{2^\ell}{en} \; . \end{aligned}$$

Let $\mathcal{B}(P,\ell)$ be the event that P is $[T_\ell, T_{\ell+1}]$-blocking. Then

$$\begin{aligned} \mathbb{P}(\mathcal{B}(P,\ell)) &\le \left(1 - \frac{2^\ell}{en}\right)^{cn \ln n} \;\approx\; \exp(-c \ln n \cdot 2^\ell / e) \\ &\le \exp(-c' \ln n \cdot 2^\ell) \; , \end{aligned} \tag{1}$$

[1] For notational simplicity we ignore rounding throughout the paper; whenever an integer value is called for, $\log n$ and $\ln n$ should be thought of as rounded upwards to $\lceil \log n \rceil$ and $\lceil \ln n \rceil$, respectively.

for $c' = 12$.

For any two sets of processors S_1, S_2 let $\mathcal{P}_\ell[S_1, S_2]$ denote the subclass of $\mathcal{P}_\ell$ in which

$$\begin{cases} 0 \le p_j \le T_\ell, & j \in S_1 \ , \\ T_\ell < p_j \le T_{\ell+1}, & j \in S_2 \ , \\ T_{\ell+1} < p_j, & j \notin S_1 \cup S_2 \ . \end{cases}$$

Note that the class $\mathcal{P}_\ell$ is the union of subclasses $\mathcal{P}_\ell[S_1, S_2]$ over all possible choices of processor sets S_1, S_2 such that $2^\ell \le |S_1| < 2^{\ell+1}$ and $0 \le |S_2| < 2^{\ell+1} - |S_1|$. Formally, letting

$$\mathcal{S} \ = \ \{(S_1, S_2) \mid 2^\ell \le |S_1| < 2^{\ell+1} \text{ and } 0 \le |S_2| < 2^{\ell+1} - |S_1|\} \ ,$$

we have

$$\mathcal{P}_\ell \ = \ \bigcup_{(S_1,S_2)\in\mathcal{S}} \mathcal{P}_\ell[S_1, S_2] \ .$$

Let $\beta(S_1)$ denote the set of all shift patterns for the processors of S_1 in the range $[0, T_\ell]$. Thus, for $S_1 = \{j_1, \ldots, j_s\}$,

$$\beta(S_1) \ = \ \{(p_{j_1}, \ldots, p_{j_s}) \mid 0 \le p_{j_l} \le T_\ell \text{ for } 1 \le l \le s\} \ .$$

Similarly, let $\gamma(S_2)$ denote the set of all shift patterns for the processors of S_2 in the range $(T_\ell, T_{\ell+1}]$, i.e., for $S_2 = \{j_1, \ldots, j_r\}$,

$$\gamma(S_2) \ = \ \{(p_{j_1}, \ldots, p_{j_r}) \mid T_\ell < p_{j_l} \le T_{\ell+1} \text{ for } 1 \le l \le r\} \ .$$

For shift patterns $\beta \in \beta(S_1)$ and $\gamma \in \gamma(S_2)$, let $\mathcal{P}_\ell[\beta, S_1, \gamma, S_2]$ denote the subclass of $\mathcal{P}_\ell[S_1, S_2]$ consisting of all shift patterns $P \in \mathcal{P}_\ell$ that match β over S_1 and γ over S_2. Hence

$$\mathcal{P}_\ell \ = \ \bigcup_{(S_1,S_2)\in\mathcal{S}} \ \bigcup_{\substack{\beta\in\beta(S_1)\\ \gamma\in\gamma(S_2)}} \mathcal{P}_\ell[\beta, S_1, \gamma, S_2] \ .$$

For any set $\mathcal{E}$ of shift patterns, let $\mathcal{B}(\mathcal{E}, \ell)$ denote the event that some $P \in \mathcal{E}$ is $[T_\ell, T_{\ell+1}]$-blocking and let $\mathcal{B}(\mathcal{E})$ denote the event that some $P \in \mathcal{E}$ is blocking. Clearly,

$$\mathbb{P}(\mathcal{B}(\mathcal{E})) \ \le \ \mathbb{P}(\mathcal{B}(\mathcal{E}, \ell)) \tag{2}$$

for every $\mathcal{E}$ and ℓ. Note that the event $\mathcal{B}(\mathcal{P}_\ell[\beta, S_1, \gamma, S_2], \ell)$ happens if and only if *every* $P \in \mathcal{P}_\ell[\beta, S_1, \gamma, S_2]$ is $[T_\ell, T_{\ell+1}]$-blocking, since for all $P, P' \in \mathcal{P}_\ell[\beta, S_1, \gamma, S_2]$, P and P' have the same shift configuration in the range $(T_\ell, T_{\ell+1}]$; hence P is $[T_\ell, T_{\ell+1}]$-blocking if and only if P' is $[T_\ell, T_{\ell+1}]$-blocking. Hence for every $P \in \mathcal{P}_\ell[\beta, S_1, \gamma, S_2]$,

$$\mathbb{P}(\mathcal{B}(\mathcal{P}_\ell[\beta, S_1, \gamma, S_2], \ell)) \ = \ \mathbb{P}(\mathcal{B}(P, \ell)) \ ,$$

and using inequalities (2) and (1),

$$\begin{aligned} \mathbb{P}(\mathcal{B}(\mathcal{P}_\ell[\beta, S_1, \gamma, S_2])) &\le \mathbb{P}(\mathcal{B}(\mathcal{P}_\ell[\beta, S_1, \gamma, S_2], \ell)) \\ &\le \exp(-c' \ln n \cdot 2^\ell) \ . \end{aligned} \tag{3}$$

As

$$\mathcal{B}(\mathcal{P}_\ell) \;=\; \bigcup_{(S_1,S_2)\in\mathcal{S}} \; \bigcup_{\substack{\beta\in\beta(S_1)\\ \gamma\in\gamma(S_2)}} \mathcal{B}(\mathcal{P}_\ell[\beta,S_1,\gamma,S_2])\ ,$$

we have that

$$\begin{aligned}\mathbb{P}(\mathcal{B}(\mathcal{P}_\ell)) &= \sum_{(S_1,S_2)\in\mathcal{S}} \; \sum_{\substack{\beta\in\beta(S_1)\\ \gamma\in\gamma(S_2)}} \mathbb{P}(\mathcal{B}(\mathcal{P}_\ell[\beta,S_1,\gamma,S_2]))\\ &\le \sum_{(S_1,S_2)\in\mathcal{S}} |\beta(S_1)|\cdot|\gamma(S_2)|\cdot\mathbb{P}(\mathcal{B}(P,\ell))\ .\end{aligned}$$

As

$$\begin{aligned}|\beta(S_1)| = (T_\ell+1)^s &\;=\; (1+cn\ln n\cdot\ell)^s\ ,\\ |\gamma(S_2)| = (T_{\ell+1}-T_\ell)^r &\;=\; (cn\ln n)^r \;\le\; (cn\ln n)^{2^{\ell+1}-s}\ ,\end{aligned}$$

we have

$$|\beta(S_1)|\cdot|\gamma(S_2)| \;\le\; (1+cn\ln n\log n)^{2^{\ell+1}}\ ;$$

hence by inequality (3)

$$\begin{aligned}\mathbb{P}(\mathcal{B}(\mathcal{P}_\ell)) &\le \sum_{(S_1,S_2)\in\mathcal{S}} (1+cn\ln n\log n)^{2^{\ell+1}}\cdot\exp(-c'\ln n\cdot 2^\ell)\\ &\le \sum_{s\le 2^{\ell+1}}\binom{n}{s}\sum_{r\le 2^{\ell+1}-s}\binom{n-s}{r}\cdot\exp(2\ln n\cdot 2^{\ell+1}-c'\ln n\cdot 2^\ell)\\ &\le n^{2^{\ell+2}+2}\cdot\exp((4-c')\ln n\cdot 2^\ell)\\ &\le \exp((10-c')\ln n\cdot 2^\ell)\\ &\le \exp(-2\ln n)\ .\end{aligned}$$

$\mathcal{B}(\mathcal{P}) = \bigcup_\ell \mathcal{B}(\mathcal{P}_\ell)$, so

$$\mathbb{P}(\mathcal{B}(\mathcal{P})) \;\le\; \log n\cdot\mathbb{P}(\mathcal{B}(\mathcal{P}_\ell)) \;\le\; \frac{\log n}{n^2} \ll \frac{1}{n}\ .$$

The above analysis yields the following theorem.

THEOREM 3.5. *For an n-processor system with n known to the processors, algorithm Π^{random} constructs an activation set A whose wakeup time is $W(A) = O(n\log^2 n)$ with probability at least $1-\frac{1}{n}$.*

COROLLARY 3.6. *For an n-processor system with n known to the processors, there exists a fixed schedule with wakeup time $O(n\log^2 n)$.*

It is important to underline the difference between algorithm Rand-Try and algorithm Π^{random}. Although randomness is involved in both of them, the former is an online algorithm, while the latter is a randomized method to produce a *fixed schedule*. Consequently, in algorithm Rand-Try the adversary does not know the behavior of processors in advance and must decide if and when to wake up each of them based only on the history to date. In the second scenario, on the other hand, the adversary is given a fixed schedule in advance, and hence it also has knowledge of the behavior of

processors in the future. This additional information gives the adversary more power which is reflected by the worse performance of the fixed schedule, as compared to the randomized online algorithm Rand-Try. In principle, our existence proof can be used for *generating* a (provably correct) short schedule; simply construct schedules one by one as indicated by the proof and test each of them until finding one satisfying the requirements. Unfortunately, testing the correctness of a given schedule (i.e., verifying that it succeeds in waking up the processors against any adversarial behavior) seems to be a difficult task, and we do not see any way of achieving it short of the naive brute-force testing of all possible schedules.

3.5. Lower bound for fixed schedules. We do not know what is the optimal time of a deterministic online wakeup algorithm (or fixed schedule). Below we give a lower bound that serves primarily to establish a gap between the global clock and local clocks scenarios; while in the former scenario we showed an algorithm working in time n, it turns out that without a global clock, the wakeup time of any fixed schedule (including in particular those generated by a deterministic algorithm) must be greater than $(1+\varepsilon)n$ for some positive constant ε.

Consider an activation set consisting of n infinite binary activation sequences, $A = \{\alpha_1, \ldots, \alpha_n\}$. We show that the adversary can select the waking times of processors, i.e., the shift pattern P for these sequences on the time axis, in such a way that no message is heard within the first $(1+\varepsilon)n$ rounds, counted from waking up the first processor, for some constant $\varepsilon > 0$. (No attempt has been made to optimize the constant. We give the proof for $\varepsilon = 0.001$, when n is sufficiently large.)

To establish the lower bound, it suffices to show that for some $1 \le x \le n$, there exist an activation set $B \subset A$ of cardinality x and an $[0, x+\varepsilon n]$-blocking shift pattern P for B. The remaining $n-x$ sequences can be used to block at least $n-x$ additional positions, $x+\varepsilon n+1, \ldots, n+\varepsilon n$, using one shifted sequence to block each of these positions.

LEMMA 3.7 (blocking technique). *Let Q be any set of m (not necessarily consecutive) positions and let B be any aligned activation set of $m+1$ sequences. Then there exists a nonempty subset of B which blocks all positions in Q.*

Proof. Repeat the following process at most m times. For any position j in Q covered by B, remove the sequence covering j from the set B. After removing at most m sequences, all positions of Q are blocked and at least one sequence remains in B. □

Let δ and σ be positive integer constants to be determined later and let $N = n/\sigma$.

For any sequence α_i, the *m-head* of α_i, denoted $head(\alpha_i, m)$, is its prefix of length m. Let $ones(\alpha_i, m)$ denote the number of ones in $head(\alpha_i, m)$.

LEMMA 3.8. *For every two integers $a \ge 1$ and $m \ge 2^a$, and for every aligned activation set B of m or more sequences satisfying $ones(\alpha_i, m) \le a$, there exist an integer $0 \le \ell \le a-1$ and a set $C \subseteq B$ of size $|C| > m/2^{\ell+1}$ such that every $\alpha_i \in C$ contains zeros at all positions $m/2^{\ell+1}+1$ to $m/2^{\ell}$.*

Proof. The proof is by induction on a. For $a = 1$, each sequence in B contains a single 1 at the first position and zeros at all other positions; hence the set $C = B$ satisfies the claim with $\ell = 0$. Considering $a > 1$, assume the claim holds for every $a' < a$. Let B' be the set of all sequences $\alpha_i \in B$ which contain a 1 at some position from $m/2+1$ to m. If $|B'| < m/2$, then the claim holds for $\ell = 0$ and $C = B \setminus B'$, and we are done. Otherwise, letting $m' = m/2$ and $a' = a-1$, the set B' satisfies $|B'| \ge m'$ and every sequence $\alpha_i \in B'$ satisfies $ones(\alpha_i, m') \le a'$. Hence by the inductive hypothesis, there exist an integer $0 \le \ell' \le a'-1$ and a set $C \subseteq B'$ of size

$|C| > m'/2^{\ell'+1}$ such that every $\alpha_i \in C$ contains zeros at all positions $m'/2^{\ell'+1}+1$ to $m'/2^{\ell'}$. The claim now holds for C and $\ell = \ell'+1$. □

Returning to the analysis, there are two cases.

Case 1*:* At least N sequences have N-heads with at most $\delta - 1$ ones.

Let B be an activation set consisting of N sequences with the above property. By Lemma 3.8, there exist an integer $0 \le \ell \le \delta - 2$ and a set $C \subseteq B$ of size $|C| > N/2^{\ell+1}$ such that every $\alpha_i \in C$ contains zeros at all positions $N/2^{\ell+1}+1$ to $N/2^{\ell}$. Using the blocking technique of Lemma 3.7, we can now use a subset $C' \subseteq C$ of at most $N/2^{\ell+1}+1$ sequences to block the first $N/2^{\ell+1}$ positions and gain $N/2^{\ell+1}$ blocked positions "for free." Then we use the remaining $n - |C'|$ sequences to block $n - |C'|$ additional positions for a total of $n + \frac{N}{2^{\ell+1}} - 1$ positions. In the worst case, occurring when $\ell = \delta - 2$, this amounts to $n + \frac{n}{\sigma \cdot 2^{\delta-1}} - 1$ blocked positions.

Case 2*:* More than $n - N$ sequences have N-heads with at least δ ones.

Let $M = n - N$ and let B be an activation set of M sequences with the above property. Let I_n denote the time interval of the first $n+1$ positions, $0, 1, \ldots, n$. We show that we can block all positions of I_n by selecting a subset $C \subseteq B$ of ρn sequences, for $\rho \le 1 - 1/\sigma$, and a shift pattern P for C, in such a way that each position of I_n is filled by at least two sequences of C^P. Assume, without loss of generality, that the number of ones in each N-head is exactly δ (ignore other ones).

For each sequence $\alpha_i \in B$ we consider only M possibilities of right shifts from the aligned position, namely, by $p_i \in \{1, \ldots, M\}$. We call these the *possible* shifts. For each of them, the entire N-head of α_i corresponds to positions within the interval I_n. We construct a shift pattern P for a subset of sequences $C \subseteq B$. The set C is initialized to $\emptyset$. Our goal is to fill all positions in I_n by C^P. Suppose that at some stage of the construction, some positions in I_n are already filled by C^P. Selecting a new sequence α_i to be added to C, we look for a shift p_i that fills as many new positions as possible (i.e., we try to align many ones from the N-head of α_i with yet unfilled positions in I_n).

LEMMA 3.9. *If the number of filled positions in I_n is strictly less than kM/δ, for $1 \le k \le \delta$, then for any sequence $\alpha_i \in B$ there exists a possible shift p_i which fills at least $\delta - k + 1$ new positions in I_n.*

Proof. Assume the contrary and consider a sequence α_i such that for each possible shift p_i of α_i, at least k ones of its N-head correspond to filled positions in I_n. Thus in total we have at least kM such *matches*.

On the other hand, each filled position in I_n can be matched with at most δ ones from the N-head of α_i. There are strictly fewer than kM/δ filled positions in I_n, so the total number of matches is strictly less than kM, a contradiction. □

The process of constructing C and filling the positions of the interval I_n is divided into δ consecutive stages. In stage k, when the number of filled positions in I_n is at least $(k-1)M/\delta$ but strictly smaller than kM/δ, it is possible to add in each step an appropriately shifted sequence to C filling at least new $\delta - k + 1$ positions in I_n. Thus stage k consists of at most $(\frac{kM}{\delta} - \frac{(k-1)M}{\delta}) \cdot \frac{1}{\delta-k+1}$ such steps, and the total number of sequences needed to fill all positions of I_n is at most

$$\sum_{k=1}^{\delta} \left\lceil \frac{1}{\delta - k + 1} \cdot \frac{M}{\delta} \right\rceil + N \;\le\; \frac{M}{\delta} \sum_{k=1}^{\delta} \frac{1}{k} + N + \delta.$$

Notice that each of the last N sequences is used to fill only a single position thus we can use sequences outside of B for this purpose.

It follows that all positions of I_n can be *blocked* by repeating this process twice, and the number of sequences used is at most

$$2n\left(\frac{1}{\delta}\left(1-\frac{1}{\sigma}\right)\sum_{k=1}^{\delta}\frac{1}{k}+\frac{1}{\sigma}\right)+2\delta\ .$$

Finally take constants $\delta = 8$ and $\sigma = 5$. Then $\frac{1}{\delta}\sum_{k=1}^{\delta}\frac{1}{k} < 0.34$, and the total number of sequences used to block all positions of I_n is at most $\lceil 0.95n\rceil + 16$. The remaining $\lfloor 0.05n\rfloor - 16$ sequences are used to block $\lfloor 0.05n\rfloor - 16$ additional positions for a total of $\lfloor 1.05n\rfloor - 16$ positions. In Case 1 we could block at least $n + \frac{n}{\sigma\cdot 2^{\delta-1}} - 1 = n + \frac{n}{640} - 1$ positions. This gives a lower bound $1.001n$ in both cases for sufficiently large n.

THEOREM 3.10. *If a global clock is not available, then every deterministic algorithm waking up a system of n processors must use worst-case time at least $(1+\varepsilon)n$ for some positive constant ε and sufficiently large n.*

4. The locally synchronous model with unknown n. We conclude the paper by presenting two wakeup algorithms working under the weakest scenario, namely, when a global clock is not available and the size n of the system is not known to processors. The first is a randomized online algorithm waking up the system in time $O(n^2\log(1/\varepsilon))$ with probability at least $1-\varepsilon$, and the second is a deterministic online algorithm working in worst case time $O(n^4\log^5 n)$.

4.1. Randomized algorithm Size-Probing. For any integer $j \geq 1$ let $T_j = \lceil e\ln(1/\varepsilon)\rceil \cdot 2^{j+1}$. Also let $S_i = \sum_{j=1}^{i} T_j = \lceil e\ln(1/\varepsilon)\rceil \cdot (2^{j+2}-2)$. Any processor, after waking up spontaneously, operates in consecutive phases numbered by positive integers. Phase j lasts T_j rounds. In each of these rounds the processor randomly sets a bit

$$b \leftarrow_R \begin{cases} 1 & \text{with probability } 1/2^j, \\ 0 & \text{with probability } 1-1/2^j. \end{cases}$$

If the outcome is $b = 1$, then it broadcasts a wakeup message.

THEOREM 4.1. *Algorithm Size-Probing succeeds in waking up an n-processor system in time $O(n^2\log(1/\varepsilon))$ with probability at least $1-\varepsilon$.*

Proof. Let i be the unique integer satisfying that $2^{i-1} < n \leq 2^i$. Our analysis focuses on what happens in the system beginning on the *special* round τ which is the first round such that some processor wakes up (spontaneously) on round τ and no processor wakes up on rounds $\tau+1,\ldots,\tau+S_i$. Clearly $\tau \leq nS_i = O(n^2\log(1/\varepsilon))$.

Consider the set of rounds $R = \{\tau+S_{i-1}+1,\ldots,\tau+S_i\}$. Let p be the processor that woke up on round τ and let X be the set of awake processors at the beginning of round τ (excluding p). In each round $t \in R$, processor p broadcasts with probability $1/2^i$ and all other awake processors from X broadcast with probabilities *at most* $1/2^i$. For a round $t \in R$, let $\mathcal{W}(t)$ denote the event that round $t \in R$ *succeeds*, namely, exactly one processor broadcasts. The probability that this happens is

$$\begin{aligned}\mathbb{P}(\mathcal{W}(t)) &\geq \frac{1}{2^i}\left(1-\frac{1}{2^i}\right)^n \ \geq\ \frac{1}{2^i}\left(1-\frac{1}{n}\right)^n \\ &\geq \frac{1}{2^i}\cdot\frac{1}{2e} \ =\ \frac{1}{2^{i+1}\cdot e}\ .\end{aligned}$$

Hence the probability that none of the rounds in R succeed is at most

$$\mathbb{P}\left(\bigcap_{t\in R}\overline{\mathcal{W}(t)}\right) \leq \left(1-\frac{1}{2^{i+1}\cdot e}\right)^{T_i}$$

$$\leq \left(1 - \frac{1}{2^{i+1} \cdot e}\right)^{2^{i+1} e \ln(1/\varepsilon)}$$
$$\leq (1/e)^{\ln(1/\varepsilon)} \;=\; \varepsilon.$$

Hence the system is woken up with probability at least $1-\varepsilon$ after $\tau+S_i = O(n^2 \log(1/\varepsilon))$ rounds. ∎

4.2. Deterministic algorithm Squared Prime-Steps. In order to present our last algorithm, define the integer sequence $S = \langle s_1, s_2, \ldots \rangle$ by setting

$$s_j = \begin{cases} 1, & j = 1, \\ 2, & j = 2, \\ s_{j-1}^2 \log^2 s_{j-1}, & j \geq 3. \end{cases}$$

For every processor i, find a value j, such that $s_{j-1} < i \leq s_j$, and construct the sequence α_i describing the behavior of processor i as in algorithm Prime-Steps assuming the size of the system is s_j. Take the resulting set of sequences α_i, padded by zeros on the right, to be the activation set of the algorithm.

THEOREM 4.2. *Algorithm Squared Prime-Steps wakes up a system of n processors in time* $O(n^4 \log^5 n)$.

Proof. Fix any shift pattern P. Consider the set $\mathcal{S}$ of processors that are active during the execution of algorithm Squared Prime-Steps. Let t be the largest processor number in $\mathcal{S}$ and let j be such that $s_{j-1} < t \leq s_j$. Let $\hat{\mathcal{S}} \subseteq \mathcal{S}$ be the set of all processors with indices larger than s_{j-1} and $\hat{A} = \{\alpha_i : i \in \hat{\mathcal{S}}\}$. Recall that $|\alpha_l| \in O(s_{j-1}^2 \log s_{j-1})$ for all $l \in \mathcal{S} \setminus \hat{\mathcal{S}}$. Therefore

$$\sum_{l \in \mathcal{S} \setminus \hat{\mathcal{S}}} |\alpha_l| \in s_{j-1} \cdot O(s_{j-1}^2 \log s_{j-1}) = O(s_{j-1}^3 \log s_{j-1}) \subseteq O(s_j^2).$$

This means that the time used by algorithm Squared Prime-Steps, before processors in $\hat{\mathcal{S}}$ become active, is in $O(s_j^2)$. In what follows we show that the system is woken up in time at most $|\alpha_i|$ after the first processor $i \in \hat{\mathcal{S}}$ become active. Recall that sequence α_i has $s_j + 1$ positions set to 1. Since any two sequences in $\hat{A}^P$ collide at most once (see proof of Theorem 3.3) and $|\hat{\mathcal{S}}| \leq s_j - s_{j-1}$, the number of positions where α_i^P is set to 1 but all other sequences in $\hat{A}^P$ are set to 0 is larger than $s_{j-1} + 1$. Notice that each sequence α_l^P, for $l \in \mathcal{S} \setminus \hat{\mathcal{S}}$, can collide in at most one position with the sequence α_i^P. This holds since $|\alpha_l| \in O(s_{j-1}^2 \log s_{j-1})$ and distances between consecutive ones in α_i are at least $s_j \geq s_{j-1}^2 \log^2 s_{j-1}$. Since $|\mathcal{S} \setminus \hat{\mathcal{S}}| \leq s_{j-1}$, the number of covered positions is larger than $s_{j-1} + 1 - s_{j-1} = 1$.

Hence the wakeup time is bounded by $|\alpha_i| + O(s_j^2) \subseteq O(s_j^2 \log s_j)$. Since $n \geq t$ and $s_j \leq t^2 \log^2 t$ we get the bound $O(n^4 \log^5 n)$ of wakeup time for a system of n processors. ∎

REFERENCES

[1] N. ALON, A. BAR-NOY, N. LINIAL, AND D. PELEG, *A lower bound for radio broadcast*, J. Comput. System Sci., 43 (1991), pp. 290–298.

[2] R. BAR-YEHUDA, O. GOLDREICH, AND A. ITAI, *On the time complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization*, J. Comput. System Sci., 45 (1992), pp. 104–126.

[3] R. Bar-Yehuda, O. Goldreich, and A. Itai, *Efficient emulation of single-hop radio network with collision detection on multi-hop radio network with no collision detection*, Distrib. Comput., 5 (1991), pp. 67–71.
[4] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Englewood Cliffs, NJ, 1987.
[5] I. Chlamtac and S. Kutten, *On broadcasting in radio networks – problem analysis and protocol design*, IEEE Trans. Comm., 33 (1985), pp. 1240–1246.
[6] I. Chlamtac and S. Kutten, *Tree based broadcasting in multihop radio networks*, IEEE Trans. Comput., 36 (1987), pp. 1209–1223.
[7] I. Chlamtac and S. Kutten, *A spatial reuse TDMA/FDMA for mobile multi-hop radio networks*, in Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Washington, D.C., 1985, pp. 389–394.
[8] I. Chlamtac and O. Weinstein, *The wave expansion approach to broadcasting in multihop radio networks*, IEEE Trans. Comm., 39 (1991), pp. 426–433.
[9] B.S. Chlebus, L. Gasieniec, A. Gibbons, A. Pelc, and W. Rytter, *Deterministic broadcasting in unknown radio networks*, in Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, 2000, pp. 861–870.
[10] K. Diks, E. Kranakis, D. Krizanc, and A. Pelc, *The impact of knowledge on broadcasting time in radio networks*, in Proceedings of the 7th Annual European Symposium on Algorithms, ESA'99, Prague, Czech Republic, 1999, Lecture Notes in Comput. Sci. 1643, Springer, Berlin, 1999, pp. 41–52.
[11] S. Even and S. Rajsbaum, *Unison, canon, and sluggish clocks in networks controlled by a synchronizer*, Math. Systems Theory, 28 (1995), pp. 421–435.
[12] F. Fich, R. Impagliazzo, B. Kapron, V. King, and M. Kutylowski, *Limits on the power of parallel random access machines with weak forms of write conflict resolution*, J. Comput. System Sci., 53 (1996), pp. 104–111.
[13] M.J. Fischer, S. Moran, S. Rudich, and G. Taubenfeld, *The wakeup problem*, SIAM J. Comput., 25 (1996), pp. 1332–1357.
[14] I. Gitman, R. M. Van Slyke, and H. Frank, *Routing in packet-switching broadcast radio networks*, IEEE Trans. Comm., 24 (1976), pp. 926–930.
[15] J. Goodman, A. G. Greenberg, N. Madras, and P. March, *On the stability of Ethernet*, in Proceedings of the 17th Symposium on Theory of Computing, Providence, RI, 1985, pp. 379–387.
[16] A. G. Greenberg, P. Flajolet, and R. E. Ladner, *Estimating the multiplicities of conflicts to speed their resolutions in multiple access channels*, J. ACM, 34 (1987), pp. 289–325.
[17] J. Hastad, T. Leighton, and B. Rogoff, *Analysis of backoff protocols for multiple access channels*, in Proceedings of the 19th ACM Symposium on Theory of Computing, 1987, pp. 241–253.
[18] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman, *Advances in packet radio technology*, Proc. IEEE, 66 (1978), pp. 1468–1496.
[19] R. C. Kunzelman, *Overview of the ARPA packet radio experimental network*, in Proceedings of COMPCON, 1978, pp. 157–160.
[20] E. Kushilevitz and Y. Mansour, *Computation in noisy radio networks*, in Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, 1998, pp. 236–243.
[21] J. Mazoyer, *On optimal solutions to the firing squad synchronization problem*, Theoret. Comput. Sci., 168 (1996), pp. 367–404.
[22] N. Shacham and E. J. Craighill, *Dynamic routing for real-time transport in packet radio networks*, in Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 1982, pp. 152–158.
[23] A. Tannenbaum, *Computer Networks*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 223–229

A NEW PROPERTY AND A FASTER ALGORITHM FOR BASEBALL ELIMINATION*

KEVIN D. WAYNE†

Abstract. In the baseball elimination problem, there is a league consisting of n teams. At some point during the season, team i has w_i wins and g_{ij} games left to play against team j. A team is eliminated if it cannot possibly finish the season in first place or tied for first place. The goal is to determine exactly which teams are eliminated. The problem is not as easy as many sports writers would have you believe, in part because the answer depends not only on the number of games won and left to play but also on the schedule of remaining games. In the 1960's, Schwartz showed how to determine whether *one particular team* is eliminated using a maximum flow computation.

This paper indicates that the problem is not as difficult as many mathematicians would have you believe. For each team i, let g_i denote the number of games remaining. We prove that there exists a value W^* such that team i is eliminated if and only if $w_i + g_i < W^*$. Using this surprising fact, we can determine *all* eliminated teams in time proportional to a single maximum flow computation in a graph with n nodes; this improves upon the previous best known complexity bound by a factor of n.

Key words. network flow, combinatorial optimization

AMS subject classifications. 05C85, 68R10, 68Q25, 90B10, 90C27, 90C90

PII. S0895480198348847

1. Introduction.

In the *baseball elimination problem*, there is a league consisting of n teams, which we denote by the set T. At some point during the season, each team has played some number of games. Team $i \in T$ has w_i wins, g_{ij} remaining games against team $j \in T$, and $g_i = \sum_{j \in T} g_{ij}$ total remaining games. Table 1.1 gives the input data for a sample league. The goal of a team is to finish the season with the most wins. We say that a team is *eliminated* if it cannot finish in first place (i.e., with the most wins or tied for the most wins) for *any* possible outcome of the remaining games. We assume there are no ties (i.e., each game has a winner and loser) and no rain-outs (i.e., all remaining games are played). Without loss of generality, we assume all of the remaining games are against other teams in the same league. This classical problem was first popularized by Alan Hoffman in the 1960's as a nice application of optimization and network flow. The reader is referred to [2, 4] for textbook treatments of the problem.

Schwartz [15] proposed a method to determine whether a *single* team is eliminated using a maximum flow computation. Hoffman and Rivlin [11] generalized the result of [15], providing a characterization of when a team is eliminated from finishing in tth place. Robinson [14] gave a linear programming based model that finds the maximum lead a team can have at the end of the season. Gusfield and Martel [10] and McCormick [12] determined the *elimination number*, i.e., the minimum number of remaining games a team must win in order to have any chance of finishing in first place. Their methods use different extensions of the parametric maximum flow techniques of Gallo, Grigoriadis, and Tarjan [5]. McCormick [12] also showed that it is

*Received by the editors November 23, 1998; accepted for publication (in revised form) January 10, 2001; published electronically April 3, 2001. A preliminary version of this paper has appeared in *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, Baltimore, 1999, pp. 815–819.

http://www.siam.org/journals/sidma/14-2/34884.html

†Computer Science Department, Princeton University, Princeton, NJ 08544 (wayne@cs.princeton.edu). This research was supported by ONR through grant AASERT N00014-97-1-0681 while the author was at Cornell University.

TABLE 1.1
Team standings and remaining schedule.

team	wins	to play	schedule			
i	w_i	g_i	Atl	Phi	NY	Mon
Atlanta	83	8	–	1	6	1
Philadelphia	79	4	1	–	0	3
New York	78	7	6	0	–	1
Montreal	76	5	1	3	1	–

NP-complete to determine whether a team is eliminated from finishing the season in tth place or better. Adler et al. [1] proposed an integer programming formulation to determine which teams are eliminated from finishing the season in first place or as a wildcard playoff team and corresponding elimination numbers. Their Web site www.riot.ieor.berkeley.edu/~ baseball maintains these statistics on-line for major league baseball.

In this paper we introduce a new structural property for the baseball elimination problem. Specifically, we order the teams according to their total number of wins possible (current wins + remaining games). We show that if a team is eliminated, then so are all teams below it in the ordering. For example, this implies that if two teams have the same number of wins and remaining games, then they are either both eliminated or both not eliminated, regardless of their remaining opponents. Using our new ordering and binary search, we can find *all* eliminated teams with $\log n$ maximum flow computations. Using the parametric maximum flow techniques of Gallo, Grigoriadis, and Tarjan [5], we show how to determine all eliminated teams in the same complexity as a single maximum flow computation. It is also straightforward to determine all of the elimination numbers from our computation.

We note that the new structural property was independently proved by Adler et al. [1] using linear programming techniques. They also describe how to compute all eliminated teams by solving a single linear program. Our proof is based on flows and cuts and, as a result, leads to a faster algorithm.

2. Preliminaries. In this section we review the necessary and sufficient conditions for a team to be eliminated. Also, we show how to determine whether a single team is eliminated using a maximum flow computation.

Let x_{ij} be a variable representing the number of games that team $i \in T$ wins among games remaining to be played against team $j \in T$. Team k is *not eliminated* if there is some assignment of nonnegative integer values $\{x_{ij} : i, j \in T\}$ such that

$$\forall i, j \in T : \qquad x_{ij} + x_{ji} = g_{ij} = g_{ji}, \tag{2.1}$$

$$\forall j \in T : \qquad w_k + \sum_{j \in T} x_{kj} \geq w_i + \sum_{j \in T} x_{ij}. \tag{2.2}$$

Equations (2.1) imply that all remaining games are played; inequalities (2.2) imply that no team finishes the season with more wins than team k.

Consider Table 1.1 above. Montreal is eliminated since it can finish with at most 81 wins, but Atlanta already has 83 wins. This is the simplest reason for elimination. However, there can be more complicated reasons. For example, Philadelphia is also eliminated. It can finish the season with at most 83 wins. However, either Atlanta will win more than 83 games, or it will lose all 6 of its remaining games against New York, in which case New York will finish with at least 84 wins.

For any subset of teams $R \subseteq N$, let $w(R) = \sum_{i \in R} w_i$ denote the total number of games already won by teams in R and let $g(R) = \sum_{\{i,j\} \subseteq R} g_{ij}$ denote the number of games remaining to be played by teams both in R. We define $a(R) = \frac{w(R)+g(R)}{|R|}$ and note that $a(R)$ gives a lower bound on the average number of games (including games already won) that must be won by teams in R: the teams in R have already won $w(R)$ games, and some team in R must win each of the $g(R)$ games played between teams both in R.

LEMMA 2.1. *Let $i \in T$ and $R \subseteq T - \{i\}$. If $a(R) > w_i + g_i$, then team i is eliminated.*

Proof. If team i wins all of its remaining games, then it will finish the season with $w_i + g_i$ wins. On average, the teams in R win at least $a(R) > w_i + g_i$ games. Thus (at least) one team in R will finish with more wins than team i. □

In this case, we say that R *eliminates* i, since it provides a certificate of elimination for team i. Surprisingly, if a team is eliminated, there is always such a simple certificate of elimination, as stated in Theorem 2.3. First, we review how to determine whether or not a single team is eliminated. The following theorem is due to Schwartz [15].

THEOREM 2.2. *Using a single s-t minimum cut computation, we can determine whether one particular team k is eliminated.*

Proof. Clearly, the best possible scenario for team k is if it wins all of its remaining games, in which case it will end up with $W := w_k + g_k$ wins. If $W < w_i$ for any $i \in T$, then $\{i\}$ trivially eliminates k.

Now, we check for more complicated reasons for elimination. We construct a bipartite network in which feasible integral flows correspond to outcomes of the remaining schedule. The following network flow formulation is due to Schwartz [15]: Gusfield and Martel [10] give an alternate construction. There are nodes corresponding to teams and to remaining games. Intuitively, each unit of flow in the network corresponds to a remaining game. As it flows through the network, it passes from a game node, say, between teams i and j, then through one of the team nodes i or j, classifying this game as being won by that team.

The flow network for the baseball elimination problem is shown in Figure 2.1. Formally, let $N := T - \{k\}$ denote the set of teams other than team k. Let $P := \{\{i,j\} \subseteq N : g_{ij} > 0\}$ denote the set of pairs of teams (that don't involve team k) with remaining games to be played. Let $V := P \cup N \cup \{s,t\}$ denote the set of nodes in the network. For each $\{i,j\} \in P$ we include an arc $(s, \{i,j\})$ with capacity g_{ij}. For each team $i \in N$ we include an arc (i,t) with capacity $W - w_i$. Finally, for each $\{i,j\} \in P$ we include arcs $(\{i,j\}, i)$ and $(\{i,j\}, j)$ with infinite capacity. The flow on arc $(\{i,j\}, i)$ represents the total number of remaining games in which i beats j. The flow on arc (i,t) represents the total number of remaining games won by i.

It is easy to see that integral feasible flows of value $g(N)$ in the resulting network are in one-to-one correspondence with possible outcomes of the remaining games in which team i is not eliminated, i.e., they satisfy (2.1) and (2.2). It follows that we can determine whether i is eliminated with a single maximum integer flow (or minimum s-t cut) computation in the above bipartite network. □

The previous theorem says that we can determine whether any single team k is eliminated using a maximum flow computation in an appropriate bipartite network. In fact, if team k is eliminated, then the minimum s-t cut (in the same bipartite network) indicates a subset of teams that eliminates k. The next theorem is due to Hoffman and Rivlin [11]. We include its proof only for completeness.

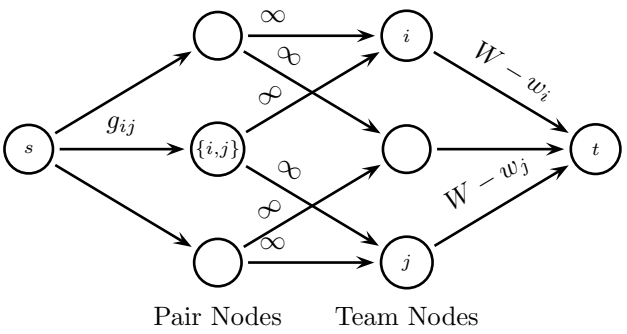


FIG. 2.1. *Flow network for baseball elimination.*

THEOREM 2.3. *Suppose team $k \in T$ is eliminated. Then there exists $R \subseteq T - \{k\}$ that eliminates k. Moreover, we can find such a subset R with a single s-t minimum cut computation.*

Proof. Consider the maximum flow network described in Theorem 2.2. Let S denote the source side of a minimum s-t cut and let $R = N \cap S$ denote the team nodes on the source side of the cut. For example, for the four team league considered above with k = Philadelphia, it turns out that the minimum cut is $S = \{s, \{\text{Atl, NY}\}, \text{Atl, NY}\}$ and $R = \{\text{Atl, NY}\}$. We note that R eliminates Philadelphia since $a(R) = 167/2 > 83 = W = w_k + g_k$.

In general, if team k is eliminated, we show that $R = N \cap S$ eliminates k. Since k is eliminated, the maximum flow in the network is less than $g(N)$. Hence, by the max-flow min-cut theorem, the capacity of the minimum cut S is also less than $g(N)$; it is the sum of the capacities of some arcs leaving the source and some arcs entering the sink:

$$\begin{aligned} \sum_{\{i,j\} \in P} g_{ij} = g(N) &> cap(S) \\ &= \sum_{\{i,j\} \in P \setminus S} g_{ij} + \sum_{i \in R} (W - w_i) \\ &= \sum_{\{i,j\} \in P \setminus S} g_{ij} + W|R| - w(R). \end{aligned} \tag{2.3}$$

Let $\{i, j\} \in P \cap S$. Then $i \in R$ and $j \in R$, since otherwise the cut would have infinite capacity. Thus

$$\sum_{\{i,j\} \in P \cap S} g_{ij} \leq \sum_{\{i,j\} \subseteq R} g_{ij} = g(R). \tag{2.4}$$

Combining (2.3) and (2.4) we obtain

$$\begin{aligned} W|R| - w(R) &< \sum_{\{i,j\} \in P} g_{ij} - \sum_{\{i,j\} \in P \setminus S} g_{ij} \\ &= \sum_{\{i,j\} \in P \cap S} g_{ij} \\ &\leq g(R). \end{aligned}$$

In other words, R eliminates team k. ∎

3. Problem structure. We now provide a new structural property for the baseball elimination problem. We use the total order $i \preceq j$ to indicate $w_i + g_i \leq w_j + g_j$. The following theorem indicates that if a team is eliminated, then so are all lower ordered teams.

THEOREM 3.1. *Suppose team $k \in T$ is eliminated. If $i \preceq k$, then team i is also eliminated.*

Proof. Since k is eliminated, by Theorem 2.3 there exists $R \subseteq T - \{k\}$ that eliminates k. That is

$$a(R) > w_k + g_k \geq w_i + g_i.$$

If $i \notin R$, then R also eliminates i. Now suppose $i \in R$. Clearly $R \neq \{i\}$. Then, $R \setminus \{i\}$ eliminates i since

$$\begin{aligned} a(R \setminus \{j\}) &= \frac{g(R - \{i\}) + w(R - \{i\})}{|R| - 1} \\ &\geq \frac{g(R) - g_i + w(R) - w_i}{|R| - 1} \\ &> \frac{g(R) + w(R) - a(R)}{|R| - 1} \\ &= a(R) \\ &> w_i + g_i. \qquad \square \end{aligned}$$

The following corollary was also derived independently by Adler et al. [1] using linear programming techniques instead of flows of cuts.

COROLLARY 3.2. *There exists a team $i^* \in T$ such that all teams $i \preceq i^*$ are eliminated and all teams $i \succ i^*$ are not eliminated.*

Proof. Choose i^* to be the eliminated team with the largest value of $w_i + g_i$. $\square$

COROLLARY 3.3. *There exists a single subset of teams $R^* \subseteq T$ that eliminates every eliminated team.*

Proof. Choose R^* to be a nonempty subset of teams that maximizes $a(R)$. First we observe that if team k is eliminated, then $k \notin R^*$. This follows from our choice of R^* because the proof of Theorem 3.1 would then imply $a(R^* - \{k\}) > a(R^*)$. By Theorem 2.3, if team k is eliminated, then there exists a subset R such that $a(R) > w_k + g_k$. Now $a(R^*) \geq a(R)$ and $k \notin R^*$, so R^* also eliminates k. $\square$

4. Determining all eliminated teams. In this section we show how to find all eliminated teams efficiently. It suffices to find the i^* guaranteed by Corollary 3.2. We can order the n teams according to their $w_i + g_i$ values and use binary search to find i^*. This requires $\log n$ minimum cut computations.

Now, we give an even faster method to find all eliminated teams. It suffices to find the R^* guaranteed by Corollary 3.3. We introduce an artificial team 0 which has no remaining games and a variable number of wins W. Let R^* be a nonempty subset that maximizes $a(R)$ and let $W^* = a(R^*)$. Note that team 0 is eliminated if and only if $W < W^*$. Also the elimination number for team i is easily seen to be $\lceil W^* \rceil - w_i$.

Now, we show how to find W^* and R^* efficiently. We construct a bipartite maximum flow network as in Figure 2.1, but now $k = 0$ and $N = T$. Also for each $i \in N$, the capacity of arc (i, t) is $W - w_i$, where W is a parameter. Note that all of the "parametric arcs" enter the sink and are increasing linear functions of the

parameter W. Therefore, we are in a position to apply the parametric maximum flow technique of Gallo, Grigoriadis, and Tarjan [5] which computes all minimum cut values parametrically in terms of W in the same complexity as a single preflow-push maximum flow computation. Thus, we can compute W^* efficiently. The team nodes on the source side of the minimum cut gives R^*. The following theorem summarizes this discussion.

THEOREM 4.1. *Let $G = (V, E)$ be an undirected graph with arc weights g_{ij} and node weights w_i. We can find a nonempty subset of nodes R that maximizes $a(R) := \frac{g(R)+w(R)}{|R|}$ using a single monotone parametric maximum flow computation.*

If the undirected network G has n nodes and m arcs, then the bipartite network we construct has $\mathcal{O}(m)$ nodes and $\mathcal{O}(m)$ arcs. However, the smaller side of the bipartition has only $n_1 = \mathcal{O}(n)$ nodes. For a network with n nodes and m arcs, the Goldberg–Tarjan [9] preflow-push algorithm solves the maximum flow problem in $\mathcal{O}(mn \log(m/n^2))$ time, and the Goldberg–Rao [8] algorithm requires $\mathcal{O}(\min(n^{2/3}, m^{1/2})m \log(n^2/m) \log U)$ time if the capacities are integers between 1 and U. In our problem $U \leq \max_{i \in T}(w_i + g_i)$. Using the bipartite maximum flow techniques of Ahuja et al. [3] and Goldberg [7], the running times remain valid for bipartite networks when the number of nodes n is replaced by the number of nodes on the smaller side of the bipartition n_1.

COROLLARY 4.2. *All eliminated teams can be determined in time proportional to one preflow-push maximum flow computation in a network with n nodes.*

The problem considered in Theorem 4.1 generalizes the *maximum density subgraph* problem considered by Goldberg [6]. In the maximum density subgraph problem, the goal is to find a subset of nodes that maximizes the ratio of the number of internal arcs to the number of nodes. This is the special case of our problem when the arc weights are uniform and the node weights are zero. Picard and Queyranne [13] and Gallo, Grigoriadis, and Tarjan [5] considered a different generalization of the maximum density subgraph problem that maximizes $g(R)/w(R)$.

Acknowledgments. The author is thankful to Éva Tardos for helpful discussions. The author also thanks the referees for providing useful comments that aided in the presentation of this paper.

REFERENCES

[1] I. ADLER, A. L. ERERA, D. S. HOCHBAUM, AND E. V. OLINICK, *Baseball, optimization, and the world wide web*, Interfaces, to appear.

[2] R. K. AHUJA, T. L. MAGNANTI, AND J. B. ORLIN, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[3] R. K. AHUJA, J. B. ORLIN, C. STEIN, AND R. E. TARJAN, *Improved algorithms for bipartite network flow*, SIAM J. Comput., 23 (1994), pp. 906–933.

[4] W. J. COOK, W. H. CUNNINGHAM, W. R. PULLEYBLANK, AND A. SCHRIJVER, *Combinatorial Optimization*, John Wiley, New York, 1998.

[5] G. GALLO, M. D. GRIGORIADIS, AND R. E. TARJAN, *A fast parametric maximum flow algorithm and applications*, SIAM J. Comput., 18 (1989), pp. 30–55.

[6] A. V. GOLDBERG, *Finding a Maximum Density Subgraph*, Technical report UCB CSD 84/171, University of California, Berkeley, CA, 1984.

[7] A. V. GOLDBERG, *private communication*, NEC Research Institute, Princeton, NJ, 1998.

[8] A. V. GOLDBERG AND S. RAO, *Beyond the flow decomposition barrier*, J. ACM, 45 (1998), pp. 753–782.

[9] A. V. GOLDBERG AND R. E. TARJAN, *A new approach to the maximum flow problem*, J. ACM, 35 (1988), pp. 921–940.

[10] D. GUSFIELD AND C. MARTEL, *A fast algorithm for the generalized parametric minimum cut problem and applications*, Algorithmica, 7 (1992), pp. 499–519.

[11] A. Hoffman and T. Rivlin, *When is a team "mathematically" eliminated?*, in Proceedings of the Princeton Symposium on Mathematical Programming, Princeton, NJ, 1967, pp. 391–401.

[12] S. T. McCormick, *Fast algorithms for parametric scheduling come from extensions to parametric maximum flow*, Operations Research, 47 (1999), pp. 744–756.

[13] J. C. Picard and M. Queyranne, *Selected applications of minimum cuts in networks*, Information Systems and Oper. Res., 20 (1982), pp. 394–422.

[14] L. W. Robinson, *Baseball playoff eliminations: an application of linear programming*, Oper. Res. Lett., 10 (1991), pp. 67–74.

[15] B. L. Schwartz, *Possible winners in partially completed tournaments*, SIAM Rev., 8 (1966), pp. 302–308.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 230–236

A LOWER BOUND FOR HEILBRONN'S TRIANGLE PROBLEM IN d DIMENSIONS*

GILL BAREQUET†

Abstract. In this paper we show a lower bound for the generalization of Heilbronn's triangle problem to d dimensions; namely, we show that there exists a set S of n points in the d-dimensional unit cube so that every $d+1$ points of S define a simplex of volume $\Omega(\frac{1}{n^d})$. We also show a constructive *incremental* positioning of n points in a unit 3-cube for which every tetrahedron defined by four of these points has volume $\Omega(\frac{1}{n^4})$.

Key words. Heilbronn's triangle problem, probabilistic method

AMS subject classifications. 05D40, 51M16

PII. S0895480100365859

1. Introduction.

Heilbronn's triangle problem is the following problem.

PROBLEM 1. *Let $\{P_1, P_2, \ldots, P_n\}$ be a set of n points in $[0,1]^2$ such that the minimum of the areas of the triangles $P_iP_jP_k$ (for $1 \le i < j < k \le n$) assumes its maximum possible value $\mathcal{H}_2(n)$. Estimate $\mathcal{H}_2(n)$.*

Heilbronn [5] conjectured that $\mathcal{H}_2(n) = O(\frac{1}{n^2})$, and Erdős set an $\Omega(\frac{1}{n^2})$ lower bound for $\mathcal{H}_2(n)$ by an example [ibid., appendix]. However, Komlós, Pintz, and Szemerédi [4] showed by a rather involved probabilistic construction that $\mathcal{H}_2(n) = \Omega(\frac{\log n}{n^2})$. A simpler construction (which we follow in the present paper) by Alon and Spencer [1] proves a weaker lower bound of $\Omega(\frac{1}{n^2})$.

It is trivial to obtain $\mathcal{H}_2(n) = O(\frac{1}{n})$ (any triangulation of any point set in the unit square admits $O(n)$ triangles). The first nontrivial upper bound, $O(1/(n\sqrt{\log\log n}))$, was given by Roth [5]. Schmidt [9] improved this result 20 years later to $O(1/(n\sqrt{\log n}))$. Soon after that Roth [6, 7] improved the upper bound twice to $O(\frac{1}{n^{1.105\ldots}})$ and to $O(\frac{1}{n^{1.117\ldots}})$.[1] Currently, the best known upper bound, $\mathcal{H}_2(n) = O(\frac{1}{n^{1.142\ldots}})$, is due to Komlós, Pintz, and Szemerédi [3], who refined the proof of [6, 7]. A comprehensive survey of the history of this problem (excluding the results of Komlós, Pintz, and Szemeredi) is given by Roth in [8].

We are not aware of any generalization of Heilbronn's problem to higher dimensions. In this paper we use a probabilistic argument, as well as a specific example, to show that $\mathcal{H}_d(n) = \Omega(\frac{1}{n^d})$, where $\mathcal{H}_d(n)$ is the d-dimensional analogue of $\mathcal{H}_2(n)$. In particular, $\mathcal{H}_3(n) = \Omega(\frac{1}{n^3})$. We also give a constructive incremental method for positioning n points in a unit 3-cube so that all the tetrahedra defined by quadruples of the points have volume $\Omega(\frac{1}{n^4})$.

*Received by the editors January 5, 2000; accepted for publication (in revised form) December 19, 2000; published electronically April 3, 2001. Work on this paper has been supported by the U.S. Army Research Office under grant DAAH04-96-1-0013. A preliminary version of this paper appeared in *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, Baltimore, MD, 1999, pp. 76–81.

http://www.siam.org/journals/sidma/14-2/36585.html

†Faculty of Computer Science, The Technion—Israel Institute of Technology, Haifa 32000, Israel (barequet@cs.technion.ac.il). This work was done while the author was affiliated with the Center for Geometric Computing, Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218.

[1] The exponents of n in these bounds are the smaller roots $\mu = (17 - \sqrt{65})/8$ and $\nu = 2 - \sqrt{0.8}$ of the equations $4\mu^2 - 17\mu + 14 = 0$ and $5\nu^2 - 20\nu + 16 = 0$, respectively.

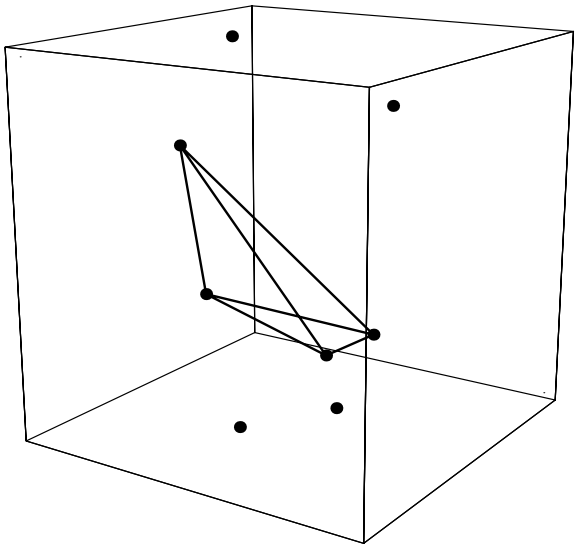


FIG. 1. *Heilbronn's problem in three dimensions.*

The paper is organized as follows. In section 2 we give the incremental construction in three dimensions. In section 3 we use a probabilistic construction to improve the lower bound for three dimensions and use the same method for proving a lower bound for any fixed dimension. In section 4 we show the same lower bound by an example (a generalization of Erdős's planar construction) and argue why the much more complex probabilistic construction is still of interest. Section 5 contains some concluding remarks.

2. An incremental construction in three dimensions. The generalization of Heilbronn's triangle problem to three dimensions is straightforward.

PROBLEM 2. *Let* $\{P_1, P_2, \ldots, P_n\}$ *be a set of* n *points in* $[0,1]^3$ *such that the minimum of the volumes of the tetrahedra* $P_iP_jP_kP_l$ *(for* $1 \le i < j < k < l \le n$*) assumes its maximum possible value* $\mathcal{H}_3(n)$*. Estimate* $\mathcal{H}_3(n)$*.*

Figure 1 shows eight points in the three-dimensional unit cube and the tetrahedron of minimum volume defined by four of these points.

We first present an incremental construction for positioning n points in the unit 3-cube, in which we position one point at a time while maintaining the invariant that after positioning the vth point, no two, three, or four of the already positioned v points are "too close together." In particular, no four points define a tetrahedron with volume less than $\frac{1}{c\,n^4}$ (for some constant $c > 0$). After each step of the construction the already v positioned points induce portions of the unit cube in which the next point P_{v+1} cannot be positioned. The task is to position P_{v+1} outside the union of all these "forbidden zones." We show that this is always possible. Moreover, the $(v+1)$st point can be positioned *anywhere* outside the forbidden zones while still not spoiling the positioning of the remaining points. This is a generalization of Schmidt's method [9] for the analog two-dimensional problem; we comment in section 2.2 on the use of his idea and mention the caveat in the generalization to three dimensions. The construction gives us a constructive lower bound of $\Omega(\frac{1}{n^4})$ which is inferior to the $\Omega(\frac{1}{n^3})$ bound shown in section 3.2. However, it solves a harder problem, the *on-line*

Heilbronn's triangle problem, in which the number of points is not known in advance.

2.1. The construction. Denote by d_{ij} the distance between the points P_i and P_j, by Δ_{ijk} the area of the triangle defined by P_i, P_j, and P_k, and by V_{ijkl} the volume of the tetrahedron defined by P_i, P_j, P_k, and P_l (for $1 \le i < j < k < l \le n$). When we position the vth point we make sure that

(i) no two points are too close: for all $1 \le i < j \le v$ we have $d_{ij} > \frac{1}{a\,n^{1/3}}$;

(ii) no three points are too close to being collinear: for all $1 \le i < j < k \le v$ we have $\Delta_{ijk} > \frac{1}{b\,n}$;

(iii) no four points are too close to being coplanar: for all $1 \le i < j < k < l \le v$ we have $V_{ijkl} > \frac{1}{c\,n^4}$.

The constants a, b, and c are defined later. The first two conditions ensure (as we detail below) the satisfaction of the third condition, which implies the sought lower bound: after positioning the nth point, all the tetrahedra defined by quadruples of points have volume $\Omega(\frac{1}{n^4})$.

The main idea is that after positioning the vth point (for $1 \le v \le n$), the cumulative volume of all the forbidden zones induced by the three conditions is less than 1. Therefore the construction cannot break down before it is complete. We do not exploit the fact that the forbidden zones may, and in fact, must overlap. Thus we obtain by this incremental construction a weaker lower bound than that of the nonconstructive method of section 3.2. Let us then compute the volumes of the forbidden zones after the vth step of the construction (throughout the computations we ignore, as we may, low-order powers of n):

(i) Each of the first v points induces a forbidden ball of radius $\frac{1}{a\,n^{1/3}}$ (see Figure 2(a)). The total forbidden volume of the v balls is

$$\sum_{i=1}^{v} \frac{4}{3}\pi \left(\frac{1}{a\,n^{1/3}}\right)^3 = \frac{4\pi}{3a^3}\left(\frac{v}{n}\right) \le \frac{4\pi}{3a^3}.$$

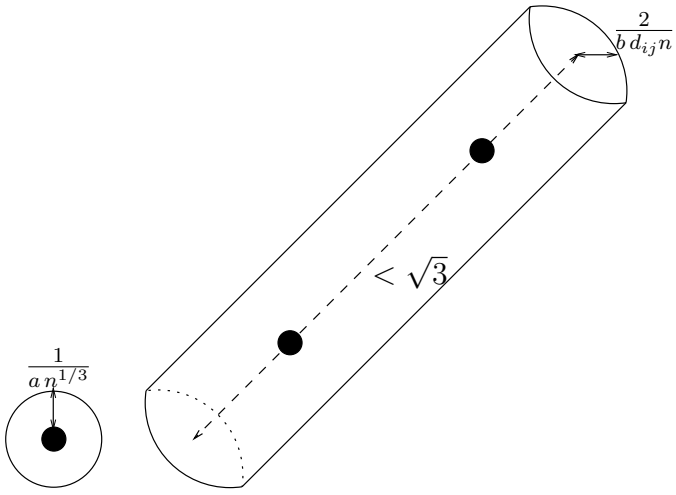
(ii) Each pair of points P_i and P_j induces a forbidden cylinder of radius $\frac{2}{b\,d_{ij}\,n}$ and of height at most $\sqrt{3}$ (see Figure 2(b)). The total forbidden volume of the $\binom{v}{2}$ cylinders is thus at most

$$\sum_{1 \le i < j \le v} \sqrt{3}\pi \left(\frac{2}{b\,d_{ij}n}\right)^2 = \frac{4\sqrt{3}\pi}{b^2n^2} \sum_{1 \le i < j \le v} \frac{1}{d_{ij}^2}. \tag{1}$$

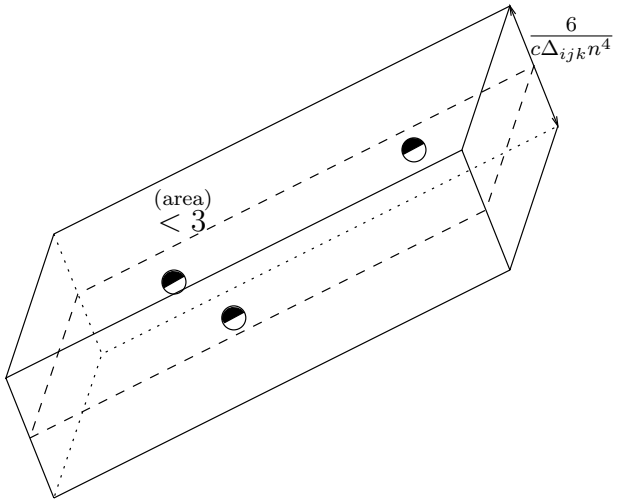
To bound $\sum_{1 \le i < j \le v} \frac{1}{d_{ij}^2}$, we fix P_i and sum over P_j using a volume-counting argument

$$\sum_{\substack{1 \le j \le v \\ j \ne i}} \frac{1}{d_{ij}^2} \le \sum_{s=1}^{\sqrt{3}a\,n^{1/3}} \frac{N_s a^2 n^{2/3}}{s^2}, \tag{2}$$

where N_s is the number of points (out of the first v points) that lie in the spherical shell centered at P_i with inner radius $\frac{s}{a\,n^{1/3}}$ and outer radius $\frac{s+1}{a\,n^{1/3}}$. Since $d_{ij} > \frac{1}{a\,n^{1/3}}$, we have $N_s < \frac{8s^2}{3}$ (this follows by an argument of packing spheres—actually, at least half of each sphere—of volume at least $\frac{4\pi}{3a^3n}$ within a shell whose volume is $\frac{4\pi(s+1)^3}{3a^3n} - \frac{4\pi s^3}{3a^3n} < \frac{16\pi s^2}{3a^3n}$). Hence the sum in (2) is less than $8\sqrt{3}a^3n$. Summing up for all P_i and substituting this in (1), we conclude that the total volume of the forbidden cylinders is less than $\frac{96\pi a^3 v}{b^2 n} \le \frac{96\pi a^3}{b^2}$.



(a) Ball (b) Cylinder



(c) Box

FIG. 2. *Forbidden zones.*

(iii) Each triple of points P_i, P_j, and P_k induces a forbidden box of height $\frac{6}{c\Delta_{ijk}n^4}$ and with base-area at most $\frac{3\pi}{4}$ (see Figure 2(c)). The total forbidden volume of the $\binom{v}{3}$ boxes is thus at most

$$\sum_{1\le i<j<k\le v} \left(\frac{3\pi}{4}\cdot\frac{6}{c\Delta_{ijk}n^4}\right) < \frac{9\pi}{2c\,n^4}\cdot\frac{v^3}{6}\cdot(b\,n) = \frac{3\pi b}{4c}\left(\frac{v}{n}\right)^3 \le \frac{3\pi b}{4c}.$$

After positioning each point, the total volume of the forbidden zones is therefore less than $\frac{4\pi}{3a^3} + \frac{96\pi a^3}{b^2} + \frac{3\pi b}{4c}$. It is trivial to set values to a, b, and c so that the forbidden volume is always less than 1.

In section 3.2 we establish a probabilistic construction which yields a better lower bound for $\mathcal{H}_3(n)$.

2.2. A comment on the construction. Schmidt has probably used a packing argument (similar to the one we use for upper bounding the total volume of the forbidden cylinders) for his incremental positioning of points in the planar version of the problem. He mentions without proof (see [9, last line of p. 548 and first line of p. 549]), as this seems trivial, that if $d_{ij} \geq \frac{1}{2\sqrt{n}}$, then $\sum_{1 \leq i < j \leq v} \frac{1}{d_{ij}} < c\sqrt{n}\, v^{3/2}$ (for some constant $c > 0$). This is easily obtained by a planar packing argument.

We have used this idea for obtaining an efficient upper bound for the total volume of forbidden cylinders defined by triples of points. However, we were not able to use a similar idea for bounding the total volume of the forbidden boxes defined by quadruples of points. (This type of forbidden zone has no counterpart in the planar version of the problem.)

3. A probabilistic construction in $d \geq 3$ dimensions.

3.1. A probabilistic lemma. We generalize a probabilistic argument of Alon and Spencer [1, p. 30]. Let $H(P_1, P_2, \ldots, P_{d+1})$ be a mapping from $(d+1)$-tuples of points $P_1, P_2, \ldots, P_{d+1} \in [0,1]^d$ to $\Re^+ \cup \{0\}$.

LEMMA 3.1. *If there exist constants $c_1 > 0$, c_2 such that*

$$\text{Prob}[H(P_1, P_2, \ldots, P_{d+1}) \leq \varepsilon] \leq c_1 \varepsilon^{c_2},$$

where $P_1, P_2, \ldots, P_{d+1}$ are chosen randomly, uniformly, and independently in $[0,1]^d$, then there exists a set S of n points in $[0,1]^d$ and a constant $c_3 > 0$ such that $\min_{P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}} \in S} H(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}) > c_3 n^{-\frac{d}{c_2}}$.

Proof (see [1]). Let $P_1, P_2, \ldots, P_{2n}$ be a set of $2n$ points selected randomly, uniformly, and independently in $[0,1]^d$. Set $c_3 = (\frac{(d+1)!}{2^{d+1} c_1})^{1/c_2}$. Let X denote the number of $(d+1)$-tuples $P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}$ for which $H(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}) \leq c_3 n^{-\frac{d}{c_2}}$. Then,

$$E[X] \leq \binom{2n}{d+1} c_1 (c_3 n^{-\frac{d}{c_2}})^{c_2} < \frac{(2n)^{d+1}}{(d+1)!} \cdot \frac{(d+1)!}{2^{d+1} n^d} = n.$$

Therefore there exists a specific set of $2n$ points with fewer than n $(d+1)$-tuples $P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}$ for which $H(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}) \leq c_3 n^{-\frac{d}{c_2}}$. Remove one point from the set from each such $(d+1)$-tuple. (The same point may be deleted more than once but this only helps.) This leaves at least n points and now all $(d+1)$-tuples $P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}$ satisfy $H(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}) > c_3 n^{-\frac{d}{c_2}}$. □

Alon and Spencer [1, p. 30] prove a special case of Lemma 3.1 in which $c_2 = 1$ and $d = 2$ and use it for showing that $\mathcal{H}_2(n) = \Omega(\frac{1}{n^2})$. Note that the proof of the lemma does not use the fact that the points are located in a d-dimensional space. The parameter d plays a role only in the number of arguments of the function H.

3.2. The construction in three dimensions. We now show the following theorem.

THEOREM 3.2. $\mathcal{H}_3(n) = \Omega(\frac{1}{n^3})$.

Proof. Let $H_3(P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4})$ be the volume of the tetrahedron defined by P_{i_1}, P_{i_2}, P_{i_3}, and P_{i_4}. We first bound $\text{Prob}[H_3(P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4}) \leq \varepsilon]$ (generalizing the method of [1, p. 30]). Let x be the distance from P_{i_1} to P_{i_2}. Then, $\text{Prob}[b \leq x \leq b + db] \leq d(\frac{4}{3}\pi b^3) = 4\pi b^2\, db$ (the difference between the volumes of the corresponding balls). Let y be the distance from P_{i_3} to the line $\ell_{i_1 i_2}$ that passes through P_{i_1} and P_{i_2}. Then, $\text{Prob}[c \leq y \leq c + dc] \leq d(\pi b c^2) = 2\pi b c\, dc$ (the difference between the volumes of the corresponding cylinders). Given P_{i_1} and P_{i_2} at distance b, and P_{i_3} at distance

c from $\ell_{i_1 i_2}$, the altitude h from P_{i_4} to the plane defined by P_{i_1}, P_{i_2}, and P_{i_3} satisfies $\frac{bch}{6} \le \varepsilon$, i.e., $h \le \frac{6\varepsilon}{bc}$. Thus P_{i_4} must lie within a box of height $\frac{12\varepsilon}{bc}$ and of base-area at most 3. This occurs with probability at most $\frac{36\varepsilon}{bc}$. Since $0 \le b, c \le \sqrt{3}$, $\mathrm{Prob}[H_3(P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4}) \le \varepsilon] \le \int_0^{\sqrt{3}} \int_0^{\sqrt{3}} (4\pi b^2)(2\pi bc)(\frac{36\varepsilon}{bc}) dc\, db = 864\pi^2\varepsilon$. Now apply Lemma 3.1 (with $c_1 = 864\pi^2$, $c_2 = 1$, and $d = 3$) and conclude that there exists a set $S \subset [0,1]^3$ of n points for which $\min_{P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4} \in S} H_3(P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4}) > \frac{1}{5685\, n^3}$. ∎

3.3. The construction in d dimensions. Using the method of Theorem 3.2 we are able to show our main result.

THEOREM 3.3. $\mathcal{H}_d(n) = \Omega(\frac{1}{n^d})$.

Proof. We closely follow the previous proof. Let $H_d(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}})$ be the volume of the d-dimensional simplex defined by $P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}$ and let b_k be the distance from P_{i_k} to E_{k-1}, the $(k-2)$-dimensional flat defined by $P_{i_1}, P_{i_2}, \ldots, P_{i_{k-1}}$. In considering P_{i_k} (for $2 \le k \le d$) and estimating the probability that its distance x_k from E_{k-1} is in an infinitesimal range, we always have

$$\mathrm{Prob}[b_k \le x_k \le b_k + db_k] \le p_k(b_2, \ldots, b_k)\, db_k,$$

where $p_k(b_2, \ldots, b_k)$ is a polynomial of the form $c_{0,k} b_2^{c_{2,k}} \cdots b_k^{c_{k,k}}$ ($c_{i,k}$ are all positive constants, $i = 0, 2, 3, \ldots, k$). We apply the condition $H_d(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}) \le \varepsilon$ only when we consider $P_{i_{d+1}}$. Then we conclude that the respective event occurs with probability at most $\frac{C_{d+1}\varepsilon}{b_2 \cdots b_k}$ (C_{d+1} is also a positive constant). Since $0 \le b_k \le \sqrt{d}$ (for $2 \le k \le d$),

$$\begin{aligned}
&\mathrm{Prob}[H_d(P_{i_1}, P_{i_2}, \ldots, P_{i_{d+1}}) \le \varepsilon] \\
&\le \underbrace{\int_0^{\sqrt{d}} \int_0^{\sqrt{d}} \cdots \int_0^{\sqrt{d}}}_{d-1 \text{ integrations}} p_2(b_2) p_3(b_2, b_3) \cdots p_d(b_2, b_3, \ldots, b_d) \left(\frac{C_{d+1}\varepsilon}{b_2 \cdots b_k} \right) db_k \cdots db_3\, db_2 \\
&= \mathcal{C}_d \varepsilon,
\end{aligned}$$

where the constant $\mathcal{C}_d > 0$ depends only on d. Now apply Lemma 3.1 (with $c_1 = \mathcal{C}_d$ and $c_2 = 1$) and the claim follows. ∎

4. An alternative proof. In this section we give an alternative proof to Theorem 3.3 by generalizing Erdős's example [5, appendix] that shows that $\mathcal{H}_2(n) = \Omega(\frac{1}{n^2})$.

Assume that n is prime. (This involves no loss of generality since the ratio of consecutive primes is asymptotically 1.) Put n points on the d-dimensional moment curve (modulo n), so that their coordinates are integer multiples of $1/n$. That is, take the ith point (for $0 \le i \le n-1$) to be $P_i = (i/n, (i^2 \bmod n)/n, \ldots, (i^d \bmod n)/n)$. Now choose any $d+1$ points $P_{i_1}, P_{i_2}, \ldots, P_{d+1}$. The volume of the d-dimensional simplex defined by these points is a Vandermonde determinant of order $d+1$:

$$\frac{1}{d!\, n^d} \left(\left| \begin{array}{ccccc} 1 & i_1 & i_1^2 & \cdots & i_1^d \\ 1 & i_2 & i_2^2 & \cdots & i_2^d \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ 1 & i_{d+1} & i_{d+1}^2 & \cdots & i_{d+1}^d \end{array} \right| \bmod n \right) = \frac{(\prod_{1 \le j < k \le d+1} (i_k - i_j)) \bmod n}{d!\, n^d}.$$

This term cannot vanish; hence $\mathcal{H}_d(n) \ge \frac{1}{d!\, n^d}$.

Although the moment-curve example is much simpler than the probabilistic construction given in the previous section, the latter is more general and can be applied for other versions of the problem:

1. The probabilistic construction also holds for the more general case in which one is asked to position n points in a nonconvex shape of unit volume, or, in fact, to any shape, possibly not simply connected or not connected at all, of unit volume. The moment-curve example can be applied (with the appropriate scaling) to any convex unit-volume shape, since such shape always circumscribes a box of constant volume,[2] but it cannot be used for concave shapes.

2. The probabilistic construction can be applied for other measures of $d+1$ points in d dimensions in addition to the volume of the polytope defined by the points. For example, let $L_3(P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4})$ be the sum of lengths of the edges of the tetrahedron defined by the points P_{i_1}, P_{i_2}, P_{i_3}, and P_{i_4}, and let $\mathcal{L}_3(n)$ be the minimum of L_3 taken over all quadruples of points of S spread in the unit cube, in the distribution of a set S of n points in which this value assumes its maximum. The moment-curve example sets $\mathcal{L}_3(n) = \Omega(\frac{1}{n})$. An easy calculation shows that $\text{Prob}[L_3(P_{i_1}, P_{i_2}, P_{i_3}, P_{i_4}) \leq \varepsilon] = O(\varepsilon^7)$; hence the application of Lemma 3.1 yields a better lower bound of $\Omega(\frac{1}{n^{3/7}})$. In fact, spreading the points on the vertices of a full regular grid gives the optimum distribution for this measure: $\mathcal{L}_3(n) = \Theta(\frac{1}{n^{1/3}})$. (The matching upper bound is shown by a simple volume-counting argument: a full grid whose step is $(\frac{4}{n})^{1/3}$ must have a cell that contains at least four points.)

5. Conclusion. In this paper we give a lower bound for the d-dimensional version of Heilbronn's triangle problem. We believe that as in the planar case the lower bound can still be improved. We also show a constructive (but weaker) lower bound in three dimensions.

Possible further research directions include the following:

1. obtaining an upper bound for $\mathcal{H}_d(n)$;
2. optimizing other measures of tetrahedra (or simplices) in d dimensions;
3. developing algorithms that, given values of n and d, and a specific measure of a d-dimensional simplex, find an optimal positioning of n points in the d-dimensional unit cube.

REFERENCES

[1] N. Alon and J. H. Spencer, *The Probabilistic Method*, John Wiley, New York, 1992.

[2] G. Barequet and S. Har-Peled, *Efficiently approximating the minimum-volume bounding box of a point set in three dimensions*, J. Algorithms, 38 (2001), pp. 91–109.

[3] J. Komlós, J. Pintz, and E. Szemerédi, *On Heilbronn's triangle problem*, J. London Math. Soc. (2), 24 (1981), pp. 385–396.

[4] J. Komlós, J. Pintz, and E. Szemerédi, *A lower bound for Heilbronn's problem*, J. London Math. Soc. (2), 25 (1982), pp. 13–24.

[5] K. F. Roth, *On a problem of Heilbronn*, Proc. London Math. Soc., 26 (1951), pp. 198–204.

[6] K. F. Roth, *On a problem of Heilbronn* II, Proc. London Math. Soc. (3), 25 (1972), pp. 193–212.

[7] K. F. Roth, *On a problem of Heilbronn* III, Proc. London Math. Soc. (3), 25 (1972), pp. 543–549.

[8] K. F. Roth, *Developments in Heilbronn's triangle problem*, Adv. Math., 22 (1976), pp. 364–385.

[9] W. M. Schmidt, *On a problem of Heilbronn*, J. London Math. Soc. (2), 4 (1971), pp. 545–550.

[2] See, e.g., Lemma 3.4 of [2].

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 237–239

A NOTE ON ITERATING AN α-ARY GRAY CODE*

CHI-JEN LU† AND SHI-CHUN TSAI‡

Abstract. In this note we consider the number of distinct α-ary codes produced by repeatedly applying the Gray code mapping of Sharma and Khanna [*Inform. Sci.*, 15 (1978), pp. 31–43]. This number was derived before by Lichtner [*SIAM J. Discrete Math.*, 11 (1998), pp. 381–386], and we give an alternative proof here. Our key observation is a simple connection between this number and the period of binomial coefficients modulo α. Then the result follows immediately from a known periodic property of binomial coefficients modulo α [*Fibonacci Quart.*, 27 (1989), pp. 64–79; *SIAM J. Discrete Math.*, 9 (1996), pp. 55–62; *Ann. Univ. Mariae Curie-Sklodowska Sect. A*, 10 (1956), pp. 37–47].

Key words. gray code, binomial coefficient

AMS subject classifications. 68Q25, 68R01

PII. S0895480100367688

1. Introduction.

Let ℓ and α be positive integers greater than one. An α-ary Gray code of dimension ℓ is a sequence of α^ℓ distinct α-ary strings of length ℓ such that any two adjacent strings differ in exactly one position. Therefore a Gray code corresponds to a bijection, mapping integers between 0 and $\alpha^\ell - 1$ to α-ary strings of length ℓ. When viewing such integers in base α with ℓ digits, the mapping can be seen as a permutation among α-ary strings of length ℓ. Sharma and Khanna [4] defined one such mapping, denoted by $\mathcal{K}$, in the following way.

DEFINITION 1.1 (see [4]). *$\mathcal{K}(x_1x_2\cdots x_\ell) = g_1g_2\cdots g_\ell$, where*

$$g_i = \begin{cases} x_1 & \text{if } i = 1, \\ x_i - x_{i-1} \pmod{\alpha} & 1 < i \le \ell. \end{cases}$$

Equivalently, $\mathcal{K}^{-1}(g_1g_2\cdots g_\ell) = x_1x_2\cdots x_\ell$, where

$$x_i = \begin{cases} g_1 & \text{if } i = 1, \\ g_i + x_{i-1} \pmod{\alpha} & 1 < i \le \ell. \end{cases}$$

Culberson [1] and Lichtner [3] studied the number of distinct sets of codes that can be generated by repeated applications of the mapping $\mathcal{K}$. This is the same question as determining the order of the mapping $\mathcal{K}$, defined in the following.

DEFINITION 1.2. *Let $\mathcal{K}^0 = \mathcal{I}$, the identity mapping, i.e., $\mathcal{I}(x) = x$ for all x. For $i \ge 0$, let $\mathcal{K}^{i+1} = \mathcal{K} \circ \mathcal{K}^i$, i.e., $\mathcal{K}^{i+1}(x) = \mathcal{K}(\mathcal{K}^i(x))$. The order of $\mathcal{K}$ is the smallest positive integer m such that $\mathcal{K}^m = \mathcal{I}$. Similarly, we can define $\mathcal{K}^{-(i+1)} = \mathcal{K}^{-1} \circ \mathcal{K}^{-i}$ for $i \ge 0$.*

Note that the order of $\mathcal{K}$ is the same as the order of $\mathcal{K}^{-1}$, and we will work on $\mathcal{K}^{-1}$ instead. Lichtner [3] derived an explicit formula for the order of $\mathcal{K}^{-1}$. In this note,

*Received by the editors February 1, 2000; accepted for publication (in revised form) January 9, 2001; published electronically April 3, 2001.
http://www.siam.org/journals/sidma/14-2/36768.html

†Institute of Information Science, Academia Sinica, Nankang 11529, Taipei, Taiwan (cjlu@iis.sinica.edu.tw).

‡Department of Information Management, National Chi-Nan University, 1 University Road, Pu-Li, Nan-Tou 545, Taiwan (tsai@csie.ncnu.edu.tw). The work of this author was supported in part by the National Science Council of Taiwan under contract NSC 89-2213-E-260-009.

we give an alternative proof for Lichtner's result. The key idea is an observation on a simple connection between the order of $\mathcal{K}^{-1}$ and the period of binomial coefficients modulo α. Then the result follows immediately from a known periodic property of binomial coefficients modulo α [2, 5, 6].

In section 2, we state some relevant results. In section 3, we prove our main technical lemma. Finally, we conclude with some remarks in section 4.

2. Results. Let $x = x_1x_2\cdots x_\ell$. Define $x^i = \mathcal{K}^{-i}(x)$ for $i \geq 0$ with $x^0 = \mathcal{K}^0(x) = x$. Let x^i_j denote the jth digit of x^i. Here are some simple facts.

LEMMA 2.1 (see [3]).

1. $x^i_j \equiv x^i_{j-1} + x^{i-1}_j \pmod{\alpha}$, *where* $i \geq 1$ *and* $1 < j \leq \ell$.
2. $x^i_j \equiv \sum_{k=1}^{j} \binom{i+j-k-1}{j-k} x_k \pmod{\alpha}$, *where* $i \geq 1$ *and* $1 \leq j \leq \ell$.
3. *If* $\binom{i+j-2}{j-1} \equiv 0 \pmod{\alpha}$, *for* $2 \leq j \leq \ell$, *then* $x^i = x$.
4. *Let* $x = x_1x_2\cdots x_\ell$ *be an* α*-ary string such that* $x_1 \not\equiv 0 \pmod{\alpha}$ *and* $GCD(x_1, \alpha) = 1$. *If there is a* j, $2 \leq j \leq \ell$, *such that* $\binom{i+j-2}{j-1} \not\equiv 0 \pmod{\alpha}$, *then* $x^i \neq x$.

Lemma 2.1(1–2) follows from the definition of $\mathcal{K}$ and by induction. Lemma 2.1(3–4) follows from Lemma 2.1(1–2).

From now on, we assume that α has the form of

$$\alpha = p_1^{n_1} \cdots p_q^{n_q},$$

where p_i's are distinct primes and n_i's are positive integers. Lichtner generalized Culberson's result [1] and proved the following theorem.

THEOREM 2.2 (see [3]). *Let* $\ell > 1$ *and let* $e_i = \lfloor \log_{p_i}(\ell - 1) \rfloor$ *for* $1 \leq i \leq q$. *Then the order of* $\mathcal{K}^{-1}$ *is exactly* $L = \prod_{i=1}^{q} p_i^{n_i+e_i}$.

We will give a more straightforward proof of Lichtner's theorem above, given the following known result on the period of the function $f_{\alpha,k}$, defined as

$$f_{\alpha,k}(x) = \binom{x}{k} \pmod{\alpha}.$$

LEMMA 2.3 (see [2, 5, 6]). *Let* k *be a positive integer and let* $d_i = \lfloor \log_{p_i} k \rfloor$ *for* $1 \leq i \leq q$. *Then the period of the function* $f_{\alpha,k}$ *is exactly* $\prod_{i=1}^{q} p_i^{n_i+d_i}$.

COROLLARY 2.4. *For any positive integers* k_1, k_2 *with* $k_1 \leq k_2$, *the period of* f_{α,k_1} *divides the period of* f_{α,k_2}.

Our main technical contribution is the following simple connection.

LEMMA 2.5. *The order of* $\mathcal{K}^{-1}$ *is the same as the period of the function* $f_{\alpha,\ell-1}$.

Lemmas 2.3 and 2.5 together immediately imply Theorem 2.2. It remains to prove Lemma 2.5.

3. Proof of Lemma 2.5. Let $x = x_1x_2\cdots x_\ell$ be an arbitrary α-ary string and let $x^i = \mathcal{K}^{-i}(x)$. Item 2 of Lemma 2.1 states that

$$x^i_1 \equiv \binom{i-1}{0} x_1 \pmod{\alpha},$$

$$x^i_2 \equiv \binom{i}{1} x_1 + \binom{i-1}{0} x_2 \pmod{\alpha},$$

$$x^i_3 \equiv \binom{i+1}{2} x_1 + \binom{i}{1} x_2 + \binom{i-1}{0} x_3 \pmod{\alpha},$$

$$\vdots$$
$$x_\ell^i \equiv \binom{i+\ell-2}{\ell-1}x_1 + \binom{i+\ell-3}{\ell-2}x_2 + \cdots + \binom{i}{1}x_{\ell-1} + \binom{i-1}{0}x_\ell \pmod{\alpha}.$$

Observe that there are ℓ different binomial coefficients above, so $\mathcal{K}^{-i}$ is completely determined by the vector $c^i = c_1^i c_2^i \cdots c_\ell^i$, where

$$c_j^i = \binom{i+j-2}{j-1} \pmod{\alpha}.$$

We use the convention that $\binom{a}{0} = 1$ for any integer a. Note that $c^0 = 100\cdots0$. As i increases, the vector c^i changes in a periodic fashion due to the periodic behavior of its components. Recall that $L = \prod_{i=1}^{q} p_i^{n_i+e_i}$. We show that the period is L and the cycle occurs exactly when the vector becomes $100\cdots0$ again.

Clearly $c_1^L = 1$. From Lemma 2.3 and Corollary 2.4, we know that for $1 < j \le \ell$, $c_j^L = \binom{L+j-2}{j-1} \pmod{\alpha} = \binom{0+j-2}{j-1} \pmod{\alpha} = 0$. Therefore $c^L = 100\cdots0 = c^0$. Now, as $c_1^i = 1$ and for $j > 1$,

$$c_j^i = c_{j-1}^i + c_j^{i-1} \pmod{\alpha},$$

we see that the vector c^i is completely determined by the vector c^{i-1}. Therefore the same pattern repeats again from c^L. The cycle could not be shorter since $c_\ell^i = f_{\alpha,\ell-1}(i+\ell-2)$, considered as a function of i, has a period of exactly L. Therefore L is the smallest positive integer i such that $c^i = c^0$.

Since $c^L = 100\cdots0$, $\mathcal{K}^{-L}$ is the identity mapping. On the other hand, for any positive $i < L$, $c^i \neq 100\cdots0$. Item 4 of Lemma 2.1 tells us that $\mathcal{K}^{-i}x \neq x$ for any x with $x_1 = 1$ and thus $\mathcal{K}^{-i} \neq \mathcal{I}$. Therefore the order of $\mathcal{K}^{-1}$ is exactly L.

4. Conclusion and remarks. By using the periodic property of binomial coefficients, we give a proof on the order of the mapping $\mathcal{K}$. Lichtner's proof could have been more straightforward if he had known the periodic property of binomial coefficients. Such a periodic property has also been applied to results in computational complexity [5]. We expect that this nice property should have more applications to be discovered.

REFERENCES

[1] J. Culberson, *Mutation-crossover isomorphisms and the construction of discriminating functions*, Evolutionary Comput., 2 (1995), pp. 279–311.

[2] Y.H. Kwong, *Minimum periods of binomial coefficients modulo M*, Fibonacci Quart., 27 (1989), pp. 64–79.

[3] J. Lichtner, *Iterating an α-ary Gray code*, SIAM J. Discrete Math., 11 (1998), pp. 381–386.

[4] B.D. Sharma and R.K. Khanna, *On m-ary Gray codes*, Inform. Sci., 15 (1978), pp. 31–43.

[5] S-C. Tsai, *Lower bounds on representing boolean functions as polynomials in Z_m*, SIAM J. Discrete Math., 9 (1996), pp. 55–62.

[6] S. Zabek, *Sur la periodicite modulo m des suits de nombres $\binom{n}{k}$*, Ann. Univ. Mariae Curie-Sklodowska Sect. A, 10 (1956), pp. 37–47.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 240–245

THE NUMBER OF IRREDUCIBLE POLYNOMIALS AND LYNDON WORDS WITH GIVEN TRACE*

F. RUSKEY†, C. R. MIERS‡, AND J. SAWADA†

Abstract. The *trace* of a degree n polynomial $f(x)$ over $GF(q)$ is the coefficient of x^{n-1}. Carlitz [*Proc. Amer. Math. Soc.*, 3 (1952), pp. 693–700] obtained an expression $I_q(n,t)$ for the number of monic irreducible polynomials over $GF(q)$ of degree n and trace t. Using a different approach, we derive a simple explicit expression for $I_q(n,t)$. If $t > 0$, $I_q(n,t) = (\sum \mu(d)q^{n/d})/(qn)$, where the sum is over all divisors d of n which are relatively prime to q. This same approach is used to count $L_q(n,t)$, the number of q-ary Lyndon words whose characters sum to t mod q. This number is given by $L_q(n,t) = (\sum \gcd(d,q)\mu(d)q^{n/d})/(qn)$, where the sum is over all divisors d of n for which $\gcd(d,q)|t$. Both results rely on a new form of Möbius inversion.

Key words. irreducible polynomial, trace, finite field, Lyndon word, Möbius inversion

AMS subject classifications. 05T06, 11T06

PII. S0895480100368050

1. Introduction.

The *trace* of a degree n polynomial $f(x)$ over $GF(q)$ is the coefficient of x^{n-1}. It is well known that the number of degree n irreducible polynomials over $GF(q)$ is given by

$$I_q(n) = \frac{1}{n}\sum_{d|n}\mu(d)q^{n/d}, \tag{1.1}$$

where $\mu(d)$ is the Möbius function. Less well known is the formula

$$I_2(n,1) = \frac{1}{2n}\sum_{\substack{d|n\\ d \text{ odd}}}\mu(d)2^{n/d}, \tag{1.2}$$

which is the number of degree n irreducible polynomials over $GF(2)$ with trace 1 (this can be inferred from results in Jungnickel [3, section 2.7]). One purpose of this paper is to refine (1.1) and (1.2) by enumerating the irreducible degree n polynomials over $GF(q)$ with a given trace. Carlitz [1] also solved this problem, arriving via a different technique at an expression that is different but equivalent to the one given below. Our version of the result is stated in Theorem 1.1.

THEOREM 1.1. *Let q be a power of prime p. The number of irreducible polynomials of degree $n > 0$ over $GF(q)$ with a given nonzero trace t is*

$$I_q(n,t) = \frac{1}{qn}\sum_{\substack{d|n\\ p\nmid d}}\mu(d)q^{n/d}. \tag{1.3}$$

*Received by the editors January 10, 2000; accepted for publication (in revised form) January 2, 2001; published electronically April 3, 2001.
http://www.siam.org/journals/sidma/14-2/36805.html

†Department of Computer Science, University of Victoria, EOW 348, 3800 Finnerty Road, P.O. Box 3055–MS 7209, Victoria, BC V8W 3P6, Canada (fruskey@csr.uvic.ca, jsawada@csr.uvic.ca).
The research of these authors was supported in part by NSERC.
‡Department of Mathematics and Statistics, University of Victoria, Clearihue Building, Room D268, 3800 Finnerty Road, Victoria, BC V8P 5C2, Canada (crmiers@math.uvic.ca).

Note that the expression on the right-hand side of (1.3) is independent of t and that $I_q(n,0)$ can be obtained by subtracting

$$I_q(n,0) = I_q(n) - (q-1)I_q(n,1).$$

A Lyndon word is the lexicographically smallest rotation of an aperiodic string. If $L_q(n)$ denotes the number of q-ary Lyndon words of length n, then it is well known that $L_q(n) = I_q(n)$. The *trace* of a Lyndon word is the sum of its characters mod q. Let $L_q(n,t)$ denote the number of Lyndon words of trace t. The second purpose of this paper is to obtain an explicit formula for $L_q(n,t)$. This result is stated in Theorem 1.2.

THEOREM 1.2. *For all integers $n > 0$, $q > 1$, and $t \in \{0, 1, \ldots, q-1\}$,*

$$L_q(n,t) = \frac{1}{qn} \sum_{\substack{d|n \\ \gcd(d,q)|t}} \gcd(d,q)\mu(d)q^{n/d}.$$

Note that $I_q(n,t) = L_q(n,s)$ whenever $t \neq 0$ and $\gcd(n,s) = 1$. In order to prove Theorems 1.1 and 1.2 we need a new form of Möbius inversion. This is presented in the next section.

2. A generalized Möbius inversion formula. The defining property of the Möbius functions is

$$\sum_{d|n} \mu(d) \;=\; [\![n = 1]\!], \tag{2.1}$$

where $[\![P]\!]$ for proposition P represents the "Iversonian convention": $[\![P]\!]$ has value 1 if P is true and value 0 if P is false (see [4, p. 24]).

DEFINITION 2.1. *Let $\mathcal{R}$ be a set, $\mathbb{N} = \{1, 2, 3, \ldots\}$, and let $\{X(d,t)\}_{t\in\mathcal{R}, d\in\mathbb{N}}$ be a family of subsets of $\mathcal{R}$. We say that $\{X(d,t)\}_{t\in\mathcal{R}, d\in\mathbb{N}}$ is* recombinant *if*

(i) $X(1,t) = \{t\}$ *for all $t \in \mathcal{R}$ and*

(ii) $\{e' \in X(d',e) : e \in X(d,t)\} = \{e \in X(dd',t)\}$ *for all $d, d' \in \mathbb{N}, t \in \mathcal{R}$.*

THEOREM 2.2. *Let $\{X(d,t)\}_{t\in\mathcal{R}, d\in\mathbb{N}}$ be a recombinant family of subsets of $\mathcal{R}$. Let $A : \mathbb{N} \times \mathcal{R} \to \mathcal{C}$ and $B : \mathbb{N} \times \mathcal{R} \to \mathcal{C}$ be functions, where $\mathcal{C}$ is a commutative ring with identity. Then*

$$A(n,t) = \sum_{d|n} \sum_{e\in X(d,t)} B\left(\frac{n}{d}, e\right)$$

for all $n \in \mathbb{N}$ and $t \in \mathcal{R}$ if and only if

$$B(n,t) = \sum_{d|n} \mu(d) \sum_{e\in X(d,t)} A\left(\frac{n}{d}, e\right)$$

for all $n \in \mathbb{N}$ and $t \in \mathcal{R}$.

Proof. Consider the sum, call it S, on the right-hand side of the first equation

$$\begin{aligned} S &= \sum_{d|n} \sum_{e\in X(d,t)} B\left(\frac{n}{d}, e\right) \\ &= \sum_{d|n} \sum_{e\in X(d,t)} \sum_{d'|(n/d)} \sum_{e'\in X(d',e)} \mu(d') A\left(\frac{n}{dd'}, e'\right) \\ &= \sum_{d|n} \sum_{dd'|n} \mu(d') \sum_{e\in X(d,t)} \sum_{e'\in X(d',e)} A\left(\frac{n}{dd'}, e'\right). \end{aligned}$$

Now substitute $f = dd'$ and use recombination to get

$$\begin{aligned}
S &= \sum_{d|n} \sum_{f|n} [\![f = dd']\!] \mu\left(\frac{f}{d}\right) \sum_{e \in X(d,t)} \sum_{e' \in X(d',e)} A\left(\frac{n}{f}, e'\right) \\
&= \sum_{f|n} \sum_{d|f} \mu\left(\frac{f}{d}\right) \sum_{e \in X(f,t)} A\left(\frac{n}{f}, e\right) \\
&= \sum_{f|n} \sum_{e \in X(f,t)} A\left(\frac{n}{f}, e\right) \sum_{d|f} \mu\left(\frac{f}{d}\right) \\
&= \sum_{f|n} \sum_{e \in X(f,t)} A\left(\frac{n}{f}, e\right) [\![f = 1]\!] \\
&= A(n,t).
\end{aligned}$$

Verification in the other direction is similar and is omitted. □

LEMMA 2.3. *Let $d \in \mathbb{N}$ and e, t be members of an additive monoid $\mathcal{R}$. The sets $\{e : de = t\}$ form a recombinant family.*

Proof. Here de means $e + e + \cdots + e$ (d terms). Suppose that $de = t$ and $d'e' = e$. Clearly, $dd'e' = t$. Conversely, if $dd'e' = t$, then $d'e'$ is equal to some element of $\mathcal{R}$, call it e. Then $d'e' = e$ and $de = t$. □

COROLLARY 2.4. *For a fixed prime power q, the sets $X_q(d,t) = \{e \in GF(q) : de = t\}$ form a recombinant family of subsets of $GF(q)$.*

COROLLARY 2.5. *For a fixed integer q, the sets $X_q(d,t) = \{e \in \mathbb{Z}_q : de \equiv t(q)\}$ form a recombinant family of subsets of $\mathbb{Z}_q$, where $\mathbb{Z}_q$ are the integers mod q.*

3. Irreducible polynomials with given trace. In this section, the irreducible polynomials with a given trace are counted. We begin by introducing some notation that will be used in the remainder of the paper. We use Jungnickel [3] as a reference for terminology and basic results from finite field theory.

The trace of an element $\beta \in GF(q^n)$ over $GF(q)$ is denoted $Tr(\beta)$ and is given by

$$Tr(\beta) = \beta + \beta^q + \beta^{q^2} + \cdots + \beta^{q^{n-1}}.$$

If $\beta \in GF(q^n)$ and d is the smallest positive integer for which $\beta^{q^d} = 1$, then $f(x)$ is the minimal polynomial of β, denoted $\mathrm{Min}(\beta)$, where

$$f(x) = (x - \beta)(x - \beta^q) \cdots (x - \beta^{q^{d-1}}).$$

The value of d must be a divisor of n.

Let $\mathbf{Irr}_q(n,t)$ denote the set of all monic irreducible polynomials over $GF(q)$ of degree n and trace t. By $a \cdot \mathbf{Irr}_q(n,t)$ we denote the multiset consisting of a copies of $\mathbf{Irr}_q(n,t)$. Classic results of finite field theory imply the following equality of multisets:

$$\bigcup_{\beta \in \mathrm{GF}(q^n)} \{\mathrm{Min}(\beta)\} = \bigcup_{d|n} d \cdot \mathbf{Irr}_q(d) = \bigcup_{d|n} \frac{n}{d} \cdot \mathbf{Irr}_q\left(\frac{n}{d}\right), \tag{3.1}$$

where $\mathbf{Irr}_q(d)$ is the set of monic irreducible polynomials of degree d over $GF(q)$. From (3.1) it is easy to derive (1.1) via a standard application of Möbius inversion.

Now we restrict the equality (3.1) to trace t field elements to obtain

$$\bigcup_{\substack{\beta\in \mathrm{GF}(q^n)\\ Tr(\beta)=t}} \{\mathrm{Min}(\beta)\} = \bigcup_{d|n} \frac{n}{d}\cdot\left\{f\in \mathbf{Irr}_q\left(\frac{n}{d}\right) \ : \ Tr(f^d)=t\right\} \tag{3.2}$$

$$= \bigcup_{d|n} \frac{n}{d}\cdot\left\{f\in \mathbf{Irr}_q\left(\frac{n}{d}\right) \ : \ d\cdot Tr(f)=t\right\} \tag{3.3}$$

$$= \bigcup_{d|n}\bigcup_{de=t} \frac{n}{d}\cdot\left\{f\in \mathbf{Irr}_q\left(\frac{n}{d}\right) \ : \ Tr(f)=e\right\} \tag{3.4}$$

$$= \bigcup_{d|n}\bigcup_{de=t} \frac{n}{d}\cdot\left\{f\in \mathbf{Irr}_q\left(\frac{n}{d},e\right)\right\}. \tag{3.5}$$

Note that the equation $de = t$ is asking whether the d-fold sum of $e \in GF(q)$ is equal to $t \in GF(q)$. We use the notation $GF(q^n, t)$ for the set of elements in $GF(q^n)$ with trace t, for $t = 0, 1, \ldots, q-1$, where $q = p^m$ and p is prime. Consider the map ρ that sends α to $\alpha + \gamma$, where $\gamma \in GF(q^n)$ has trace 1. We claim that $\rho(GF(q^n, t)) = GF(q^n, t+1)$, and so the number of elements is the same for each trace value. Thus

$$|GF(q^n, t)| = q^{n-1}.$$

Taking cardinalities in (3.5) gives

$$q^{n-1} = \sum_{d|n}\sum_{de=t} \frac{n}{d} I_q\left(\frac{n}{d}, e\right).$$

From Theorem 2.2 and Corollary 2.4, we obtain

$$I_q(n,t) = \frac{1}{qn}\sum_{d|n}\sum_{de=t} \mu(d) q^{n/d}.$$

The equation $de = t$ where d is an integer and $e, t \in GF(q)$ has a unique solution e if $t \neq 0$ and $p \nmid d$. If $t = 0$, then there is one solution $e = 0$ if $p \nmid d$ and there are q solutions if $p \mid d$. Thus, if $t \neq 0$, then

$$I_q(n,t) = \frac{1}{qn}\sum_{\substack{d|n\\ p\nmid d}} \mu(d) q^{n/d},$$

thereby proving Theorem 1.1. Otherwise, if $t = 0$, then

$$I_q(n,0) = I_q(n,1) + \frac{1}{n}\sum_{\substack{d|n\\ p\mid d}} \mu(d) q^{n/d}.$$

4. Lyndon words with given trace. If $\mathbf{a} = a_1 a_2 \cdots a_n$ is a word, then we define its trace mod q, $Tr_q(\mathbf{a})$, to be $\sum a_i \bmod q$. Let $L_q(n,t)$ denote the number of q-ary Lyndon words of length n and trace t mod q. Note that any q-ary string of length n can be expressed as the concatenation of d copies of the rotation of some Lyndon word of length n/d for some $d \mid n$. Note further that there are precisely q^{n-1}

words of length n with trace t because any word of length $n-1$ can have a final nth character appended in only one way to have trace t. It therefore follows that

$$q^{n-1} = \sum_{d|n} \sum_{de \equiv t(q)} \frac{n}{d} L_q\left(\frac{n}{d}, e\right). \tag{4.1}$$

This can be solved using Theorem 2.2 and Corollary 2.5 to yield

$$nL_q(n,t) = \sum_{d|n} \mu(d) \sum_{de \equiv t(q)} q^{n/d-1}.$$

Hence

$$L_q(n,t) = \frac{1}{qn} \sum_{\substack{d|n \\ \gcd(q,d)|t}} \gcd(q,d)\mu(d) q^{n/d}. \tag{4.2}$$

Equation (4.2) is true because $de \equiv t(q)$ has a solution if and only if $\gcd(d,q) \mid t$. If a solution exists, then it has precisely $\gcd(d,q)$ solutions (e.g., [2, Corollary 33.22, p. 821]). This proves Theorem 1.2.

We could also consider the more general question of computing $L_{q,r}(n,t)$, the number of q-ary Lyndon words with trace mod r, and derive similar but more complicated formulae. If $M_q(n,t)$ is the number of q-ary length n strings whose characters sum to t, then clearly $M_q(1,t) = [\![0 \le t < q]\!]$ and for $n > 1$

$$M_q(n,t) = \sum_{i=0}^{q} M_q(n-1, t-i).$$

If $T_{q,r}(n,t)$ denotes the number of q-ary length n strings with trace mod r equal to t, then

$$T_{q,r}(n,t) = \sum_{s \equiv t(r)} M_q(n,s).$$

Using the same approach as before

$$L_{q,r}(n,t) = \frac{1}{n} \sum_{d|n} \mu(d) \sum_{de \equiv t(r)} T_{q,r}\left(\frac{n}{d}, e\right).$$

The equation for $L_{q,r}(n,t)$ seems to produce no particularly nice formulae, except in the case seen previously where $q = r$ or if $q = 2$. When $q = 2$, $M_2(n,t) = \binom{n}{t}$ and

$$T_{2,r}(n,t) = \sum_{s \equiv t(r)} \binom{n}{s}.$$

However, in this case there is already a well-known formula for the number of Lyndon words with k 1's, namely,

$$P_2(n,k) = \frac{1}{n} \sum_{d|\gcd(n,k)} \mu(d) \binom{n/d}{k/d},$$

from which we obtain $L_{2,r}(n,t) = \sum_{s \equiv t(2)} P_2(n,s)$.

5. Final remarks. Our generalized Möbius inversion theorem can be extended to a Möbius inversion theorem on posets. Background material on Möbius inversion on posets may be found in Stanley [5]. We state here the modified definition of recombinant and the inversion theorem but omit the proof.

DEFINITION 5.1. *Let $\mathcal{P}$ be a poset, let $\mathcal{R}$ be a set, and let $\{X(y,x,t)\}_{x,y\in\mathcal{P},y\preceq x,t\in\mathcal{R}}$ be a family of subsets of $\mathcal{R}$. The family $\{X(y,x,t)\}_{x,y\in\mathcal{P},y\preceq x,t\in\mathcal{R}}$ is* recombinant *if*

(i) *$X(x,x,t)=\{t\}$ for all $t\in\mathcal{R}$ and*

(ii) *$\{e'\in X(z,y,e) : e\in X(y,x,t)\}=\{e\in X(z,x,t)\}$ for all $z\preceq y\preceq x\in\mathcal{P}, t\in\mathcal{R}$.*

We note that if $\mathcal{P}$ is the divisor lattice and $\mathcal{R}$ is an additive monoid, then the collection $\{X(x,y,t)\}_{x,y\in\mathcal{P},x\leq y,t\in\mathcal{R}}$ where $X(x,y,t)=\{e\in\mathcal{R} : (y/x)e=t\}$ is recombinant, as per Lemma 2.3.

THEOREM 5.2. *Let $\mathcal{P}$ be a poset, let $\mathcal{R}$ be a set, and let $\{X(y,x,t)\}_{x,y\in\mathcal{P},y\preceq x,t\in\mathcal{R}}$ be a recombinant family. Let $A:\mathcal{P}\times\mathcal{R}\to\mathcal{C}$, and $B:\mathcal{P}\times\mathcal{R}\to\mathcal{C}$, be functions where $\mathcal{C}$ is a commutative ring with identity. Then*

$$A(x,t)=\sum_{y\preceq x}\sum_{e\in X(y,x,t)}B(y,e)$$

for all $x\in\mathcal{P}$ and $t\in\mathcal{R}$ if and only if

$$B(x,t)=\sum_{y\preceq x}\mu(y,x)\sum_{e\in X(y,x,t)}A(y,e)$$

for all $x\in\mathcal{P}$ and $t\in\mathcal{R}$. (Here $\mu(y,x)$ is the Möbius function of the poset $\mathcal{P}$.)

Tables of the numbers $I_q(n,t)$ and $L_q(n,t)$ for small values of q and n may be found on Frank Ruskey's combinatorial object server (COS) at www.theory.csc.uvic.ca/~cos/inf/{lyndon.html,irreducible.html}. They also appear in Neil Sloane's on-line encyclopedia of integer sequences (at http://www.research.att.com/~njas/sequences/) as $I_2(n,0)=L_2(n,0)=$ A051841, $I_2(n,1)=L_2(n,1)=$ A000048, $I_3(n,0)=L_3(n,0)=$ A046209, $I_3(n,1)=L_3(n,1)=$ A046211, $L_4(n,0)=$ A054664, $I_4(n,1)=L_4(n,1)=$ A054660, $L_5(n,0)=$ A054661, $I_5(n,1)=L_5(n,1)=$ A054662, $L_6(n,0)=$ A054665, $L_6(n,1)=$ A054666, $L_6(n,2)=$ A054667, $L_6(n,3)=$ A054700.

Acknowledgment. The authors wish to thank Aaron Gulliver for helpful discussions regarding this paper.

REFERENCES

[1] L. CARLITZ, *A theorem of Dickson on irreducible polynomials*, Proc. Amer. Math. Soc., 3 (1952), pp. 693-700.

[2] T.H. CORMEN, C.E. LEISERSON, AND R.L. RIVEST, *Introduction to Algorithms*, McGraw-Hill, New York, 1990.

[3] D. JUNGNICKEL, *Finite Fields: Structure and arithmetics*, B.I. Wissenschaftsverlag, Mannheim, Germany, 1993.

[4] D.E. KNUTH, R.L. GRAHAM, AND O. PATASHNIK, *Concrete Mathematics*, Addison-Wesley, Reading, MA, 1989.

[5] R.P. STANLEY, *Enumerative Combinatorics, Vol.* I, Cambridge University Press, Cambridge, UK, 1997.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 246–255

A 0.5-APPROXIMATION ALGORITHM FOR MAX DICUT WITH GIVEN SIZES OF PARTS*

ALEXANDER AGEEV†, REFAEL HASSIN‡, AND MAXIM SVIRIDENKO§

Abstract. Given a directed graph G and an arc weight function $w : E(G) \to \mathbb{R}_+$, the maximum directed cut problem (MAX DICUT) is that of finding a directed cut $\delta(X)$ with maximum total weight. In this paper we consider a version of MAX DICUT—MAX DICUT with given sizes of parts or MAX DICUT WITH GSP—whose instance is that of MAX DICUT plus a positive integer p, and it is required to find a directed cut $\delta(X)$ having maximum weight over all cuts $\delta(X)$ with $|X| = p$. Our main result is a 0.5-approximation algorithm for solving the problem. The algorithm is based on a tricky application of the pipage rounding technique developed in some earlier papers by two of the authors and a remarkable structural property of basic solutions to a linear relaxation. The property is that each component of any basic solution is an element of a set $\{0, \delta, 1/2, 1-\delta, 1\}$, where δ is a constant that satisfies $0 < \delta < 1/2$ and is the same for all components.

Key words. approximation algorithm, directed cut, linear relaxation, basic solution

AMS subject classifications. 68W25, 05C85, 90C27, 90C35

PII. S089548010036813X

1. Introduction.

Let G be a directed graph. A directed cut in G is defined to be the set of arcs leaving some vertex subset X (we denote it by $\delta(X)$). Given a directed graph G and an arc weight function $w : E(G) \to \mathbb{R}_+$, the maximum directed cut problem (MAX DICUT) is that of finding a directed cut $\delta(X)$ with maximum total weight. In this paper we consider a version of MAX DICUT (MAX DICUT with given sizes of parts or MAX DICUT WITH GSP) whose instance is that of MAX DICUT plus a positive integer p, and it is required to find a directed cut $\delta(X)$ having maximum weight over all cuts $\delta(X)$ with $|X| = p$. MAX DICUT is well known to be NP-hard and so is MAX DICUT WITH GSP as the former evidently reduces to the latter.

The NP-hardness of MAX DICUT follows from the observation that the well-known undirected version of MAX DICUT—the maximum cut problem (MAX CUT), which is on the original Karp's list of NP-complete problems [Ka72]—reduces to MAX DICUT by substituting each edge for two oppositely oriented arcs. This means that for both problems there is no choice but to develop approximation algorithms. Nevertheless, this task turned out to be highly nontrivial, as for a long time it was an open problem whether it is possible to design approximations with factors better than trivial 1/2 for MAX CUT and 1/4 for MAX DICUT. Only quite recently, using a novel technique of rounding semidefinite relaxations, Goemans and Williamson [GW95] worked out algorithms solving MAX CUT and MAX DICUT approximately within factors of 0.878 and 0.796, respectively. A bit later Feige and Goemans [FG95] developed an algorithm for MAX DICUT with a better approximation ratio of 0.859. Recently, using a new

*Received by the editors February 18, 2000; accepted for publication (in revised form) January 10, 2001; published electronically April 3, 2001.
http://www.siam.org/journals/sidma/14-2/36813.html

†Sobolev Institute of Mathematics, pr. Koptyuga 4, Novosibirsk 630090, Russia (ageev@math.nsc.ru). The research of this author was partially supported by the Russian Foundation for Basic Research, grant 99-01-00601.

‡Department of Statistics and Operations Research, School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel (hassin@math.tau.ac.il).

§IBM T. J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (sviri@us.ibm.com). The research of this author was partially supported by the Russian Foundation for Basic Research, grant 99-01-00510.

method of rounding linear relaxations—the pipage rounding—Ageev and Sviridenko [AS99] developed a 0.5-approximation algorithm for the version of MAX CUT in which the parts of a vertex set bipartition are constrained to have given sizes (MAX CUT with given sizes of parts or MAX CUT WITH GSP). Later Hassin and Rubinstein [HR00] presented a different 0.5-approximation with a better running time. The paper [AS00] presents an extension of the algorithm in [AS99] to a hypergraph generalization of MAX CUT WITH GSP. Feige and Langberg [FL99] combined the method in [AS99] with the semidefinite programming approach to design a $0.5 + \varepsilon$-approximation for MAX CUT WITH GSP, where ε is some unspecified small positive number.

It is easy to see that MAX CUT WITH GSP reduces to MAX DICUT WITH GSP in the same way as MAX DICUT reduces to MAX CUT. However, unlike MAX CUT WITH GSP, MAX DICUT WITH GSP provides no possibilities for a straightforward application of the pipage rounding since the F/L lower bound condition in the description of the method (see section 2) does not, in general, hold.

Fortunately, the other main condition, ε-convexity, always holds. In the final section of this paper, we show that the F/L lower bound condition is still satisfied with $C = 0.5$ in the case when the arc weights form a circulation in the given graph as well as when the parts of a cut are restricted to have the same size (the DIGRAPH BISECTION problem). Thus, these cases can be approximated within a factor of 0.5 by the direct application of the pipage rounding method.

The main result of this paper is an algorithm that finds a feasible dicut of weight within a factor of 0.5 in the case of arbitrary weights. It turns out that to construct such an algorithm one needs to carry out a more profound study of the problem structure. A heaven-sent opportunity is provided by a remarkable structural property of basic solutions to a linear relaxation (Theorem 4.1). At this point we should notice the papers of Jain [Ja98] and Melkonian and Tardos [MT99], where exploiting structural properties of basic solutions was also crucial in designing better approximations for some network design problems.

The resulting algorithm (DIRCUT) is of rounding type and as such consists of two phases: the first phase is to find an optimal (fractional) solution to a linear relaxation; the second (rounding) phase is to transform this solution to a feasible (integral) solution. A special feature of the rounding phase is that it uses two different rounding algorithms (ROUND1 and ROUND2) based on the pipage rounding method and takes the best solution for the output. The worst-case analysis of the algorithm relies heavily on Theorem 4.1.

2. Pipage rounding: A general scheme. In this section, to make the paper self-contained, we give a general description of the pipage rounding method as it was presented in [AS99].

Assume that a problem P can be formulated as the following nonlinear binary program:

$$\max \quad F(x) \tag{2.1}$$

$$\text{subject to (s.t.)} \quad \sum_{i=1}^{n} x_i = p, \tag{2.2}$$

$$0 \le x_i \le 1, \quad i = 1, \dots, n, \tag{2.3}$$

$$x_i \in \{0, 1\}, \quad i = 1, \dots, n, \tag{2.4}$$

where p is a positive integer and $F(x)$ is a function defined on the rational points $x = (x_i)$ of the n-dimensional cube $[0, 1]^n$ and computable in polynomial time. Further

assume that one can associate with $F(x)$ another function, $L(x)$, which is defined on the same set, coincides with $F(x)$ on binary x satisfying (2.2), and the program

$$\max L(x) \tag{2.5}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_i = p, \tag{2.6}$$

$$0 \le x_i \le 1, \quad i = 1, \dots, n, \tag{2.7}$$

which we call *a nice relaxation,* is polynomially solvable. Next assume that the following two main conditions hold:

***F/L* lower bound condition:** there exists a constant C such that $0 < C \le 1$ and $F(x) \ge CL(x)$ for each rational point x in $[0,1]^n$;

ε-convexity condition: the function

$$\varphi(\varepsilon, x, i, j) = F(x_1, \dots, x_i + \varepsilon, \dots, x_j - \varepsilon, \dots, x_n) \tag{2.8}$$

is convex with respect to $\varepsilon \in [-\min\{x_i, 1-x_j\}, \min\{1-x_i, x_j\}]$ for each pair of indices i and j and each $x \in [0,1]^n$.

We now describe the pipage rounding procedure. Its input is a fractional solution x satisfying (2.2)–(2.3) and its output is an integral solution x satisfying (2.2)–(2.4) and having the property that $F(x) \ge F(\tilde{x})$. The pipage rounding consists of uniform "pipage steps." We describe the first step. If the solution x is not binary, then due to (2.2) it has at least two different components x_i and x_j with values lying strictly between 0 and 1. By ε-convexity condition, $\varphi(\varepsilon, x, i, j) \ge F(x)$ either for $\varepsilon = \min\{1 - x_i, x_j\}$ or for $\varepsilon = -\min\{x_i, 1 - x_j\}$. Thus we obtain a new feasible solution $x' = (x_1, \dots, x_i + \varepsilon, \dots, x_j - \varepsilon, \dots, x_n)$ with a smaller number of noninteger components and such that $F(x') \ge F(x)$. After repeating the "pipage" step at most $n - 1$ times we arrive at a binary feasible solution $\tilde{x}$ with $F(\tilde{x}) \ge F(x)$. Since each step can be performed in polynomial time, the overall running time of the described procedure is polynomially bounded.

Now suppose that x is an optimal solution to (2.5)–(2.7), satisfying both the ε-convexity and the F/L lower bound condition. Then $F(\tilde{x}) \ge F(x) \ge CL(x) \ge CF^*$, where F^* is the optimal value of (2.1)–(2.4). Thus the algorithm consisting of a polynomial-time procedure to solve the nice relaxation (2.5)–(2.7) and the pipage rounding finds a feasible solution to (2.1)–(2.4) of weight within a factor of C of the optimum. Note that if instead of a procedure for solving (2.5)–(2.7) we use *any* polynomial-time procedure to find a solution x satisfying (2.2)–(2.3) and $F(x) \ge CF^*$, then we also obtain a C-approximation algorithm.

3. Application: MAX DICUT WITH GSP. In this section, we show an implementation of the above scheme in the case of MAX DICUT WITH GSP and, on the side, specify the character of obstacles to the direct application of the pipage rounding method.

In what follows, $G = (V, A)$ stands for the graph in the input of MAX DICUT WITH GSP. Since assigning zero weights to missing arcs yields an equivalent problem, we may assume that A contains all possible arcs. Let $|V| = n$.

First, note that MAX DICUT WITH GSP can be formulated as the following non-

linear binary program:

$$\begin{aligned} \max \quad & F(x) = \sum_{ij \in A} w_{ij} x_i (1 - x_j) \\ \text{s.t.} \quad & \sum_{i \in V} x_i = p, \\ & x_i \in \{0, 1\} \quad \text{for all } i \in V. \end{aligned}$$

Second, just like MAX CUT WITH GSP in [AS99], MAX DICUT WITH GSP can be formulated as the following integer program:

$$\max \quad \sum_{ij \in A} w_{ij} z_{ij} \tag{3.1}$$

$$\text{s.t.} \quad z_{ij} \le x_i \text{ for all } ij \in A, \tag{3.2}$$

$$z_{ij} \le 1 - x_j \text{ for all } ij \in A, \tag{3.3}$$

$$\sum_i x_i = p, \tag{3.4}$$

$$0 \le x_i \le 1 \text{ for all } i \in V, \tag{3.5}$$

$$x_i, z_{kj} \in \{0, 1\} \text{ for all } i \in V, kj \in A. \tag{3.6}$$

Now observe that the variables z_{ij} can be excluded from (3.1)–(3.5) by setting

$$z_{ij} = \min\{x_i, (1 - x_j)\} \text{ for all } ij \in A.$$

Hence (3.1)–(3.5) is equivalent to maximizing

$$L(x) = \sum_{ij \in A} w_{ij} \min\{x_i, (1 - x_j)\} \tag{3.7}$$

subject to (3.4), (3.5).

Thus we have functions F and L that can be considered as those involved in the description of the pipage rounding (see section 2). Notice now that for each pair of indices i and j the function $\varphi(\varepsilon, x, i, j)$ defined by (2.8) is the sum of $w_{ij}(x_i + \varepsilon)[1 - (x_j - \varepsilon)] + w_{ji}(x_j - \varepsilon)[1 - (x_i + \varepsilon)]$ and a term linear in ε. It follows that $\varphi(\varepsilon, x, i, j)$ is a quadratic polynomial in ε having a nonnegative leading coefficient for each pair of indices i and j and each $x \in [0, 1]^n$. Thus, the function F obeys the ε-convexity condition. Unfortunately, the other, F/L lower bound condition, does not hold for any $C > 0$. We present below an example showing that the ratio $F(x)/L(x)$ may be arbitrarily close to 0 even when the underlying graph is bipartite.

Example 1. Consider the following instance of MAX DICUT WITH GSP. Let $V = V_1 \cup V_2 \cup V_3$, where $|V_1| = k$, $|V_2| = |V_3| = 2$. Let $A = A_1 \cup A_2$, where A_1 is the set of $2k$ arcs from V_1 to V_2 inducing a complete bipartite graph on (V_1, V_2) and A_2 is the set of 4 arcs from V_2 to V_3 inducing a complete bipartite graph on (V_2, V_3) (see Figure 3.1). Assume the arc weights to be units and $p \le k$.

Then for any feasible solution x,

$$\begin{aligned} L(x) &= \sum_{ij \in A} \min\{x_i, 1 - x_j\} \le \sum_{ij \in A} x_i \\ &= 2 \sum_{i \in V_1} x_i + 2 \sum_{i \in V_2} x_i \le 2p. \end{aligned}$$

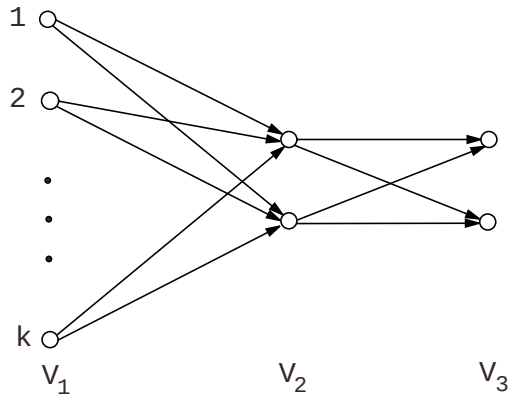


FIG. 3.1. *An example demonstrating that the ratio $F(x)/L(x)$ may be arbitrarily close to* 0.

In fact, $2p$ is the optimal value of the nice relaxation and it can be obtained in more than one way. One way is to let $x_i = r = \frac{p-2}{k-2}$ for $i \in V_1$, $x_i = 1 - r$ for $i \in V_2$, and $x_i = 0$ for $i \in V_3$. Then $F(x) = 2kr^2 + 4(1 - r)$. Now if p/k tends to 0 as p tends to infinity (for example, set $k = p^2$), $F(x)/L(x)$ tends to 0.

Note that the same can be done with $|V_3| > 2$ and then the above x will be the unique optimal solution to the nice relaxation.

Thus straightforward application of the pipage rounding method does not provide a constant-factor approximation.

Moreover, Example 1 shows that the greedy algorithm (at each step add a vertex which increases most or decreases least the weight of the cut) also does not yield any constant-factor approximation. For this instance the greedy algorithm may first choose the vertices of V_2 and then no more arcs can be added and a solution with only four arcs will be the outcome (while the optimal one is to choose p vertices from V_1, which gives a cut of size $2p$).

4. The structure of basic solutions. The following fact, which may be of interest beyond the purposes of this paper, is crucial in constructing a 0.5-approximation for MAX DICUT WITH GSP in the general case.

THEOREM 4.1. *Let (x, z) be a basic feasible solution to the linear relaxation* (3.1)–(3.5). *Then*

$$x_i \in \{0, \delta, 1/2, 1 - \delta, 1\} \text{ for each } i \tag{4.1}$$

for some $0 < \delta < 1/2$.

Proof. Let (x, z) be a basic feasible solution. By definition, (x, z) is the unique solution to the system of linear equations formed by those constraints in (3.2)–(3.5) which hold with equality. First, observe that for any $ij \in A$, either both $z_{ij} \le x_i$ and $z_{ij} \le 1 - x_j$ hold with equalities or exactly one holds with equality and the other with strict inequality. In the former case we exclude z_{ij} by replacing these equations with the single equation $x_i + x_j = 1$. In the latter case we delete that equality from the linear system. In either case the variable z_{ij} retires from the system while the number of equations reduces by one and, moreover, the value of z_{ij} can be uniquely

determined by the values of x_i and x_j. After performing this operation for each $ij \in A$, we arrive at a system that can be written in the following form:

$$y_i + y_j = 1 \text{ for } ij \in A' \subseteq A, \tag{4.2}$$

$$\sum_i y_i = p, \tag{4.3}$$

$$y_i = 0 \text{ for every } i \text{ such that } x_i = 0, \tag{4.4}$$

$$y_i = 1 \text{ for every } i \text{ such that } x_i = 1. \tag{4.5}$$

Since the removed z can be uniquely restored from the vector x, x must be the unique solution to this system. Remove all components of x equal to either 0 or 1 or 1/2 and denote the set of the remaining indices by I^*. Denote by x' the subvector of x consisting of the components with indices in I^*. Now consider the system that is obtained from (4.2)–(4.5) by substituting $y_i = x_i$ for each i such that $x_i \in \{0, 1/2, 1\}$. Since x is the unique solution to (4.2)–(4.5), x' is the unique solution to the resulting system, which can be written in the following form:

$$y'_i + y'_j = 1 \text{ for } ij \in A'' \subseteq A', \tag{4.6}$$

$$\sum_i y'_i = p', \tag{4.7}$$

where $p' \leq p$. It follows that one can choose $|I^*|$ independent equations from (4.6)–(4.7). We claim that any subsystem of this sort must contain (4.7). Assume to the contrary that $|I^*|$ equations from the set (4.6) form an independent system. Consider the (undirected) subgraph H of G (we ignore orientations) corresponding to these equations. Note that $|E(H)| = |I^*|$. Since $x'_i \neq 1/2$ for all $i \in I^*$, H does not contain odd cycles. Moreover, H cannot have even cycles as the subsystem corresponding to such a cycle is clearly dependent. Thus H is acyclic. But then $|E(H)| \leq |I^*| - 1$, a contradiction.

Now fix $|I^*|$ independent equations from (4.6)–(4.7). Then, by the above claim, we obtain the following system:

$$y'_i + y'_j = 1 \text{ for } ij \in A^*, \tag{4.8}$$

$$\sum_{i \in I^*} y'_i = p', \tag{4.9}$$

where $A^* \subseteq A'$. Since all the equations in (4.8)–(4.9) are independent, $|I^*| = |A^*| + 1$. We have proved above that the subgraph spanned by A^* is acyclic, which together with $|I^*| = |A^*| + 1$ implies that it is a tree. Recall also that x' is the unique solution to (4.8)–(4.9) and $x'_i = x_i \neq 0, 1, 1/2$ for all $i \in I^*$. All these facts together with the structure of equalities (4.8) imply that the components of x with indices in I^* split into two sets—those equal to some $0 < \delta < 1/2$ and those equal to $1 - \delta$. □

5. Algorithm DIRCUT. Section 3 demonstrates that MAX DICUT WITH GSP in the general setting does not admit a direct application of the pipage rounding method. In this section we show that by using Theorem 4.1 and some tricks one is able to design not only a constant factor but even a 0.5-approximation for solving MAX DICUT WITH GSP. Moreover, the performance bound of 0.5 cannot be improved using different methods of rounding as the integrality gap of (3.1)–(3.5) can be arbitrarily close to 1/2 (this can be shown exactly in the same way as it was done for MAX CUT WITH GSP in [AS99]).

For $ij \in A$, call the number $w_{ij} \min\{x_i, (1 - x_j)\}$ the *contributed weight* of the arc ij. Observe that for any $a, b \in [0, 1]$, $ab = \max\{a, b\} \min\{a, b\}$. It follows that

$$F(x) = \sum_{ij \in A} w_{ij} x_i (1 - x_j) = \sum_{ij \in A} w_{ij} \max\{x_i, 1 - x_j\} \min\{x_i, 1 - x_j\}$$

$$= \sum_{ij \in A} \max\{x_i, 1 - x_j\} (\text{“ the contributed weight of } ij \text{ ”}). \tag{5.1}$$

Algorithm DIRCUT consists of two phases: the first phase is to find an optimal (fractional) basic solution to the linear relaxation (3.1)–(3.5); the second (rounding) phase is to transform this solution to a feasible (integral) solution. The rounding phase runs two different rounding algorithms based on the pipage rounding method and takes the best solution for the output. Let (x, z) denote a basic optimal solution to (3.1)–(3.5) obtained at the first phase. Recall that, by Theorem 4.1, the vector x satisfies (4.1). Set $V_1 = \{i : x_i = \delta\}$, $V_2 = \{i : x_i = 1 - \delta\}$, $V_3 = \{i : x_i = 1/2\}$, $V_4 = \{i : x_i = 0 \text{ or } 1\}$. Denote by l_{kl} $(k, l = 1, 2, 3, 4)$ the sum of contributed weights over all arcs going from V_k to V_l. Set $l_0 = l_{11} + l_{21} + l_{22} + l_{23} + l_{31}$, $l_1 = l_{33} + l_{13} + l_{32}$, $l_2 = \sum_{i=1}^{4} (l_{i4} + l_{4i})$ (Figure 5.1 might be helpful to the reader).

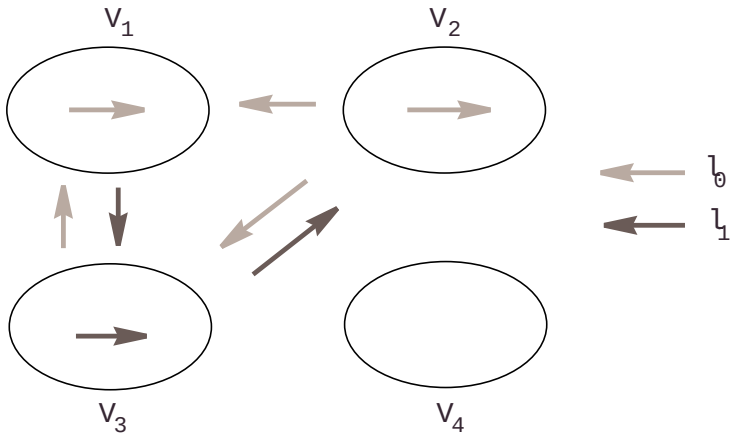


FIG. 5.1. *l_0 and l_1 are the sums of contributed weights over the displayed collections of arcs.*

The second phase of the algorithm successively calls two rounding algorithms—ROUND1 and ROUND2—and takes a solution with maximum weight for the output.

ROUND1 is the pipage rounding applied to the optimal basic solution x. Let $\tilde{x}$ denote the output of ROUND1. Then by the description of pipage rounding, $F(\tilde{x}) \geq F(x)$. Since the number $\max\{x_i, 1 - x_j\}$ remains unchanged for any $ij \in A$ such that $i \in V_k$ and $j \in V_l$, and can be easily computed, (5.1) implies that

$$F(\tilde{x}) \geq F(x) \geq \delta l_{12} + (1 - \delta) l_0 + l_1/2 + l_2. \tag{5.2}$$

Algorithm ROUND2 is the pipage rounding applied to a different fractional solution x' which is obtained by an alteration of x.

ALGORITHM ROUND2. Define a new vector x' by the following formulas:

$$x'_i = \begin{cases} \min\{1, \delta + (1 - \delta)|V_2|/|V_1|\} & \text{if } i \in V_1, \\ \max\{0, (1 - \delta) - (1 - \delta)|V_1|/|V_2|\} & \text{if } i \in V_2, \\ x_i & \text{if } i \in V \setminus (V_1 \cup V_2). \end{cases} \tag{5.3}$$

Apply the pipage rounding to x'.

Analysis. The vector x' is obtained from x by redistributing uniformly the values from the components in V_2 to those in V_1 and keeping the same the remaining ones. It follows from the description of ROUND2 that x' is feasible. Applying the pipage rounding to x' results in an integral feasible vector of weight at least $F(x')$. We claim that $F(x') \geq l_{12} + l_1/2$. Consider first the case when $|V_1| \geq |V_2|$. Then by (5.3), $x'_i = 0$ for all $i \in V_2$ and $x'_i \geq \delta$ for all $i \in V_1$. Therefore, by (5.1) and definitions of V_k it follows that $F(x') \geq l_{32} + l_{12} + 1/2(l_{13} + l_{33})$. Now assume that $|V_1| \leq |V_2|$. Then by (5.3), $x'_i = 1$ and $x'_i \leq 1 - \delta$ for all $i \in V_1$. Hence, again by (5.1), $F(x') \geq l_{13} + l_{12} + 1/2(l_{32} + l_{33})$. Therefore, in either case $F(x') \geq l_{12} + l_1/2$. Thus ROUND2 outputs a solution of weight at least $l_{12} + l_1/2$. Together with (5.2) this implies that the output of DIRCUT has weight at least

$$\max\{l_{12} + l_1/2, \delta l_{12} + (1 - \delta)l_0 + l_1/2 + l_2\},$$

which is bounded from below by

$$q = \max\{l_{12}, \delta l_{12} + (1 - \delta)l^*\} + l_1/2,$$

where $l^* = l_0 + l_2$. Now recall that $0 < \delta < 1/2$. Hence, if $l_{12} \geq l^*$, then $q = l_{12} + l_1/2 \geq (l_{12} + l^* + l_1)/2$ and if $l_{12} < l^*$, then $q = \delta l_{12} + (1 - \delta)l^* + l_1/2 > (l_{12} + l^*)/2 + l_1/2$. Thus, in either case algorithm DIRCUT outputs a solution of weight at least $(l_{12} + l_0 + l_1 + l_2)/2$, which is at least half of the optimum.

6. Directly tractable special cases. In the final section we consider two special cases of MAX DICUT WITH GSP which admit direct application of the pipage rounding method.

6.1. The circulation case. We first consider the case when the weight function w is a circulation in the given graph. This means that the function w obeys the condition

$$\sum_{j:ij\in A} w_{ij} = \sum_{k:ki\in A} w_{ki}$$

for each vertex $i \in V$. We will show that the circulation case of MAX DICUT WITH GSP admits a 0.5-approximation algorithm which is a straightforward implementation of the scheme described in section 2. In the next subsection we will show that it also finds a cut of weight within a factor of 0.5 of the optimum in the case when the cuts are constrained to have equal parts (the DIGRAPH BISECTION problem).

Note first that for any a and b between 0 and 1,

$$2a(1 - b) = a(1 - b) + b(1 - a) + a - b,$$
$$2\min\{a, 1 - b\} = \min\{a + b, 2 - a - b\} + a - b.$$

Using these identities we can rearrange the functions F and L in the following way:

$$F(x) = 1/2 \sum_{ij\in A} w_{ij}\big[x_i(1 - x_j) + x_j(1 - x_i) + x_i - x_j\big],$$

$$L(x) = 1/2 \sum_{ij\in A} w_{ij}\big[\min\{x_i + x_j, 2 - x_i - x_j\} + x_i - x_j\big].$$

Since

$$\sum_{ij\in A} w_{ij}(x_i - x_j) = \sum_{i\in V} q_i x_i,$$

where q_i is the difference between the sum of weights of arcs leaving the node i and the sum of weights of arcs entering the node i, both functions can be expressed as the sums of two summands

$$F(x) = 1/2 \sum_{ij\in A} w_{ij}\big[x_i(1-x_j) + x_j(1-x_i)\big] + 1/2 \sum_{i\in V} q_i x_i, \tag{6.1}$$

$$L(x) = 1/2 \sum_{ij\in A} w_{ij}\big[\min\{x_i + x_j, 2 - x_i - x_j\}\big] + 1/2 \sum_{i\in V} q_i x_i. \tag{6.2}$$

Ageev and Sviridenko [AS99] proved that for $0 \le x_i, x_j \le 1$,

$$\frac{x_i(1-x_j) + x_j(1-x_i)}{\min\{x_i + x_j, 2 - x_i - x_j\}} \ge 1/2. \tag{6.3}$$

It follows that $F(x)/L(x) \ge 1/2$ if $\sum_{i\in V} q_i x_i \ge 0$. In the case when the weights w form a circulation in G, each q_i is equal to zero. This proves the bound of $1/2$ for this case.

6.2. DIGRAPH BISECTION. The DIGRAPH BISECTION problem is the special case of MAX DICUT WITH GSP where $n = 2p$. Let (x, z) be an optimal solution to the linear relaxation (3.1)–(3.5). We claim that in this case $\sum_{i\in V} q_i x_i \ge 0$, which means by (6.1), (6.2), and (6.3) that the algorithm presented for the circulation case also has an approximation ratio of $1/2$ for DIGRAPH BISECTION. Indeed, assume to the contrary that $\sum_{i\in V} q_i x_i < 0$. Since $n = 2p$, the vector y defined by

$$y_i = 1 - x_i \quad \text{for all } i \in V$$

is also feasible for the nice relaxation (3.7), (3.4), (3.5). Moreover,

$$\sum_{i\in V} q_i x_i = -\sum_{i\in V} q_i y_i$$

and

$$\min\{x_i + x_j, 2 - x_i - x_j\} = \min\{y_i + y_j, 2 - y_i - y_j\}$$

for every $ij \in A$. This means that $L(y) > L(x)$, which contradicts the optimality of x.

REFERENCES

[AS99] A. A. AGEEV AND M.I. SVIRIDENKO, *Approximation algorithms for maximum coverage and max cut with given sizes of parts*, in Integer Programming and Combinatorial Optimization, G. Cornuéjols, R. E. Burcard, and G. J. Woeginger, eds., Lecture Notes in Comput. Sci. 1610, Springer-Verlag, New York, 1999, pp. 17–30.

[AS00] A. A. AGEEV AND M. I. SVIRIDENKO, *An approximation algorithm for hypergraph max k-cut with given sizes of parts*, in Proceedings of the 8th Annual European Symposium on Algorithms (ESA'00), M. Paterson, ed., Lecture Notes in Comput. Sci. 1879, Springer-Verlag, New York, 2000, pp. 32–41.

[FG95] U. Feige and M.X. Goemans, *Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT*, in Proceedings of the Third Israel Symposium on Theory of Computing and Systems, Tel Aviv, Israel, 1995, pp. 182–189.

[FL99] U. Feige and M. Langberg, *Approximation Algorithms for Maximization Problems Arising in Graph Partitioning*, manuscript, 1999.

[GW95] M. X. Goemans and D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. ACM, 42 (1995), pp. 1115–1145.

[HR00] R. Hassin and S. Rubinstein, *Approximation algorithms for maximum linear arrangement*, in Proceedings of the 7th Scandinavian Workshop on Algorithm Theory (SWAT'00), M. M. Halldórsson, ed., Lecture Notes in Comput. Sci. 1851, Springer-Verlag, New York, 2000, pp. 231–236.

[Ja98] K. Jain, *A factor* 2 *approximation algorithm for the generalized Steiner network problem*, in Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS'98), IEEE Computer Society, 1998, pp. 448–457.

[Ka72] R. M. Karp, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, R. E. Miller and J. W. Thatcher, eds., Plenum Press, New York, 1972, pp. 85–103.

[MT99] V. Melkonian and É. Tardos, *Approximation algorithms for a directed network design problem*, in Integer Programming and Combinatorial Optimization, G. Cornuéjols, R. E. Burcard, and G. J. Woeginger, eds., Lecture Notes in Comput. Sci. 1610, Springer-Verlag, New York, 1999, pp. 345–360.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 256–266

OPTIMAL $(9v, 4, 1)$ OPTICAL ORTHOGONAL CODES*

RYOH FUJI-HARA†, YING MIAO†, AND JIANXING YIN‡

Abstract. Optimal $(9p, 4, 1)$ optical orthogonal codes (OOCs) are constructed for all primes p congruent to 1 modulo 4. Direct constructions with explicit codewords are presented for the case $p \equiv 13 \bmod 24$, and Weil's theorem on character sums is used to settle the cases $p \equiv 1, 5, 17 \bmod 24$. By applying a known recursive construction, optimal $(9v, 4, 1)$-OOCs are obtained for all v, a product of primes congruent to 1 modulo 4.

Key words. optical orthogonal code, optimal, cyclic packing, g-regular, direct construction, Weil's theorem

AMS subject classification. 94B60

PII. S0895480100377234

1. Introduction and preliminaries.

The study of optical orthogonal codes (OOCs) was first motivated by an application in a fiber-optic code-division multiple access channel which requires binary sequences with good correlation properties. Recent work has also been done on using optical orthogonal codes for multimedia transmission in fiber-optic LANs and in multirate fiber-optic CDMA systems. For general background on OOCs, the reader may refer to [3, 5, 10, 11, 12, 13]. In this paper, we will not try to explore new applications of OOCs; instead, we will focus our attention on the constructions of (v, k, λ)-OOCs, which are defined below.

Let v, k, and λ be positive integers. A (v, k, λ) *optical orthogonal code*, or briefly (v, k, λ)-OOC, $\mathcal{C}$, is a family of $(0, 1)$-sequences (called *codewords*) of *length* v and *weight* k satisfying the following two properties:

(1) (*the auto-correlation property*)
$\sum_{0 \le t \le v-1} x_t x_{t+i} \le \lambda$ for any $\mathbf{x} = (x_0, x_1, \ldots, x_{v-1}) \in \mathcal{C}$ and any integer $i \not\equiv 0 \bmod v$;

(2) (*the cross-correlation property*)
$\sum_{0 \le t \le v-1} x_t y_{t+i} \le \lambda$ for any $\mathbf{x} = (x_0, x_1, \ldots, x_{v-1}) \in \mathcal{C}$, $\mathbf{y} = (y_0, y_1, \ldots, y_{v-1}) \in \mathcal{C}$ with $\mathbf{x} \ne \mathbf{y}$, and any integer i.

All subscripts here are reduced modulo v so that only periodic correlations are considered.

EXAMPLE 1.1. *The following three codewords form a* $(45, 4, 1)$*-OOC:*

$$1001000000000000000000000000000000000100010000,$$
$$0011000000000000001000000000000000000010000000,$$
$$0000000001000000000000000001000000100000010000.$$

*Received by the editors August 29, 2000; accepted for publication (in revised form) January 12, 2001; published electronically April 3, 2001.

http://www.siam.org/journals/sidma/14-2/37723.html

†Institute of Policy and Planning Sciences, University of Tsukuba, Tsukuba 305-8573, Japan (fujihara@sk.tsukuba.ac.jp, miao@sk.tsukuba.ac.jp). The first and second authors were supported by JSPS by Grant-in-Aid for Scientific Research (C) and Grant-in-Aid for Encouragement of Young Scientists under contracts 11640099 and 12740054, respectively.

‡Department of Mathematics, Suzhou University, Suzhou 215006, People's Republic of China (jxyin@suda.edu.cn). The third author was supported in part by NSFC grant 10071056.

Analogous to the Johnson bound [8] for constant-weight binary error-correcting codes, the size $|\mathcal{C}|$ of a (v, k, λ)-OOC $\mathcal{C}$ is upper bounded (see [3]) by

$$\frac{(v-1)(v-2)\cdots(v-\lambda)}{k(k-1)\cdots(k-\lambda)}.$$

When $\lambda = 1$, this reduces to $\frac{(v-1)}{k(k-1)}$. A $(v, k, 1)$-OOC with $\lfloor \frac{v-1}{k(k-1)} \rfloor$ codewords is said to be *optimal*. The $(45, 4, 1)$-OOC in Example 1.1 is in fact optimal.

Optimal OOCs are closely related to combinatorial configurations (see, for example, [3, 5, 6, 14]). For simplicity, we do not develop a general relationship but only the one between optimal $(v, k, 1)$-OOCs and optimal $(v, k, 1)$ cyclic packings to meet the requirement of this paper.

A $(v, k, 1)$ *packing*, denoted by $\mathrm{P}(k, 1; v)$, is a pair $(X, \mathcal{B})$ where X is a v-set (of *points*) and $\mathcal{B}$ is a collection of k-subsets (called *blocks*) of X such that every pair of distinct points from X occurs in at most one block of $\mathcal{B}$.

Consider a packing $\mathrm{P}(k, 1; v)$ $(X, \mathcal{B})$. Let σ be a permutation on X. For any block $B = \{b_1, \ldots, b_k\}$, define $B^\sigma = \{b_1^\sigma, \ldots, b_k^\sigma\}$. If $\mathcal{B}^\sigma = \{B^\sigma : B \in \mathcal{B}\} = \mathcal{B}$, then σ is called an *automorphism* of the packing $\mathrm{P}(k, 1; v)$. Any automorphism σ partitions $\mathcal{B}$ into equivalence classes called the *orbits* of $\mathcal{B}$ under σ. A collection of *base blocks* is a collection of representatives for these orbits of $\mathcal{B}$. A $\mathrm{P}(k, 1; v)$ is said to be *cyclic* if it admits an automorphism consisting of a single cycle of length v. In this case, the point set X can be identified with Z_v, the residue ring of integers modulo v. The cyclic automorphism is then just the bijection $\sigma : i \longrightarrow i + 1 \bmod v$. Following [14], we use notation $\mathrm{CP}(k, 1; v)$ to denote the collection of base blocks of a cyclic $\mathrm{P}(k, 1; v)$ in which each of its block orbits under the cyclic automorphism contains exactly v distinct blocks.

A convenient way of viewing a $\mathrm{CP}(k, 1; v)$ is from the difference family perspective. A $\mathrm{CP}(k, 1; v)$ can be defined equivalently as a family $\mathcal{B} = \{B_1, B_2, \ldots, B_t\}$ of t k-subsets (called *base blocks*) of Z_v such that the setwise stabilizer G_{B_i} of each B_i, $1 \le i \le t$, is the identity $\{0\}$, and the differences $\Delta\mathcal{B}$ from $\mathcal{B}$,

$$\Delta\mathcal{B} = \{a - b : a, b \in B, a \ne b, B \in \mathcal{B}\},$$

cover each nonzero residue of integers modulo v at most once. Obviously, we have $t \le \lfloor \frac{v-1}{k(k-1)} \rfloor$. When $t = \lfloor \frac{v-1}{k(k-1)} \rfloor$, the $\mathrm{CP}(k, 1; v)$ is called *optimal*. A $\mathrm{CP}(k, 1; v)$ is termed *g-regular* if the subset $Z_v - \Delta\mathcal{B}$ forms an additive subgroup of Z_v with order g.

From each codeword of a $(v, k, 1)$-OOC, we can construct a k-set of integers modulo v where the numbers in each k-set specify the nonzero bits of the codeword. The corresponding 4-sets of integers modulo 45 of the codewords in the optimal $(45, 4, 1)$-OOC in Example 1.1 are $\{0, 3, 36, 40\}, \{2, 3, 18, 37\}, \{9, 27, 34, 40\}$, which form a $\mathrm{CP}(4, 1; 45)$. The setwise stabilizer of each base block is the identity $\{0\}$, and the differences from these base blocks are $Z_{45} - \{0, 2, 17, 21, 22, 23, 24, 28, 43\}$. This $\mathrm{CP}(4, 1; 45)$ is optimal but not 9-regular. In fact, by a computer search, we know that no 9-regular $\mathrm{CP}(4, 1; 45)$ can exist.

The following results can be found in [14].

LEMMA 1.2. *An optimal $(v, k, 1)$-OOC is equivalent to an optimal $CP(k, 1; v)$.*

LEMMA 1.3. *If $1 \le g \le k(k-1)$, then a g-regular $CP(k, 1; v)$ is optimal.*

LEMMA 1.4. *Suppose that p_1 and $p_2 \ge k$ are two odd primes. If both a g-regular $CP(k, 1; gp_1)$ and an optimal $CP(k, 1; gp_2)$ exist, then so does an optimal*

CP$(k, 1; gp_1p_2)$. *Moreover, if the given CP*$(k, 1; gp_1p_2)$ *is g-regular, then so is the derived CP*$(k, 1; gp_1p_2)$.

In view of Lemmas 1.2–1.4, results on optimal or regular CP$(k, 1; v)$s can be used directly to produce optimal $(v, k, 1)$-OOCs. One can then expect that many of the constructions for optimal $(v, k, 1)$-OOCs are design theoretic in nature. This is indeed the case (see, for example, [2, 6, 14, 15]).

In the remainder of this paper, we will show that an optimal $(9p, 4, 1)$-OOC would exist for any prime $p = 4f + 1$ by way of a 9-regular CP$(4, 1; 9p)$. In the procedure, direct constructions with explicit codewords will be presented for the case $p \equiv 13 \bmod 24$, and Weil's theorem on character sums will be used to settle the other cases $p \equiv 1, 5, 17 \bmod 24$. Employing Lemmas 1.2–1.4, we can obtain, among others, new $(9v, 4, 1)$-OOC where v is a product of primes of form $4f + 1$.

2. Constructions. Let $p = 4f + 1$ be a prime and ω an arbitrary primitive root modulo p. By C_0^4 we denote the multiplicative subgroup of order f in Z_p, which is spanned by ω^4. The notations C_j^4 for $j = 0, 1, 2, 3$ stand for the cosets $\omega^j C_0^4$. These cosets are often referred to as the *cyclotomic classes* of index 4 of Z_p. They evidently partition $Z_p - \{0\}$.

Now we are going to describe our constructions, which are split into three cases depending on the values of p modulo 24.

2.1. The case $p \equiv 5 \bmod 24$. In this case, both 2 and 3 are quadratic nonresidue modulo p.

LEMMA 2.1. *Let p be a prime congruent to 5 modulo 24. If there exists an element x of Z_p such that $x \in C_1^4 \cup C_3^4$, $2(x-1) \in C_0^4$ and $x + 1 \in C_2^4$, then there exists a 9-regular CP$(4, 1; 9p)$.*

Proof. The desired CP$(4, 1; 9p)$ is obtained by taking the following $3(p-1)/4$ base blocks based on the additive group of $Z_p \times Z_9$, which is isomorphic to Z_{9p}:

$$\begin{gathered}\{(0,0),(2,0),(1,4),(-2,3)\}\cdot(g,1),\\ \{(-2,0),(2,1),(-1,2),(0,3)\}\cdot(g,1),\\ \{(x,0),(-x,0),(0,4),(-1,7)\}\cdot(g,1),\end{gathered}$$

where g runs over C_0^4.

Note that the differences from these base blocks can be partitioned into the following five classes depending on the values of the second component:

1. $\{(\pm 2, 0), (\pm 2x, 0)\}\cdot(C_0^4, 1)$;
2. $\{(\pm 3, 1), (4, 1), (1, 1)\}\cdot(C_0^4, \pm 1)$;
3. $\{(1, 2), (-2, 2), (x + 1, 2), (-x + 1, 2)\}\cdot(C_0^4, \pm 1)$;
4. $\{(\pm 2, 3), (-4, 3), (-1, 3)\}\cdot(C_0^4, \pm 1)$;
5. $\{(\pm 1, 4), (\pm x, 4)\}\cdot(C_0^4, \pm 1)$.

Since $p \equiv 5 \bmod 24$, we have $-1 \in C_2^4$. According to our choice of x, $2 \in C_1^4$ if and only if $x - 1 \in C_3^4$, and $2 \in C_3^4$ if and only if $x - 1 \in C_1^4$. Therefore, the above differences do cover each element of $Z_p \times Z_9 - \{0\} \times Z_9$ exactly once, while any element of the additive subgroup $\{0\} \times Z_9$ is not covered at all. The assertion then follows. □

We give an example to illustrate our construction.

EXAMPLE 2.2. *It is easy to see that 2 is a primitive root modulo 29 and*

$$\begin{aligned} C_0^4 &= \{1, 16, 24, 7, 25, 23, 20\}, \quad C_1^4 = \{2, 3, 19, 14, 21, 17, 11\},\\ C_2^4 &= \{4, 6, 9, 28, 13, 5, 22\}, \quad C_3^4 = \{8, 12, 18, 27, 26, 10, 15\}\end{aligned}$$

are the cyclotomic classes of Z_{29} *of index* 4. *Take* $x = 27$. *Then*

$$\{(0,0),(2,0),(1,4),(27,3)\},\ \{(27,0),(2,1),(28,2),(0,3)\},$$
$$\{(0,0),(3,0),(16,4),(26,3)\},\ \{(26,0),(3,1),(13,2),(0,3)\},$$
$$\{(0,0),(19,0),(24,4),(10,3)\},\ \{(10,0),(19,1),(5,2),(0,3)\},$$
$$\{(0,0),(14,0),(7,4),(15,3)\},\{(15,0),(14,1),(22,2),(0,3)\},$$
$$\{(0,0),(21,0),(25,4),(8,3)\},\ \{(8,0),(21,1),(4,2),(0,3)\},$$
$$\{(0,0),(17,0),(23,4),(12,3)\},\ \{(12,0),(17,1),(6,2),(0,3)\},$$
$$\{(0,0),(11,0),(20,4),(18,3)\},\ \{(18,0),(11,1),(9,2),(0,3)\},$$
$$\{(27,0),(2,0),(0,4),(28,7)\},\ \{(26,0),(3,0),(0,4),(13,7)\},$$
$$\{(10,0),(19,0),(0,4),(5,7)\},\{(15,0),(14,0),(0,4),(22,7)\},$$
$$\{(8,0),(21,0),(0,4),(4,7)\},\ \{(12,0),(17,0),(0,4),(6,7)\},$$
$$\{(18,0),(11,0),(0,4),(9,7)\}$$

are the base blocks of a 9*-regular* $CP(4,1;261)$.

As an immediate consequence of Lemma 2.1, we have the following result.

COROLLARY 2.3. *Let* p *be a prime congruent to* 5 *modulo* 24. *If the set*

$$T = \{x \in C_1^4 \cup C_3^4 : \{2(x-1), 4(x+1)\} \subseteq C_0^4\}$$

is not empty, then there exists a 9*-regular* $CP(4,1;9p)$.

Our main result of this subsection will be given by means of Corollary 2.3. We will make use of Weil's theorem on multiplicative character sums. This theorem has been useful in dealing with existence of various combinatorial structures such as triplewhist tournaments (see [1]).

A multiplicative character of a finite field $GF(q)$ is a homomorphism from the multiplicative group of $GF(q)$ into the multiplicative group of complex numbers of absolute value 1. The following is the statement of Weil's theorem on multiplicative character sums cited from Theorem 5.41 in [9]. In the theorem it is understood that if χ is a multiplicative character of $GF(q)$, then $\chi(0) = 0$.

THEOREM 2.4. *Let* χ *be a multiplicative character of* $GF(q)$ *of order* $m > 1$ *and let* f *be a polynomial of* $GF(q)[x]$ *which is not of the form* kg^m *for some* $k \in GF(q)$ *and some* $g \in GF(q)[x]$. *Then we have*

$$\left| \sum_{x \in GF(q)} \chi(f(x)) \right| \leq (d-1)\sqrt{q},$$

where d *is the number of distinct roots of* $f(x)$ *in its splitting field over* $GF(q)$.

For the problem under consideration, we need only a nonprinciple quartic character of the prime field Z_p with $p \equiv 5 \bmod 24$, which is a map from $Z_p - \{0\}$ into $\{\pm 1, \pm i\}$ defined by

$$\chi(x) = \begin{cases} 1 & \text{if } x \in C_0^4, \\ i & \text{if } x \in C_1^4, \\ -1 & \text{if } x \in C_2^4, \\ -i & \text{if } x \in C_3^4. \end{cases}$$

Consider the sum

$$A = \sum_{x \in Z_p} (1 - \chi(x^2)) \prod_{1 \leq i \leq 2} (1 + \chi(f_i(x)) + \chi(f_i^2(x)) + \chi(f_i^3(x))),$$

where $f_1(x) = 2(x-1)$ and $f_2(x) = 4(x+1)$.

It can be easily calculated that

$$1 - \chi(x^2) = \begin{cases} 2 & \text{if } x \in C_1^4 \cup C_3^4, \\ 0 & \text{if } x \in C_0^4 \cup C_2^4, \\ 1 & \text{if } x = 0; \end{cases}$$

and for $i = 1, 2$,

$$1 + \chi(f_i(x)) + \chi(f_i^2(x)) + \chi(f_i^3(x)) = \begin{cases} 4 & \text{if } f_i(x) \in C_0^4, \\ 0 & \text{if } f_i(x) \in Z_p - C_0^4 \cup \{0\}, \\ 1 & \text{if } f_i(x) = 0. \end{cases}$$

Therefore, the sum $A = 32|T| + d$ where T is the subset of Z_p defined in Corollary 2.3, and d is the contribution when either x or $f_1(x)$ or $f_2(x)$ is 0.

Now if $x = 0$, then $f_1(x) = -2 \in C_1^4 \cup C_3^4$ and $f_2(x) = 4 \in C_2^4$ and the contribution is 0. If $f_1(x) = 0$, which could happen only when $x = 1$, then the contribution to A is also 0. Similarly, if $f_2(x) = 0$, then $x = -1$ and hence the contribution to A is 0. Therefore we know that $d = 0$ and thus $A = 32|T|$.

To expand A, we define $S = \{0, 1\} \times \{0, 1, 2, 3\} \times \{0, 1, 2, 3\}$. For any triple $\mathbf{s} = (i, j, k) \in S$, we write

$$f_{\mathbf{s}}(x) = (-x^2)^i f_1^j(x) f_2^k(x).$$

Obviously, for any $\mathbf{s} \in S - \{(0, 0, 0)\}$, the polynomial $f_{\mathbf{s}}(x)$ is not of form ag^4 for any $a \in Z_p$ and any $g \in Z_p[x]$. When $\mathbf{s} = (0, 0, 0)$, we have $f_{\mathbf{s}}(x) = 1$, and hence $\sum_{x \in Z_p} \chi(f_{(0,0,0)}(x)) = p$. Therefore we obtain

$$A = p + \sum_{\mathbf{s} \in S - \{(0,0,0)\}} \sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x)),$$

which implies that

$$|A| \geq p - \sum_{\mathbf{s} \in S - \{(0,0,0)\}} \left| \sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x)) \right|.$$

Now for every $\mathbf{s} \in S - \{(0, 0, 0)\}$, we apply Theorem 2.4 to produce an upper bound. Suppose $j = 0$. If $k = 0$, then $i = 1$. In this case $|\sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x))| = 0$. If $k > 0$, then $f_{\mathbf{s}}(x) = (-1)^i 2^{2k} x^{2i} (x+1)^k$, which has one distinct root in its splitting field if $i = 0$ and two distinct roots in its splitting field if $i = 1$. From Theorem 2.4, we get

$$\sum_{i=0; j=0; k=1,2,3} \left| \sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x)) \right| + \sum_{i=1; j=0; k=1,2,3} \left| \sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x)) \right| \leq \sqrt{p} \cdot 3.$$

Suppose $j > 0$. If $k = 0$, then $f_{\mathbf{s}}(x) = (-1)^i 2^j x^{2i} (x-1)^j$, which has one distinct root in its splitting field if $i = 0$ and two distinct roots in its splitting field if $i = 1$. From Theorem 2.4, we get

$$\sum_{i=0; j=1,2,3; k=0} \left| \sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x)) \right| + \sum_{i=1; j=1,2,3; k=0} \left| \sum_{x \in Z_p} \chi(f_{\mathbf{s}}(x)) \right| \leq \sqrt{p} \cdot 3.$$

TABLE 2.1

(p,ω,x)	(p,ω,x)	(p,ω,x)	(p,ω,x)	(p,ω,x)
(29,2,27)	(53,2,51)	(101,2,29)	(149,2,52)	(173,2,129)
(197,2,46)	(269,2,8)	(293,2,291)	(317,2,98)	(389,2,132)
(461,2,38)	(509,2,28)	(557,2,189)	(653,2,22)	(677,2,448)
(701,2,699)	(773,2,503)	(797,2,481)	(821,2,129)	(941,2,129)
(1013,3,659)	(1061,2,313)			

If $k \neq 0$, then $f_{\mathbf{s}}(x) = (-1)^i 2^{j+2k} x^{2i}(x-1)^j(x+1)^k$, which has two distinct roots in its splitting field if $i = 0$ and three distinct roots in its splitting field if $i = 1$. From Theorem 2.4, we get

$$\sum_{i=0;j=1,2,3;k=1,2,3} \left| \sum_{x\in Z_p} \chi(f_{\mathbf{s}}(x)) \right| + \sum_{i=1;j=1,2,3;k=1,2,3} \left| \sum_{x\in Z_p} \chi(f_{\mathbf{s}}(x)) \right| \leq \sqrt{p}\cdot 3\cdot 3 + 2\sqrt{p}\cdot 3\cdot 3.$$

These inequalities yield that

$$|A| \geq p - 33\sqrt{p}.$$

It follows that $|A| > 0$ whenever $p > p_0 = 1089$ in the case when $p \equiv 5 \bmod 24$. This means that the set T defined in Corollary 2.3 is not empty whenever $p > p_0 = 1089$ and $p \equiv 5 \bmod 24$. With a computer search, we find the set T is also not empty for any prime $p \equiv 5 \bmod 24$ not greater than $p_0 = 1089$ with the only exception of $p = 5$. We list p, ω, and x in Table 2.1, where ω is the primitive root modulo p used in our computation. The results obtained above can be summarized into the main theorem of this subsection.

THEOREM 2.5. *Let p be a prime satisfying $p \equiv 5$ mod 24 and $p > 5$. Then there exists a 9-regular CP$(4, 1; 9p)$.*

We note that although there is no 9-regular CP$(4, 1; 45)$, there does exist a 9-regular CP$(4, 1; 225)$. This was pointed out to us by Dr. Gennian Ge of Suzhou University.

EXAMPLE 2.6. *The following 18 base blocks form a 9-regular CP$(4, 1; 225)$ over Z_{225}:*

$$\begin{array}{cc}
\{0,110,127,205\}, & \{0,7,76,87\},\\
\{0,174,211,223\}, & \{0,16,19,118\},\\
\{0,139,181,185\}, & \{0,41,79,113\},\\
\{0,96,164,220\}, & \{0,6,27,186\},\\
\{0,105,163,216\}, & \{0,81,82,173\},\\
\{0,22,162,177\}, & \{0,122,189,212\},\\
\{0,84,161,192\}, & \{0,73,97,132\},\\
\{0,43,71,131\}, & \{0,116,142,171\},\\
\multicolumn{2}{c}{\{0,8,18,65\},\ \{0,30,104,136\}.}
\end{array}$$

2.2. The case $p \equiv 13 \bmod 24$. We first note that when $p \equiv 13 \bmod 24$, 3 is a quadratic residue and 2 is a quadratic nonresidue modulo p. In this case, we take a primitive root ω modulo p so that $2 \in C_1^4$.

The following theorem gives direct constructions for 9-regular CP$(4, 1; 9p)$ with $p \equiv 13 \bmod 24$.

THEOREM 2.7. *For every prime $p \equiv 13 \bmod 24$, there exists a 9-regular CP$(4, 1; 9p)$.*

Proof. Let $Z_p \times Z_9$ be the point set. If 3 is a quartic residue modulo p, then we construct the following $3(p-1)/4$ base blocks $\mathcal{A}$ over the additive group of $Z_p \times Z_9$:

$$\begin{aligned}\{(-3,0),(-1,2),(1,3),(-2,7)\}\cdot(g,1),\\ \{(0,0),(3,0),(2,7),(4,1)\}\cdot(g,1),\\ \{(1,0),(-1,0),(7,5),(5,6)\}\cdot(g,1),\end{aligned}$$

where g runs over C_0^4.

Simple calculations in the additive group of $Z_p \times Z_9$ show that the differences from these base blocks are

$$\begin{aligned}\Delta\mathcal{A} = \{&(\pm 2,0),(\pm 3,0),(\pm 2,1),(4,1),(1,1),(\pm 2,2),(\pm 1,2),(\pm 4,3),\\ &(2,3),(-6,3),(-3,4),(1,4),(-6,4),(-8,4)\}\cdot(C_0^4,\pm 1).\end{aligned}$$

Noting that $3 \in C_0^4$, $2 \in C_1^4$, and $-1 \in C_2^4$, we can easily see that $\Delta\mathcal{A} = Z_p \times Z_9 - \{0\} \times Z_9$. Therefore the above construction gives a 9-regular CP$(4, 1; 9p)$ for the case when 3 is a quartic residue modulo p where $p \equiv 13 \bmod 24$.

For the case when 3 is a quadratic residue but not a quartic residue, the construction is similar. In this case, the desired $3(p-1)/4$ base blocks over $Z_p \times Z_9$ are

$$\begin{aligned}\{(0,0),(-2,1),(2,2),(4,3)\}\cdot(g,1),\\ \{(2,0),(-2,0),(0,3),(-4,4)\}\cdot(g,1),\\ \{(-3,0),(3,0),(-6,4),(6,7)\}\cdot(g,1),\end{aligned}$$

where $g \in C_0^4$. □

EXAMPLE 2.8. *Consider the case $p = 13$. It is easily seen that 2 is a primitive root modulo 13 and 3 is a quartic residue modulo 13. The multiplicative subgroup of Z_{13} with order 3 is $C_0^4 = \{1, 3, 9\}$. Then the following nine base blocks constitute a 9-regular CP$(4, 1; 117)$:*

$$\begin{aligned}&\{(10,0),(12,2),(1,3),(11,7)\},\{(4,0),(10,2),(3,3),(7,7)\},\\ &\{(12,0),(4,2),(9,3),(8,7)\},\ \{(0,0),(3,0),(2,7),(4,1)\},\\ &\{(0,0),(9,0),(6,7),(12,1)\},\{(0,0),(1,0),(5,7),(10,1)\},\\ &\{(1,0),(12,0),(7,5),(5,6)\},\{(3,0),(10,0),(8,5),(2,6)\},\\ &\{(9,0),(4,0),(11,5),(6,6)\}.\end{aligned}$$

EXAMPLE 2.9. *We now provide an example over the additive group of $Z_{37} \times Z_9$, where 3 is a quadratic residue but not a quartic residue modulo 37. 2 is a primitive root modulo 37. The multiplicative subgroup of Z_{37} with order 9 is $C_0^4 = \{1, 16, 34, 26, 9, 33, 10, 12, 7\}$. The following 27 base blocks constitute a 9-regular CP $(4, 1; 333)$:*

$$\begin{aligned}&\{(0,0),(-2,1),(2,2),(4,3)\},\ \{(2,0),(-2,0),(0,3),(-4,4)\},\\ &\{(0,0),(5,1),(32,2),(27,3)\},\ \{(32,0),(5,0),(0,3),(10,4)\},\\ &\{(0,0),(6,1),(31,2),(25,3)\},\ \{(31,0),(6,0),(0,3),(12,4)\},\\ &\{(0,0),(22,1),(15,2),(30,3)\},\ \{(15,0),(22,0),(0,3),(7,4)\},\\ &\{(0,0),(19,1),(18,2),(36,3)\},\ \{(18,0),(19,0),(0,3),(1,4)\},\end{aligned}$$

$$\{(0,0),(8,1),(29,2),(21,3)\},\ \{(29,0),(8,0),(0,3),(16,4)\},$$
$$\{(0,0),(17,1),(20,2),(3,3)\},\{(20,0),(17,0),(0,3),(34,4)\},$$
$$\{(0,0),(13,1),(24,2),(11,3)\},\{(24,0),(13,0),(0,3),(26,4)\},$$
$$\{(0,0),(23,1),(14,2),(28,3)\},\ \{(14,0),(23,0),(0,3),(9,4)\},$$
$$\{(-3,0),(3,0),(-6,4),(6,7)\},\{(26,0),(11,0),(15,4),(22,7)\},$$
$$\{(9,0),(28,0),(18,4),(19,7)\},\ \{(33,0),(4,0),(29,4),(8,7)\},$$
$$\{(10,0),(27,0),(20,4),(17,7)\},\{(12,0),(25,0),(24,4),(13,7)\},$$
$$\{(7,0),(30,0),(14,4),(23,7)\},\ \{(1,0),(36,0),(2,4),(35,7)\},$$
$$\{(16,0),(21,0),(32,4),(5,7)\}.$$

2.3. The case $p \equiv 1$ or 17 mod 24. We turn to the case where p is a prime such that $p \equiv 1$ or 17 mod 24, namely, $p \equiv 1 \bmod 8$.

First we note that there exists an explicit formula for $\sum_{x\in GF(q)}\chi(f(x))$ when q is odd and both χ and f are quadratic (see Theorem 5.48 of [9]).

THEOREM 2.10. *Let χ be the quadratic character of $GF(q)$ and let $f(x) = a_2x^2 + a_1x + a_0 \in GF(q)[x]$ with q odd and $a_2 \neq 0$. Then*

$$\sum_{x\in GF(q)} \chi(f(x)) = \begin{cases} -\chi(a_2) & \text{if } a_1^2 - 4a_0a_2 \neq 0, \\ (q-1)\chi(a_2) & \text{otherwise.} \end{cases}$$

For our purpose, we need only quadratic characters of prime fields Z_p with $p \equiv 1 \bmod 8$. Recall that the quadratic character of Z_p is the map from $Z_p - \{0\}$ into $\{1,-1\}$ defined by

$$\chi(x) = \begin{cases} 1 & \text{if } x \in C_0^4 \cup C_2^4, \\ -1 & \text{if } x \in C_1^4 \cup C_3^4. \end{cases}$$

As is usually done, we adopt the convention that $\chi(0) = 0$.

The following lemma gives us the construction for 9-regular CP$(4,1;9p)$ with $p \equiv 1 \bmod 8$.

LEMMA 2.11. *Let p be a prime and $p \equiv 1 \bmod 8$. If there exists an element $x \in C_1^4 \cup C_3^4$ such that $x^2 - 1 \in C_1^4 \cup C_3^4$ and $x^2 + 1 \in C_0^4 \cup C_2^4$, then there exists a 9-regular CP$(4,1;9p)$.*

Proof. Let $p = 8s + 1$. Take $Z_p \times Z_9$ as our point set. Let $R = \{1, \omega^4, \ldots, \omega^t\}$, where ω is a primitive root modulo p and $t = 4(s-1)$. Now form the following base block families over $Z_p \times Z_9$:

$$\mathcal{A}_1 = \{\{(x,0),(x^{-1},0),(x+x^{-1},5),(0,5)\}\cdot(r,1) : r \in R\},$$
$$\mathcal{A}_2 = \{\{(x\omega^2,0),(x^{-1}\omega^2,0),((x+x^{-1})\omega^2,3),(0,3)\}\cdot(r,1) : r \in R\},$$
$$\mathcal{A}_3 = \{\{(-x^2,0),(x^2,1),(1,3),(-1,8)\}\cdot(r,1) : r \in R\},$$
$$\mathcal{A}_4 = \{A\cdot(\omega^2,1) : A \in \mathcal{A}_3\},$$
$$\mathcal{A}_5 = \{A\cdot(-1,1) : A \in \mathcal{A}_3\},$$
$$\mathcal{A}_6 = \{A\cdot(-\omega^2,1) : A \in \mathcal{A}_3\}.$$

The differences from the above base block families are

$$\Delta(\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_6)$$

$$
\begin{aligned}
&= \{\{(x-x^{-1},0),(x+x^{-1},0),(\omega^2(x-x^{-1}),0),(\omega^2(x+x^{-1}),0),\\
&\quad (2x^2,\pm1),(x^2-1,\pm1),(2\omega^2x^2,\pm1),(\omega^2(x^2-1),\pm1),\\
&\quad (x^2-1,\pm2),(x^2+1,\pm2),(\omega^2(x^2-1),\pm2),(\omega^2(x^2+1),\pm2),\\
&\quad (\omega^2x^{-1},\pm3),(\omega^2x,\pm3),(x^2+1,\pm3),(\omega^2(x^2+1),\pm3),\\
&\quad (x^{-1},\pm4),(x,\pm4),(2,\pm4),(2\omega^2,\pm4)\}\cdot(\pm r,1) : r\in R\}.
\end{aligned}
$$

Since $\{x, x^2-1\} \subseteq C_1^4 \cup C_3^4$ and $x^2+1 \in C_0^4 \cup C_2^4$, we have $x-x^{-1} = (x^2-1)/x \in C_0^4 \cup C_2^4$ and $x + x^{-1} = (x^2+1)/x \in C_1^4 \cup C_3^4$. Also, 2 is a quadratic residue modulo p as $p \equiv 1 \bmod 8$. According to the definition of R, we also have $R \cup (-R) = C_0^4$. Further, it is obvious that $x \in C_1^4$ if and only if $x^{-1} \in C_3^4$ and $x \in C_3^4$ if and only if $x^{-1} \in C_1^4$. From these observations, we can know that the above $6s = 3(p-1)/4$ base blocks yield the desired 9-regular CP$(4, 1; 9p)$. □

EXAMPLE 2.12. *In the case $p = 17$, 3 is a primitive root modulo 17, and $R = \{1, 13\}$. The following 12 base blocks form a 9-regular CP$(4, 1; 153)$:*

$$
\begin{aligned}
&\{(10,0),(12,0),(5,5),(0,5)\},\{(11,0),(3,0),(14,5),(0,5)\},\\
&\{(5,0),(6,0),(11,3),(0,3)\},\{(14,0),(10,0),(7,3),(0,3)\},\\
&\{(2,0),(15,1),(1,3),(16,8)\},\ \{(9,0),(8,1),(13,3),(4,8)\},\\
&\{(1,0),(16,1),(9,3),(8,8)\},\{(13,0),(4,1),(15,3),(2,8)\},\\
&\{(15,0),(2,1),(16,3),(1,8)\},\ \{(8,0),(9,1),(4,3),(13,8)\},\\
&\{(16,0),(1,1),(8,3),(9,8)\},\{(4,0),(13,1),(2,3),(15,8)\}.
\end{aligned}
$$

What we need to do at this moment is to show that there exists at least one such element x which satisfies the conditions stated in Lemma 2.11.

Let us consider the sum

$$B = \sum_{x\in Z_p} (1-\chi(x))(1-\chi(x^2-1))(1+\chi(x^2+1)).$$

We have

$$1-\chi(x) = \begin{cases} 2 & \text{if } x \in C_1^4 \cup C_3^4, \\ 0 & \text{if } x \in C_0^4 \cup C_2^4, \\ 1 & \text{if } x = 0; \end{cases}$$

$$1-\chi(x^2-1) = \begin{cases} 2 & \text{if } x^2-1 \in C_1^4 \cup C_3^4, \\ 0 & \text{if } x^2-1 \in C_0^4 \cup C_2^4, \\ 1 & \text{if } x^2-1 = 0; \end{cases}$$

$$1+\chi(x^2+1) = \begin{cases} 2 & \text{if } x^2+1 \in C_0^4 \cup C_2^4, \\ 0 & \text{if } x^2+1 \in C_1^4 \cup C_3^4, \\ 1 & \text{if } x^2+1 = 0. \end{cases}$$

Similarly to section 2.1, it is easily calculated that $B = 8e$, where e is the number of elements x in Z_p which satisfies the conditions in Lemma 2.11.

Expanding B we obtain

$$
\begin{aligned}
B = &\sum_{x\in Z_p} 1 - \sum_{x\in Z_p} \chi(x) - \sum_{x\in Z_p} \chi(x^2-1) + \sum_{x\in Z_p} \chi(x^2+1)\\
&+ \sum_{x\in Z_p} \chi(x^3-x) - \sum_{x\in Z_p} \chi(x^3+x) - \sum_{x\in Z_p} \chi(x^4-1) + \sum_{x\in Z_p} \chi(x^5-x).
\end{aligned}
$$

TABLE 2.2

(p,ω,x)	(p,ω,x)	(p,ω,x)	(p,ω,x)	(p,ω,x)
(17,3,10)	(41,6,6)	(73,5,45)	(89,3,51)	(97,5,28)
(113,3,76)				

Clearly, $\sum_{x\in Z_p} 1 = p$ and $\sum_{x\in Z_p}\chi(x) = 0$.

By Theorem 2.10, we have

$$\sum_{x\in Z_p} \chi(x^2 \pm 1) = -1.$$

By Theorem 2.4, we also have

$$\left|\sum_{x\in Z_p} \chi(x^3 - x)\right| \le 2\sqrt{p},$$

$$\left|\sum_{x\in Z_p} \chi(x^3 + x)\right| \le 2\sqrt{p},$$

$$\left|\sum_{x\in Z_p} \chi(x^4 - 1)\right| \le 3\sqrt{p},$$

$$\left|\sum_{x\in Z_p} \chi(x^5 - x)\right| \le 4\sqrt{p}.$$

It follows that $|B| \ge p - 11\sqrt{p}$ which implies that $|B| > 0$ if $p > 121$. This means that there exists at least one element x in Z_p which satisfies the conditions stated in Lemma 2.11 whenever $p > p_1 = 121$ and $p \equiv 1 \bmod 8$.

The only primes congruent to 1 modulo 8 which are not greater than 121 are 17, 41, 73, 89, 97, and 113. For each of these six primes, an element x satisfying the conditions in Lemma 2.11 can be easily found by hand, which we list in Table 2.2. The results obtained above can be summarized into the following theorem.

THEOREM 2.13. *For every prime $p \equiv 1$ mod 8, there exists a 9-regular CP$(4,1;9p)$.*

3. New optimal optical orthogonal codes. The results contained in Theorems 2.5, 2.7, and 2.13 can be summarized into the following theorem.

THEOREM 3.1. *There exists a 9-regular CP$(4,1;9p)$ for any prime $p \equiv 1$ mod 4 and $p > 5$.*

Using Theorem 3.1 in conjunction with Lemma 1.4, we have the following result.

THEOREM 3.2. *There exists a 9-regular CP$(4,1;9v)$ for any positive integer v whose prime factors are all congruent to 1 modulo 4 and greater than 5.*

Combining Examples 1.1 and 2.6, Lemmas 1.2 and 1.3, and Theorem 3.2, we can obtain new optimal $(v,4,1)$-OOCs. In particular, we have the following main result of this paper.

THEOREM 3.3. *Let v be a positive integer whose prime factors are all congruent to 1 modulo 4. Then there exists an optimal $(9v,4,1)$-OOC.*

Acknowledgments. A portion of this research was carried out while the third author was visiting the University of Tsukuba. He wishes to express many thanks to the Institute of Policy and Planning Sciences for their hospitality. The authors would also like to thank Dr. Gennian Ge of Suzhou University for providing them with a 9-regular CP$(4, 1; 225)$.

REFERENCES

[1] M. Buratti, *Existence of Z-cyclic triplewhist tournaments for a prime number of players,* J. Combin. Theory Ser. A, 90 (2000), pp. 315–325.

[2] K. Chen, G. Ge, and L. Zhu, *Starters and their related codes,* J. Statist. Plann. Inference, 86 (2000), pp. 379–395.

[3] F. R. K. Chung, J. A. Salehi, and V. K. Wei, *Optical orthogonal codes: Design, analysis, and applications,* IEEE Trans. Inform. Theory, 35 (1989), pp. 595–604; Erratum: IEEE Trans. Inform. Theory, 38 (1992), p. 1429.

[4] C. J. Colbourn and J. H. Dinitz, eds., *The CRC Handbook of Combinatorial Designs,* CRC Press, Boca Raton, FL, 1996.

[5] C. J. Colbourn, J. H. Dinitz, and D. R. Stinson, *Applications of Combinatorial Designs to Communications, Cryptography, and Networking*, London Math. Soc. Lecture Note Ser. 267, Cambridge University Press, Cambridge, UK, 1999, pp. 37–100.

[6] R. Fuji-Hara and Y. Miao, *Optical orthogonal codes: Their bounds and new optimal constructions,* IEEE Trans. Inform. Theory, 46 (2000), pp. 2396–2406.

[7] M. Hall, Jr., *Combinatorial Theory*, 2nd ed., Wiley, New York, 1986.

[8] S. M. Johnson, *A new upper bound for error-correcting codes,* IEEE Trans. Inform. Theory, 8 (1962), pp. 203–207.

[9] R. Lidl and H. Niederreiter, *Finite Fields,* Cambridge University Press, Cambridge, UK, 1997.

[10] S. V. Maric and V. K. N. Lau, *Multirate fiber-optic CDMA: System design and performance analysis,* J. Lightwave Technol., 16 (1998), pp. 9–17.

[11] S. V. Maric, O. Moreno, and C. Corrada, *Multimedia transmission in fiber-optic LANs using optical CDMA,* J. Lightwave Technol., 14 (1996), pp. 2149–2153.

[12] J. A. Salehi, *Code division multiple-access techniques in optical fiber networks – part* I: *Fundamental principles,* IEEE Trans. Commun., 37 (1989), pp. 824–833.

[13] J. A. Salehi and C. A. Brackett, *Code division multiple access techniques in optical fiber networks – part* II: *Systems performance analysis,* IEEE Trans. Commun., 37 (1989), pp. 834–842.

[14] J. Yin, *Some combinatorial constructions for optical orthogonal codes,* Discrete Math., 185 (1998), pp. 201–219.

[15] J. Yin, *A direct construction for optimal* $(12p, 4, 1)$ *optical orthogonal codes,* submitted, 2000.

SIAM J. DISCRETE MATH.
Vol. 14, No. 2, pp. 267–282

ON THE OPTIMALITY OF GENERAL LOWER BOUNDS FOR BROADCASTING AND GOSSIPING *

MICHELE FLAMMINI† AND STÉPHANE PÉRENNÈS‡

Abstract. In this paper we show that many general lower bounds on the broadcasting and gossiping time are optimal. In particular, let $b(G)$ be the broadcasting time of a network G under the basic one-port model. The only lower bound on $b(G)$ holding for every n vertices graph G is $\max(\log_2 n, Diam(G))$, but the $\log_2 n$ factor cannot be achieved in bounded degree networks. In fact, let the parameter d be defined in undirected graphs as the maximum degree minus one and for directed graphs as the maximum out-degree. Then, in [*SIAM J. Discrete Math.*, 1 (1998), pp. 531–540; *SIAM J. Discrete Math.*, 5 (1992), pp. 10–24] it has been proved that, for any graph G of parameter d, $b(G) \geq \frac{\log_2 n}{\log_2 \xi}$, where ξ is the largest real number such that $\xi^d - \xi^{d-1} - \xi^{d-2} - \cdots - \xi - 1 = 0$. Since then many papers have proposed constructions of bounded degree networks having a small broadcast time [*Proceedings of the 2nd International Euro-Par Conference (EUROPAR)*, Lecture Notes in Comput. Sci. 1123, Springer-Verlag, New York, 1996, pp. 313–324; IEEE Trans. Comput., 33 (1984), pp. 190–194], but so far the optimality of [*SIAM J. Discrete Math.*, 1 (1998), pp. 531–540; *SIAM J. Discrete Math.*, 5 (1992), pp. 10–24] was still an open question.

In this paper we prove that the above lower bound is tight, improving all the existing upper bounds by means of probabilistic methods. Namely, we show that for n arbitrarily large there exist families of n vertices graphs in which a uniformly drawn graph has broadcasting time as predicted by [*SIAM J. Discrete Math.*, 1 (1998), pp. 531–540; *SIAM J. Discrete Math.*, 5 (1992), pp. 10–24] with probability converging to 1. Moreover, we show that [*SIAM J. Discrete Math.*, 1 (1998), pp. 531–540; *SIAM J. Discrete Math.*, 5 (1992), pp. 10–24] is attained even in the case of gossiping and systolic gossiping in the full-duplex mode.

Finally, new upper bounds on bounded-degree and systolic gossiping are also determined in the directed and half-duplex modes. While the systolic construction is tight and matches the lower bound of [*Inform and Comput.*, to appear], we strongly conjecture that the bounded-degree result is optimal and that a corresponding matching lower bound is still to be proven.

Key words. broadcasting, gossiping, bounded-degree, systolic, optimal probabilistic constructions

AMS subject classifications. 68M10, 05C80, 68R10, 68R05

PII. S0895480199365397

1. Introduction.

Broadcasting (one-to-all) and gossiping (all-to-all) primitives for disseminating information in communication networks have been extensively investigated in recent years for many different topologies and under a large variety of models [11, 1, 10, 6, 14, 12, 13, 2].

In this paper we consider the basic model, called one-port or whispering, where at each communication round each processor can have only one active incident link, i.e., the set of the active links forms a matching. Active links are used at the corresponding processors to deliver the items known until that communication round to their neighbors. If the network can be modeled as an undirected graph, for gossiping

*Received by the editors November 30, 1999; accepted for publication (in revised form) January 10, 2001; published electronically April 6, 2001. This work was supported by the IST Programme of the EU under contract IST-1999-14186 (ALCOM-FT), by the SLOOP project I3S-CNRS/INRIA/Université de Nice-Sophia Antipolis and by the Italian "Progetto Cofinanziato: Resource Allocation in Computer Networks."

http://www.siam.org/journals/sidma/14-2/36539.html

†Dipartimento di Matematica Pura ed Applicata, Università di L'Aquila, via Vetoio Loc. Coppito, I-67010 L'Aquila, Italy (flammini@univaq.it).

‡SLOOP I3S-CNRS/INRIA/Université de Nice-Sophia Antipolis, 2004 Route des Lucioles, BP 93, F-06902 Sophia-Antipolis Cedex, France (speren@sophia.inria.fr).

it is possible to further distinguish between two different cases: the half-duplex mode, in which active links allow the transmission of messages only in one direction, and the full-duplex mode, in which messages can travel in both directions simultaneously. Clearly such a distinction is meaningless for broadcasting, as there is always a single item traveling around and it traverses each link only once and in one direction.

There are many positive and negative results for specific networks under the one-port model (see, for instance, [6, 10, 15, 17, 14, 26, 25, 2, 8, 9]).

The only lower bound on the broadcasting time holding for every topology is $\max(\log n, Diam(G))$ (from now on all logarithms are assumed in base 2), but it can be improved for bounded-degree networks. In fact, let the parameter d be defined for undirected graphs as the maximum degree minus one and for directed graphs as the maximum out-degree. Then, in [22, 3] it has been proved that $b(G)$ in a graph G of n vertices and parameter d satisfies $b(G) \geq e(d) \log n$, where $e(d) = \frac{1}{\log \xi}$ and ξ is the largest real number such that $\xi^d - \xi^{d-1} - \xi^{d-2} - \cdots - \xi - 1 = 0$. This yields $e(2) = 1.4404$, $e(3) = 1.1374$, $e(4) = 1.0562$, and for large d $e(d) \approx (1 + (\log e)/2^d)$.

Concerning upper bounds, in graphs of parameter 2 there is protocol terminating in about $1.444 \log n$ steps for directed de Bruijn networks [4] and $1.47 \log n$ steps for undirected cubic graphs [16]. For higher values of the parameter d the known upper bounds differ significantly from the lower bounds in [22, 3] and until this paper the optimality of [22, 3] was an open question.

For gossiping, in the half-duplex mode there is a general lower bound of $1.4404 \log n$ holding for all graphs of n vertices [7, 21, 19, 27]. Such a lower bound has been generalized in [8], where it has been shown that any s-systolic (i.e., periodic with period s) gossip protocol in the directed and half-duplex modes for any graph takes at least $g(s)(\log n)$ time steps, where $g(s) = 1/(\log 1/\lambda)$ and λ is the unique real number such that $0 < \lambda < 1$ and $\sqrt{p_{\lfloor s/2 \rfloor}(\lambda)} \cdot \sqrt{p_{\lceil s/2 \rceil}(\lambda)} = 1$, with $p_j(\lambda) = \lambda + \lambda^3 + \cdots + \lambda^{2j-1}$ for any integer $j > 0$. Thus, for instance $g(4) = 1.8133$, $g(5) = 1.6502$, $g(6) = 1.5363$, $g(7) = 1.5021$, $g(8) = 1.4721$, and $g(\infty) = 1.4404$, i.e., for nonsystolic protocols (infinite period) coincides with [7, 21, 19, 27]. Finally, in [9] it has been proved that in the directed and half-duplex modes gossiping in any graph G of parameter d requires at least $g(d) \log n$ time steps, where $g(d) = 1/(\log 1/\lambda)$ and λ is the unique real number such that $0 < \lambda < 1$ and $\sqrt{p_\infty(\lambda)} \cdot \sqrt{p_d(\lambda)} = 1$, again with $p_j(\lambda) = \lambda + \lambda^3 + \cdots + \lambda^{2j-1}$ for any integer $j > 0$ and $p_\infty(\lambda) = \lim_{j \to \infty} p_j(\lambda) = \lambda/(1 - \lambda^2)$. This gives $g(2) = 1.5728$, $g(3) = 1.4829$, $g(4) = 1.4555$, and so forth with $g(\infty) = 1.4404$ as in [7, 21, 19, 27]. Besides the cases mentioned above, the other general lower bounds are those that can be inferred from broadcasting.

Concerning upper bounds, the general lower bound of $1.4404 \log n$ in the half-duplex mode proved in [7, 21, 19, 27] is attained for complete graphs [7]. All the other known upper bounds for specific networks do not match [8, 9].

In this paper we first prove that the broadcasting lower bound of [22, 3] is tight. Namely, we show that, for n arbitrarily large, there exist families of n vertices graphs such that a graph drawn uniformly in the family with probability converging to 1 has a broadcasting time matching [22, 3]. Moreover, by carefully selecting the dissemination protocol, we show that the same time holds even in the case of gossiping and systolic gossiping in the full-duplex mode.

Finally, in the directed and half-duplex modes, we provide efficient probabilistic constructions also for bounded-degree and systolic gossiping. While the systolic upper bound is tight and matches the lower bound of [8], we also strongly conjecture that

the bounded-degree result is optimal and that a corresponding matching lower bound improving [9] is still to be proven.

The paper is organized as follows. In the next section we introduce the notation and necessary definitions. In section 3 we show our tight upper bounds for broadcasting in bounded-degree networks. In section 4 we provide the new upper bounds for directed and half-duplex gossiping and we show that they are tight for systolic protocols. Finally, in section 5, we give some conclusive remarks and discuss some open questions.

2. Notation and definitions. Let us first introduce some useful notation and definitions.

We model the network as a digraph $G = (V, A)$ in which vertices represent processors and arcs communication links. The parameter d of G is the maximum out-degree of a vertex if G is not symmetric and the maximum degree minus one if G is symmetric (or undirected).

DEFINITION 2.1. *A broadcast (resp., gossip) protocol of length t for $G = (V, A)$ is a sequence $\langle A_1, \ldots, A_t \rangle$ of t subsets $A_1, \ldots, A_t \subseteq A$ subject to the following conditions:*

1. *each A_i, $1 \leq i \leq t$, is a matching in G (i.e., no two arcs in A_i have a common endpoint);*
2. *for a given root vertex $x \in V$ (resp., for any vertex $x \in V$) and for any other vertex $y \in V$, there exists a directed path $\langle x_0, x_1, \ldots, x_l \rangle$ with $l \leq t$, $x_0 = x$, and $x_l = y$, and a sequence of positive integers $j_1, \ldots, j_l$ such that $1 \leq j_1 < \cdots < j_l \leq t$ and for every i, $1 \leq i \leq l$, (x_{i-1}, x_i) belongs to A_{j_i}.*

Informally, each A_i represents the set of arcs that are active at the communication round i. If an arc (x, y) is active at a step i, then at the beginning of step $i+1$ vertex y knows all the items known by x at the beginning of step i. Then, in order for the sequence of the subsets A_i to be a broadcast (resp., gossip) procedure, starting from the chosen root x (from any vertex x), for any other vertex y there must exist a direct path from x to y whose arcs are activated in a proper sequence so that at the end of the protocol y knows the item of x.

DEFINITION 2.2. *The broadcasting time $b(G, x)$ of $G = (V, A)$ from a root vertex $x \in V$ is the minimum length of a broadcast protocol in G from x; the broadcasting time of G is $b(G) = \max_{x \in V} b(G, x)$.*

The gossiping time $g(G)$ of G is the minimum length of a gossiping protocol for G.

If we restrict our attention to symmetric digraphs, then the above definitions corresponds to half-duplex protocols. In order to obtain the full-duplex case, it is sufficient to weaken the condition on active arcs by saying that at every communication round any two active arcs (x, y) and (x', y') are such that either $\{x, y\} \cap \{x', y'\} = \emptyset$ or $x = y'$ and $y = x'$; that is, either they don't have a common endpoint or they are opposite. Clearly such a distinction is meaningless for broadcasting, as there is always a single item traveling around and it traverses each link only once and in one direction.

Periodic or "systolic" protocols have received particular attention in the past years because of their cheap realization due to a very regular, synchronized periodic behavior of all processors during the dissemination of information (see [23, 24, 15, 18, 20]). In a systolic protocol communication rounds are repeated in a periodic fashion.

DEFINITION 2.3 (see [15]). *A protocol $\langle A_1, \ldots, A_t \rangle$ for $G = (V, A)$ is s-systolic if for any i, $1 \leq i \leq t - s$, $A_i \equiv A_{i+s}$.*

A definition analogous to Definition 2.2 can also be given in the systolic case.

DEFINITION 2.4. *The systolic broadcasting time $b_s(G,x)$ of $G=(V,A)$ from a root vertex $x \in V$ is the minimum length of an s-systolic broadcast protocol in G from x; the systolic broadcasting time of G is $b_s(G) = \max_{x\in V} b_s(G,x)$.*

The systolic gossiping time $g_s(G)$ of G is the minimum length of an s-systolic gossiping protocol for G.

While broadcasting protocols can be systolized at no cost when the parameter d of G is such that $s \geq d+1$ [15], i.e., $b_s(G) = b(G)$, in general for gossiping $g_s(G) \geq g(G)$.

3. Upper bounds for broadcasting. In this section we show the existence of fixed parameter networks whose broadcasting time matches the lower bound of [22, 3], thus proving that it is optimal. Namely, by means of probabilistic arguments, for n arbitrarily large we provide a family of n vertices graphs such that a graph drawn uniformly in the family with probability converging to 1 has broadcasting time matching [22, 3]. Before going through the details of our construction, let us first briefly recall the basic idea leading to the lower bound of [22, 3] when the parameter of the graph G is $d = 2$.

Given any broadcast protocol for G and a root $r \in V$, let us denote as $s(r,i)$ the number of new vertices that receive the item of r during round i. Then, $s(r,i)$ obeys the recurrence $s(r,i) \leq s(r,i-1) + s(r,i-2)$ with $s(r,0) = 1$ and $s(r,1) = 1$, which naturally arises by observing that every vertex, upon receiving the item of r from one of its neighbors, in the best case can inform two other neighbors in the next two rounds. Solving the above recurrence we have that $s(r,i) = O(\xi^i)$, where ξ is the largest root of the equation $\gamma^2 - \gamma - 1 = 0$ (i.e., $\xi = (1+\sqrt{5})/2$ is the golden ratio), and thus $s(r,i)$ is at most proportional to ξ^i up to a constant factor depending on the initial conditions. Moreover, the total number of vertices knowing the item of r at the end of round i is also $c(r,i) = \sum_{j=0}^{i} s(r,j) = O(\xi^i)$, since $\sum_{j=0}^{i} \xi^j = (\xi^{i+1}-1)/(\xi-1)$. The lower bound on the broadcasting time t is then inferred by observing that it must be $c(r,t) \geq n$, thus yielding $b(G) \geq (\log_\xi n) - O(1) = (1.4404 \log n) - O(1)$.

In order to determine a matching upper bound one should find a graph and a protocol such that, whenever a vertex is informed by a neighbor, two other neighbors have not yet been informed and are informed in the next two rounds. In fact this would give $s(r,i) = s(r,i-1)+s(r,i-2)$ and in other words it is equivalent to say that during the broadcast protocol, if every vertex exploits its full dissemination capability by always sending the known item of information to two neighbors in the successive two steps, then no vertex is informed twice (or more) and thus the optimal growth behavior is maintained.

We now provide a class of graphs with the property that a randomly chosen graph in the class has a broadcast protocol that satisfies the recurrence $s(r,i) \approx s(r,i-1) + s(r,i-2)$ with probability converging to 1 as n increases, which means that very few vertices are informed twice during the protocol. As a consequence, almost all the graphs in the family have a broadcasting time matching the lower bound of [22, 3].

In order to deal with all the cases, we consider separately undirected (or directed symmetric) graphs and directed graphs.

3.1. Undirected graphs. In this section we consider undirected graphs or symmetric digraphs where every edge corresponds to a pair of symmetric directed arcs between its two endpoints. In order to simplify matters, we first give a simpler proof for the case of cubic graphs, that is, when each vertex has degree at most 3 (and thus parameter 2), and then we show how to extend it to any fixed parameter d.

Our construction exploits ideas in [5] and uses a cycle of n vertices with n even plus a random matching. In the family obtained by the graphs with all possible matchings, uniformly extracting a graph corresponds to choosing any possible matching with equal probability.

In any graph thus constructed it is possible to identify three disjoint matchings. Two matchings M_1 and M_2 are given by the (alternate) edges along the cycle, while the third matching M_3 is the randomly chosen one (see Figure 3.1). It is then possible to fix the following protocol: M_1, M_2 and M_3 are activated cyclically in the order; that is, the protocol $\langle A_1, \ldots, A_t \rangle$ is such that $A_1 = M_1$, $A_2 = M_2$, $A_3 = M_3$, and $A_{i+3} = A_i$ for each $1 \leq i \leq t-3$.

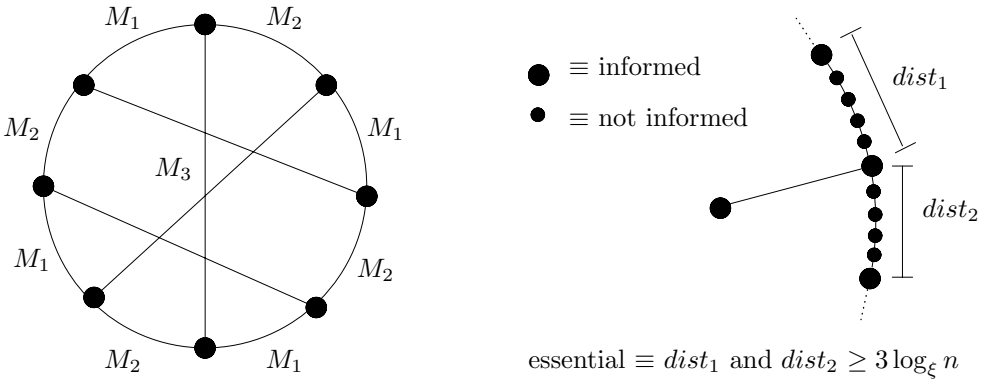


FIG. 3.1. *The cubic random graph and essential chords.*

Let $S(r,i)$ be the subset of the new vertices informed during round i, $C(r,i)$ the subset of all the vertices informed at the end of round i, and $s(r,i)$ and $c(r,i)$ their respective cardinalities. By the lower bound considerations at the beginning of this section, for any chosen root vertex r in G, $s(r,i) = O(\xi^i)$ and $c(r,i) = O(\xi^i)$, $1 \leq i \leq t$.

In order to get the claim we now prove that the following events hold with probability $1 - o(1/n)$:

- [E1] $s(r,i) \geq s(r,i-1) + s(r,i-2) - 1$ for every $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$;
- [E2] $s(r,i)(1+\frac{1}{\log_\xi n}) \geq s(r,i-1)+s(r,i-2)$ for every i such that $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$;
- [E3] after step $\lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil$ any uninformed vertex in G is informed in at most $O(\log \log n)$ additional rounds.

As we will see, these events together guarantee that the claimed upper bound holds with probability $1 - o(1)$. Notice that the distinction between events E1 and E2 is needed to avoid the collapsing of the recurrences, as at every step the number of vertices informed twice must be a small fraction of all the informed ones.

Following the approach used by Chung and Bollobás in [5], we want to avoid a priori counting vertices twice. If we call chords the edges belonging to the random matching M_3, then this is accomplished by "removing" the *unessential* chords which can lead to informing a vertex more than once. Unessential chords can be identified as follows. Assume that the chords of M_3 are chosen step by step during the rounds in which M_3 is activated, starting the process from the root r with a graph consisting

in a cycle. An informed vertex is involved in the process during the activation of M_3 when it is not yet adjacent to a chord; in this case it chooses randomly (and equiprobably) among the vertices of degree 2 one target y and then the chord $\{x, y\}$ is added to the graph. Since many vertices are involved in the process at the same round, their adjacent chords are added to the cycle in any sequential order. This probabilistic space is clearly equivalent to the random matching one, as it consists of drawing the matching sequentially instead of in a single parallel step.

A chord $\{x, y\} \in M_3$ added during round i is *unessential* at round i if there exists a vertex y' at distance less than $3\log_\xi n$ from y along the cycle such that y' is either an already informed vertex or $\{x', y'\}$ is a previously added chord during the round (according to the chosen sequential order used to add chords). Since before time $\log_\xi n$ a vertex using the matchings M_1 and M_2 can inform only the vertices at distance less than $\log_\xi n$ along the cycle, if we perform an auxiliary estimation in which we count only the vertices informed using only essential chords, then all the vertices that are informed before time $\log_\xi n$ cannot be counted twice (see Figure 3.1).

Let $SE(r, i)$ be defined as the subset of the new informed vertices during round i if only essential chords are used, $CE(r, i)$ as the subset of all the vertices informed at the end of round i again using only essential chords, and $se(r, i)$ and $ce(r, i)$ be their respective cardinalities. Clearly, $s(r, i) \geq se(r, i)$ and $c(r, i) \geq ce(r, i)$, $1 \leq i \leq t$.

By the considerations above, if $i < \log_\xi n$ and round i corresponds to the activation of M_1, then all the vertices of $SE(r, i-2)$ (informed at round $i-2$ using M_2) and of $SE(r, i-1)$ (informed at round $i-1$ using essential chords of M_3) are such that their neighbors reached through M_1 are not informed. Moreover, the neighbors along M_1 of all the vertices informed before round $i-2$ are all informed. Therefore, $se(r, i) = se(r, i-1) + se(r, i-2)$. Similarly, if round i corresponds to the activation of M_2, $se(r, i) = se(r, i-1) + se(r, i-2)$, while if it corresponds to the activation of M_3 and $un(i)$ is the number of unessential chords during round i, $se(r, i) = se(r, i-1) + se(r, i-2) - un(i)$.

Hence, in order to determine the growth rate of $se(r, i)$, it is sufficient to properly bound $un(i)$ in any given round i.

LEMMA 3.1. *With probability* $1 - o(1/n)$, $se(r, i) \geq se(r, i-1) + se(r, i-2) - 1$ *for every* $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$.

Proof. It is sufficient to show that the probability that there is more than one unessential chord in any round $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$ is $o(1/n)$.

A particular chord added during round i is unessential with probability at most $\frac{6(\log_\xi n)c(r,i)}{n-c(r,i)}$. In fact, let U be the set of all the vertices informed at the end of the previous round, that is, in $CE(r, i-1)$, or informed by the previously added chords during the round (according to the chosen sequential order used to add chords). Clearly, $U \subseteq CE(r, i)$ and thus $|U| \leq ce(r, i) \leq c(r, i)$ (recall that $c(r, i)$ is the number of all the informed vertices at the end of round i, even using unessential chords). As stated above, starting from an informed vertex x, the chord is obtained choosing randomly and equiprobably among the vertices not in U one target y. This gives rise to at least $n - |U| \geq n - c(r, i)$ possible choices, of which at most $6(\log_\xi n)c(r, i)$ corresponding to unessential chords. In fact, the target vertices yielding unessential chords are those at distance less than $3\log_\xi n$ from the vertices in U along the cycle, that is, at most $(6(\log_\xi n) - 1)|U| < 6(\log_\xi n)c(r, i)$.

Since we add at most $ce(r, i) \leq c(r, i)$ chords at round i, the probability of finding at least one unessential chord at round i is at most $c(r, i)\frac{6(\log_\xi n)c(r,i)}{n-c(r,i)}$. In the same

way, the probability that at least two chords are unessential at round i is

$$p_i \leq \binom{c(r,i)}{2}\left(\frac{6(\log_\xi n)c(r,i)}{n-c(r,i)}\right)^2$$

$$= O\left(\frac{c(r,i)^4(\log_\xi n)^2}{n^2}\right) = O\left(\xi^{4i}\frac{(\log_\xi n)^2}{n^2}\right),$$

as long as $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$.

Hence, the probability that at least two chords are unessential in any round $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$ is at most

$$\sum_{i=1}^{\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})} p_i = O\left(\sum_{i=1}^{\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})} \xi^{4i}\frac{(\log_\xi n)^2}{n^2}\right)$$

$$= O\left(\xi^{4(\frac{1}{4}\log_\xi \frac{n}{(\log_\xi n)^3})}\frac{(\log_\xi n)^2}{n^2}\right) = O\left(\frac{1}{n\log_\xi n}\right) = o\left(\frac{1}{n}\right). \qquad \square$$

According to the previous lemma, with probability $1-o(1/n)$ the growth of $se(r,i)$ for $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$ is fast and proportional to the powers of ξ. In fact, since in steps 1 and 2 the matchings M_1 and M_2 are activated, it results that $se(r,1) = 1$ and $se(r,2) = 2$. Then, $se(r,i) \geq fib(i-1)+1$ for every $i \geq 2$, where $fib(1) = fib(2) = 1$ and $fib(i)$ is the ith Fibonacci's number (actually $se(r,i)$ is even more, as this estimation does not take into account the fact that the root r also chooses a chord at round 3). This can be shown inductively by observing that $se(r,2) = 2 = fib(1)+1$, $se(r,3) \geq 2 = fib(2)+1$, and assuming that the claim holds until a given $i \geq 3$, $se(r,i+1) \geq se(r,i)+se(r,i-1)-1 \geq fib(i-1)+1+fib(i-2)+1-1 = fib(i)+1$. Thus, $se(r,i) = \Omega(\xi^i)$ and $ce(r,i) = \Omega(\xi^i)$.

Let E1 be the event associated with Lemma 3.1, i.e., that there is at most one unessential chord in every round $i \leq \frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$ or analogously $se(r,i) \geq se(r,i-1)+se(r,i-2)-1$.

As i becomes larger and $se(r,i)$ and $ce(r,i)$ increase, the probability of having at least two unessential chords per round is no longer sufficiently low. However, an optimal growth rate can still be maintained even if we allow a number of unessential chords which is a logarithmic fraction of all the chosen ones. Namely, assume that $un(i) \leq \frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}$ for every i such that $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$. Then $se(r,i) \geq se(r,i-1)+se(r,i-2)-\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n} = (se(r,i-1)+se(r,i-2))(1-\frac{1}{1+\log_\xi n})$, i.e., $se(r,i)(1+\frac{1}{\log_\xi n}) \geq se(r,i-1)+se(r,i-2)$. Thus, for any i between $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$ and $(\log_\xi \frac{n}{(\log_\xi n)^3})$, we have that $se(r,i) = \Omega(\xi_n^i)$, where ξ_n is the largest root of the equation $\gamma^2 \cdot (1+\frac{1}{\log_\xi n}) - \gamma - 1 = 0$. Clearly $\xi_n \leq \xi$ and as a limit for n going to infinity $\xi_n = \xi = (1+\sqrt{5})/2$. The following lemma allows a more exact estimation of ξ_n.

LEMMA 3.2. *For any integer $d \geq 2$ and positive real ϵ suitably small, let $\xi(\epsilon)$ be the largest root of the equation $\gamma^d(1+\epsilon) - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1 = 0$. Then $\xi(\epsilon) = \xi(0) - \Theta(\epsilon)$.*

Proof. $\xi(\epsilon)$ is an implicit function defined by the equation $f(\epsilon, \gamma) = 0$ with $f(\epsilon, \gamma) = \gamma^d(1+\epsilon) - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1$.

$f(\epsilon, \gamma)$ is continuous and has continuous partial derivatives $f_\gamma(\epsilon, \gamma) = d\gamma^{d-1}(1+\epsilon) - (d-1)\gamma^{d-2} - (d-2)\gamma^{d-3} - \cdots - 2\gamma - 1$ with respect to γ and $f_\epsilon(\epsilon, \gamma) = \gamma^d$ with respect to ϵ.

Moreover, $f_\gamma(0, \xi(0)) > 0$ since

$$\begin{aligned} & d\xi(0)^{d-1} - (d-1)\xi(0)^{d-2} - (d-2)\xi(0)^{d-3} - \cdots - 2\xi(0) - 1 \\ &= \left(\sum_{i=2}^{d-1} \xi(0)^{d-i-1} \left(\xi(0)^i - \xi(0)^{i-1} - \xi(0)^{i-2} - \cdots - \xi(0) - 1 \right) \right) \\ &\qquad + \xi(0)^{d-3} \left(2\xi(0)^2 - \xi(0) \right) > 0. \end{aligned}$$

In fact, the largest root of the equation $\gamma^i - \gamma^{i-1} - \gamma^{i-2} - \cdots - \gamma - 1 = 0$ increases as i increases from the golden ratio ($i = 2$) toward 2. Moreover, since the polynomial $\gamma^i - \gamma^{i-1} - \gamma^{i-2} - \cdots - \gamma - 1$ is monotonic increasing for $\gamma > 1$ and $\xi(0)$ is the root of $\gamma^d - \gamma^{d-1} - \cdots - \gamma - 1$, for any integer i such that $2 \leq i < d$ it results that $\xi(0)^i - \xi(0)^{i-1} - \cdots - \xi(0) - 1 > 0$. The last term $\xi(0)^{d-3}\left(2\xi(0)^2 - \xi(0)\right)$ is clearly greater than 0 as $2\xi(0)^2 - \xi(0) > 1$.

By the classical theorem of implicit functions, the derivative $\xi'(\epsilon)$ of $\xi(\epsilon)$ evaluated in the point $\epsilon = 0$ is equal to $\xi'(0) = -\frac{f_\epsilon(0,\xi(0))}{f_\gamma(0,\xi(0))}$ and approximating $\xi(\epsilon)$ by means of the Taylor series it results that

$$\xi(\epsilon) = \xi(0) + \xi'(0)\epsilon + o(\epsilon) = \xi(0) - \Theta(\epsilon). \qquad \square$$

According to Lemma 3.2, for $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$, if $se(r,i)(1 + \frac{1}{\log_\xi n}) \geq se(r,i-1) + se(r,i-2)$, then the growth of $se(r,i)$ is still optimal. In fact $\xi_n^i = (\xi(1 - \Theta(1/\log n)))^i = \Theta(\xi^i)$, since for δ converging to 0 it is $1 - \delta \leq e^{-\delta} \leq 1 - \delta/2$ and thus $(1 - \Theta(1/\log n))^i = e^{-\Theta(i/\log n)} = \Theta(1)$ if $i = O(\log n)$. Hence, $se(r,i) = \Omega(\xi^i)$, $1 \leq i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$.

Let us now show that this holds with high probability.

LEMMA 3.3. *If event* E1 *holds, then with probability* $1 - o(1/n)$ $se(r,i)(1 + \frac{1}{\log_\xi n}) \geq se(r,i-1) + se(r,i-2)$ *for every i such that* $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$.

Proof. As in Lemma 3.1, given a round i with $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$, assuming that E1 holds and that for any i', $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) \leq i' < i$, $se(r,i')(1 + \frac{1}{\log_\xi n}) \geq se(r,i'-1) + se(r,i'-2)$, it is sufficient to show that the probability that there are at least $\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}$ unessential chords is $o(1/n^2)$. In fact, this implies that $se(r,i)(1 + \frac{1}{\log_\xi n}) \geq se(r,i-1) + se(r,i-2)$ with probability $1 - o(1/n^2)$, and thus the probability that this holds for all rounds i between $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3})$ and $(\log_\xi \frac{n}{(\log_\xi n)^3})$ is $(1 - o(1/n^2))^{\log_\xi n} = 1 - o(1/n)$.

By hypothesis, $\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n} = \Omega(\xi^i/\log_\xi n)$. Therefore, the probability that at least $\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}$ chords are unessential at round i is

$$p_i \leq \binom{c(r,i)}{\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}} \left(\frac{6(\log_\xi n)c(r,i)}{n-c(r,i)}\right)^{\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}}$$

$$\leq \left(\frac{e\cdot c(r,i)}{\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}}\right)^{\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}} \left(\frac{6(\log_\xi n)c(r,i)}{n-c(r,i)}\right)^{\frac{se(r,i-1)+se(r,i-2)}{1+\log_\xi n}}$$

$$= \left(\frac{(\log_\xi n)^2\xi^i}{n}\right)^{\Omega(\frac{\xi^i}{\log_\xi n})} = o(1/n^2)$$

as $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$, where we have used the inequality $\binom{a}{b} \leq \left(\frac{e\cdot a}{b}\right)^b$. □

Notice that a logarithmic fraction of unessential chords is the maximum that we can allow in order to maintain the optimal growth behavior. This allows us to reach the highest possible order of round for which the successive phase begins.

Let E2 be the event associated with the previous lemma, that is, that $se(r,i)(1+\frac{1}{\log_\xi n}) \geq se(r,i-1)+se(r,i-2)$ for every i such that $\frac{1}{4}(\log_\xi \frac{n}{(\log_\xi n)^3}) < i \leq (\log_\xi \frac{n}{(\log_\xi n)^3})$.

We are now left with the last phase, i.e., that starting from round $\lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil$ only $O(\log\log n)$ additional rounds are needed with high probability to inform the remaining vertices.

LEMMA 3.4. *If* E1 *and* E2 *hold, then with probability* $1-o(1/n)$ *after step* $\lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil$ *any uninformed vertex in* G *receives the item of* r *in at most* $O(\log\log n)$ *additional rounds.*

Proof. Given any vertex x in G, as for r, with probability $1-o(1/n)$ $se(x, \lceil 5\log_\xi \log n \rceil) = \Omega(\xi^{5\log_\xi \log n}) = \Omega((\log n)^5)$. Routing the information paths in the opposite way, this implies that there are at least $\Omega((\log n)^5)$ different vertices that when informed can communicate the item of r to x in $O(\log\log n)$ rounds.

We now prove a slightly stronger statement, i.e., that this event holds with probability $1-o(1/n^2)$, so that it is verified for all possible vertices x with probability $1-o(1/n)$.

This is accomplished by observing that, starting from x, the probability of having at least three unessential chords in any round $i \leq 5\log_\xi \log n$ is at most

$$\sum_{i=1}^{5\log_\xi \log n} p_i \leq \sum_{i=1}^{5\log_\xi \log n} \binom{c(r,i)}{3} \left(\frac{6(\log_\xi n)c(r,i)}{n-c(r,i)}\right)^3$$

$$= O\left(\sum_{i=1}^{5\log_\xi \log n} \xi^{6i}\frac{(\log_\xi n)^3}{n^3}\right) = o\left(\frac{1}{n^2}\right).$$

Hence, $se(x,i) \geq se(x,i-1) + se(x,i-2) - 2$ holds with probability at least $1 - o(1/n^2)$. Unfortunately, with the initial conditions $se(x,1) = 1$ and $se(x,2) = 2$, such recurrence collapses, as it establishes decreasing lower bounds for $se(r,i)$. However, this problem is avoided by showing that the probability that $se(x,3) \leq 2$ is $o(1/n^2)$. In fact, $se(x,3) \leq 2$ if and only if all the chords chosen in round 3 (corresponding to M_3) are unessential. Each chord can be chosen in at most n different ways, of which at most $O(\log_\xi n)$ corresponding to unessential chords. Therefore, since there are four informed vertices at the beginning of round 3, the probability that $se(x,3) \leq 2$ is proportional to $((\log_\xi n)/n)^4 = o(1/n^2)$. If $se(x,3) \geq 3$, the recurrence produces the right growth behavior proportional to the powers of ξ. In fact, a trivial induction shows that $se(x,i) \geq fib(i-3) + 2$ for $i \geq 4$.

Therefore, $se(x, \lceil 5 \log_\xi \log n \rceil) = \Omega((\log n)^5)$ with probability $1 - o(1/n^2)$. Let us assume that $se(x, \lceil 5 \log_\xi \log n \rceil) = \Omega((\log n)^5)$. Since by hypothesis events E1 and E2 hold, $se(r, \lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil) = \Omega(\xi^{(\log_\xi \frac{n}{(\log_\xi n)^3})}) = \Omega(\frac{n}{(\log_\xi n)^3})$. If $SE(x, \lceil 5 \log_\xi \log n \rceil) \cap SE(r, \lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil) \neq \emptyset$, i.e., x is at distance $O(\log \log n)$ from at least one vertex informed at round $\lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil$, then x can be informed in $O(\log \log n)$ additional rounds (following the information path from x to a vertex in the intersection in the opposite direction); otherwise let us estimate the probability that there is not any chord connecting two vertices in the two sets.

Given any two subsets of vertices S_1 and S_2 of cardinality s_1 and s_2, respectively, if a partial random matching has already been constructed in which chords do not have endpoints in S_1 or S_2 and if $s_1 \leq s_2$, then the probability that in the whole random matching there is not any edge between them is at most

$$\frac{n - s_2}{n} \cdot \frac{n - s_2 - 2}{n - 2} \cdots \frac{n - s_2 - 2(s_1 - 1)}{n - 2(s_1 - 1)}$$

$$\leq \left(1 - \frac{s_2}{n}\right)^{s_1} \leq e^{-\frac{s_2}{n} s_1},$$

where we have used the inequality $(1 - \delta) \leq e^{-\delta}$ for any $0 \leq \delta \leq 1$.

In our case, assuming that rounds $\lceil 5 \log_\xi \log n \rceil$ and $\lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil$ do not correspond to the activation of M_3 (otherwise the argument works in the same way considering the successive rounds), $SE(x, \lceil 5 \log_\xi \log n \rceil)$ and $SE(r, \lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil)$ satisfy the conditions of S_1 and S_2 and the probability that there is not any chord between the two sets is at most

$$e^{-\frac{se(r, \lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil)}{n} se(x, \lceil 5 \log_\xi \log n \rceil)} = e^{-\Omega\left(\frac{\frac{n}{(\log_\xi n)^3}}{n} (\log n)^5\right)} = e^{-\Omega((\log n)^2)} = o(1/n^2).$$

Hence, by the above arguments the probability that there is not an informing path from $SE(r, \lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil)$ to x taking $O(\log \log n)$ rounds is at most $o(1/n^2)$.

Summing up over all possible uninformed vertices, the probability that there exists x that cannot be informed in additional $O(\log \log n)$ rounds is $o(1/n)$; hence the lemma. □

Let E3 be the event corresponding to the previous lemma, i.e., that after step $\lceil \log_\xi \frac{n}{(\log_\xi n)^3} \rceil$ any uninformed vertex in G receives the item of r in at most $O(\log \log n)$ additional rounds.

THEOREM 3.5. *With probability* $1-o(1)$, $b(G) \le (\log_\xi n) + O(\log\log n)$.

Proof. The theorem follows simply by observing that for a fixed root r events E1, E2, and E3 hold with probability $1-o(1/n)$, i.e., they hold with probability $1-o(1)$ for every possible r. Therefore, $b(G) \le (\log_\xi \frac{n}{(\log_\xi n)^3}) + O(\log\log n) = (\log_\xi n) + O(\log\log n)$ with probability $1-o(1)$. □

Let us now extend our upper bound to any fixed parameter $d \ge 2$, i.e., to graphs with vertices of degree at most $d+1$.

In this case the lower bound in [22, 3] for graphs G with parameter d generalizes simply by observing that $s(r,i)$ obeys the recurrence $s(r,i) \le s(r,i-1) + s(r,i-2) + \cdots + s(r,i-d)$, which gives $s(r,i) = O(\xi^i)$ and $c(r,i) = O(\xi^i)$, where ξ is the largest root of the equation $\gamma^d - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1 = 0$. Therefore, $b(G) \ge (\log_\xi n) - O(1)$.

Our construction is again based on a cycle of n vertices with n even, where we can still identify the two matchings M_1 and M_2. Then we add $d-1$ random matchings $M_3, \ldots, M_{d+1}$ and the protocol activates $M_1, \ldots, M_{d+1}$ cyclically in the order.

The definition of unessential chords is exactly the same, and we basically show that $se(r,i)$ grows quickly.

Lemmas 3.1, 3.3, and Lemma 3.4 apply without any modification, even if in the proof of Lemma 3.4 the probability that there is not any edge between the two sets S_1 and S_2 is even lower, as there is more than one random matching to be considered. However, considering only one does not affect the correctness of the proof.

Hence, if G is the graph with parameter d obtained from the cycle by adding the $d-1$ random matchings, the following theorem holds.

THEOREM 3.6. *With probability* $1-o(1)$, $b(G) \le (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^d - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1 = 0$.

3.2. Directed graphs. In this section we prove an analogous broadcasting lower bound for directed graphs of fixed parameter $d \ge 2$, i.e., with vertices of out-degree at most d. We don't give a detailed proof, but we just point out the few basic differences from the undirected case.

Clearly, again by [22, 3] it is $s(r,i) = O(\xi^i)$ and $c(r,i) = O(\xi^i)$, where ξ is the largest root of the equation $\gamma^d - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1 = 0$.

The construction of the random graph is slightly different from the undirected case. In fact, we have a clockwise directed cycle of n vertices with n even, where we can still identify the two matchings M_1 and M_2, both directed clockwise.

Let us denote as V_1 (resp., V_2) the subset of the $n/2$ vertices whose outgoing arcs belong to M_1 (resp., M_2). Following the cycle, vertices belong alternatively to V_1 and to V_2 and every arc of M_1 (resp., M_2) goes from V_1 to V_2 (resp., from V_2 to V_1). We add another $2d-2$ random directed matchings $M_3, \ldots, M_{2d}$ such that M_i, $3 \le i \le 2d$, goes from V_1 to V_2 if i is odd, otherwise from V_2 to V_1. Hence, every vertex has in-degree and out-degree equal to d.

The protocol is slightly different: as soon as a new vertex v is informed, it tries to inform its d neighbors reachable through its outgoing arcs in the next d rounds using in the order the d outgoing matchings M_1, M_3, ..., M_{2d-1} if $v \in V_1$, and M_2, M_4, ..., M_{2d} if $v \in V_2$. The main difference from the undirected case is that here each round does not correspond to the activation of a single matching M_i, but arcs belonging to different matchings can be activated. Moreover, the protocol is different for every fixed root vertex r.

The definition of unessential chords is exactly the same and again Lemmas 3.1, 3.3, and 3.4 apply as in the undirected case. The only difference is in the initial conditions of the recurrences, that must be properly set in order to avoid collapsing.

To this aim, observe that $se(r,i)$, $i \geq 1$, is at least equal to 1 plus the total number of essential chords generated until round i included (or analogously $se(r,i+1)$ is equal to $se(r,i)$ plus the number of essential chords in round $i+1$). In fact, from the root or from any new vertex informed along an essential chord, there is a thread of vertices following it along the directed cycle that are informed one by one in the successive rounds. Hence, the probability that $se(r,5) < 3$ corresponds to the probability of at most 1 essential chord until round 5, that is, the probability of at least $m-1$ unessential chords among the m ones drawn until round 5. Therefore, since $m \geq 4$ (at least one chord drawn for the root and one for each vertex at distance at most 3 from the root along the directed cycle), the probability of at most 1 essential chord until round 5 is $O(((\log_\xi n)/n)^3) = o(1/n^2)$

This means that both the recurrences $se(r,i) \geq se(r,i-1) + se(r,i-2) + \cdots + se(r,i-d) - 1$ generalizing the one of Lemma 3.1 and $se(r,i) \geq se(r,i-1) + se(r,i-2) + \cdots + se(r,i-d) - 2$ generalizing the one inside the proof of Lemma 3.4 do not collapse with probability $1 - o(1/n^2)$ and thus they grow with the powers of ξ; in fact, a simple induction shows that there is a suitable constant c such that $se(r,i) \geq fib(i-c,d) = \Omega(\xi^i)$, where $fib(i,d) = fib(i-1,d) + fib(i-2,d) + \cdots fib(i-d,d)$ with initial conditions $fib(1,d) = fib(2,d) = \cdots = fib(d,d) = 1$.

Hence, if G is the graph with parameter d obtained as above, the following theorem holds.

THEOREM 3.7. *With probability* $1 - o(1)$, $b(G) \leq (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^d - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1 = 0$.

4. Upper bounds for gossiping. We now provide upper bounds on the gossiping time in graphs with fixed parameter and for systolic protocols. In order to avoid unnecessarily long proofs, as for broadcasting in directed graphs, we just point out the basic differences from the previous constructions. The remaining details are completely analogous.

In the full-duplex case, we observe that in the construction of the undirected broadcasting upper bound the protocol does not depend on the particular root vertex chosen, and thus it is also a gossiping protocol. Hence, as a direct consequence of Theorem 3.6, we get the following theorem.

THEOREM 4.1. *With probability* $1 - o(1)$, $g(G) \leq (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^d - \gamma^{d-1} - \gamma^{d-2} - \cdots - \gamma - 1 = 0$.

Clearly this result is optimal, as it matches the lower bound of [22, 3]. Since by construction the protocol is s-systolic with $s = d+1$ and $g_s(G) \geq g(G)$, a matching upper bound is determined also for systolic full-duplex gossiping.

THEOREM 4.2. *With probability* $1 - o(1)$, $g_s(G) \leq (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^{s-1} - \gamma^{s-2} - \gamma^{s-3} - \cdots - \gamma - 1 = 0$.

For gossiping in directed graphs we use the same corresponding construction of broadcasting with the only difference that the protocol is now modified to be the same for every root vertex. Namely, it consists of activating cyclically in a periodic fashion the $2d$ matchings $M_1, \ldots, M_{2d}$. The definition of essential chords is exactly the same.

All the considerations for directed broadcasting apply with the difference that, since if at a certain round a matching from V_1 to V_2 is activated then the next matching is from V_2 to V_1 and vice versa, for any chosen vertex $r \in V_1$ now $se(r,i)$ follows the recurrence $se(r,i) \geq se(r,i-1) + se(r,i-3) + \cdots se(r,i-2d+1) - un(i)$. In fact, the vertices informed at times $i-2$, $i-4, \ldots, i-2d$ cannot inform any new vertex during round i because incoming matchings are activated, and thus $se(r,i)$ is at least equal to the sum of the number of new informed vertices during rounds $i-1$, $i-3, \ldots, i-2d+1$,

minus the unessential chords during round i.

Concerning the setting of the initial conditions that avoid collapsing, given any $j \geq 2$, consider the new informed vertices during rounds $2jd+1$ and $2jd+2$, i.e., the first two rounds of the $(j+1)$st period corresponding to the activations of M_1 and M_2, respectively. Generalizing the corresponding argument for directed broadcasting, $se(r, 2jd+1)$ and $se(r, 2jd+2)$ are at least equal to 1 plus the total number of essential chords generated until round $2(j-1)d$, i.e., the last round of the $(j-1)$st period activating M_{2d}. In fact, from the root or from any new vertex informed along an essential chord, there is a thread of vertices following it along the directed cycle that are informed one by one during the rounds in which M_1 and M_2 are activated. In particular, if the unessential chord is from V_1 to V_2, such a thread starts with a vertex in V_2 and during the next period is extended only by the activation of M_2 (thus contributing 1 to $se(r, 2(j-1)d+2)$ and not to $se(r, 2(j-1)d+1)$). However, starting from the $(j+1)$st period, the thread is extended both by the activation of M_1 and of M_2, thus increasing of 1 both $se(r, 2jd+1)$ and $se(r, 2jd+2)$. Similarly, if the unessential chord is from V_2 to V_1, the thread is extended both by M_1 and M_2 already in the jth period, and in every case it contributes 1 both to $se(r, 2jd+1)$ and $se(r, 2jd+2)$. Hence, taking $j = 4$, the probability that $se(r, 8d+1) < 4$ or $se(r, 8d+2) < 4$ is smaller than the probability of at most 2 essential chords until round $6d$, that is, the probability of at least $m-2$ unessential chords among the m ones drawn until round $6d$. Therefore, since $m \geq 6$ (at least 1 chord drawn for the root and for the 5 vertices at distance at most 5 from the root along the directed cycle, i.e., informed during the first 3 periods), the probability of at most 2 essential chords until round $6d$ is $O(((\log_\xi n)/n)^4) = o(1/n^2)$.

With the initial conditions $se(r, 8d+1) \geq 4$ and $se(r, 8d+2) \geq 4$, the recurrences $se(r,i) \geq se(r,i-1) + se(r,i-3) + \cdots + se(r,i-2d+1) - 1$ corresponding to the one of Lemma 3.1 and $se(r,i) \geq se(r,i-1) + se(r,i-3) + \cdots se(r,i-2d+1) - 2$ corresponding the one inside the proof of Lemma 3.4 do not collapse. Thus, with probability $1 - o(1/n^2)$ they grow asymptotically as $se(r,i) \geq se(r,i-2) + se(r,i-4) + \cdots + se(r,i-2d)$.

Finally, concerning the recurrence $se(r,i)(1 + \frac{1}{\log_\xi n}) \geq se(r,i-1) + se(r,i-3) + \cdots se(r,i-2d+1) - 2$ corresponding to Lemma 3.3, a simple modification of Lemma 3.2 shows that for any integer $d \geq 2$ and positive real ϵ suitably small, if $\xi(\epsilon)$ is the largest root of the equation $\gamma^{2d-1}(1+\epsilon) - \gamma^{2d-2} - \gamma^{2d-4} - \cdots - \gamma^2 - 1 = 0$, then $\xi(\epsilon) = \xi(0) - \Theta(\epsilon)$.

All these facts together imply that, with probability $1 - o(1/n)$, $se(r,i) = \Omega(\xi^i)$ for every $i \leq \log_\xi \frac{n}{(\log_\xi n)^3}$ and after round $\log_\xi \frac{n}{(\log_\xi n)^3}$ only $O(\log\log n)$ rounds are sufficient to deliver the item of r with ξ the largest root of $\gamma^{2d-1} - \gamma^{2d-2} - \gamma^{2d-4} - \cdots - \gamma^2 - 1 = 0$.

Hence, since the protocol does not depend on the particular chosen root r, the following theorem holds.

THEOREM 4.3. *With probability* $1 - o(1)$, $g(G) \leq (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^{2d-1} - \gamma^{2d-2} - \gamma^{2d-4} - \cdots - \gamma^2 - 1 = 0$.

Since $1 < \xi < 2$, plugging $\lambda = 1/\xi$ in the above theorem, $g(G) \leq (\log n)/(\log 1/\lambda) + O(\log\log n)$ with $0 < \lambda < 1$ and $(1/\lambda)^{2d-1} - (1/\lambda)^{2d-2} - (1/\lambda)^{2d-4} - \cdots - 1/\lambda^2 - 1 = 0$, i.e., $\lambda + \lambda^3 + \cdots \lambda^{2d-1} = \sqrt{p_d(\lambda)} \cdot \sqrt{p_d(\lambda)} = 1$, where $p_j(\lambda) = \lambda + \lambda^3 + \cdots + \lambda^{2j-1}$ for any integer $j > 0$. This is slightly different from the lower bound proved in [9], where λ satisfies $\sqrt{p_\infty(\lambda)} \cdot \sqrt{p_d(\lambda)} = 1$, with $p_\infty(\lambda) = \lim_{j\to\infty} p_j(\lambda) = \lambda/(1-\lambda^2)$.

Similarly as for broadcasting, the above protocol is also s-systolic with $s = 2d$. Hence the following theorem holds.

THEOREM 4.4. *With probability* $1 - o(1)$, $g_s(G) \leq (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^{s-1} - \gamma^{s-2} - \gamma^{s-4} - \cdots - \gamma^2 - 1 = 0$.

Again, plugging $\lambda = 1/\xi$ in the above theorem, $g_s(G) \leq (\log n)/(\log 1/\lambda) + O(\log\log n)$ with $0 < \lambda < 1$ and $\lambda + \lambda^3 + \cdots \lambda^{s-1} = \sqrt{p_{\lfloor s/2 \rfloor}(\lambda)} \cdot \sqrt{p_{\lceil s/2 \rceil}(\lambda)} = 1$. Moreover, ξ is the unique root greater than 1 of $\gamma^{s-1} - \gamma^{s-2} - \gamma^{s-4} - \cdots - \gamma^2 - 1 = 0$ and λ is the unique real between 0 and 1 that satisfies $\lambda + \lambda^3 + \cdots \lambda^{s-1} = 1$. Hence, for systolic protocols our result is tight, as it matches the lower bound of [8].

Let us conclude the section considering undirected (or directed symmetric) graphs. The construction for half-duplex gossiping is essentially the same, with the difference that, since the digraph G is undirected, there are two undirected matchings M_1 and M_2 between V_1 and V_2 along the cycle, plus $d + 1$ undirected random matchings $M_3, \ldots, M_{d+1}$, still between V_1 and V_2. Each undirected matching M_i, $1 \leq i \leq d+1$, can be seen as two opposite directed matchings $\overrightarrow{M}_i$ and $\overleftarrow{M}_i$, respectively, from V_1 to V_2 and from V_2 to V_1.

If d is even, the protocol consists of activating cyclically in the order the $2d + 2$ directed matchings $\overrightarrow{M}_1, \overleftarrow{M}_2, \ldots, \overleftarrow{M}_d, \overrightarrow{M}_{d+1}, \overleftarrow{M}_1, \overrightarrow{M}_2, \ldots, \overrightarrow{M}_d, \overleftarrow{M}_{d+1}$, while if d is odd the $2d + 2$ directed matchings $\overrightarrow{M}_1, \overleftarrow{M}_2, \ldots, \overleftarrow{M}_d, \overrightarrow{M}_{d+1}, \overleftarrow{M}_2, \overrightarrow{M}_1, \overleftarrow{M}_4, \overrightarrow{M}_3, \ldots, \overrightarrow{M}_{d+1}, \overleftarrow{M}_d$.

Although the proof from now on is equivalent to the directed gossiping case, the recurrence is slightly different. In fact, if d is even, $se(r,i) \geq se(r,i-1) + se(r,i-3) + \cdots + se(r,i-d+1) + se(r,i-d-3) + \cdots + se(r,i-2d-1) - un(i)$, since besides the vertices informed during rounds $i-2$, $i-4, \ldots, i-2d-2$, also the vertices informed during round $i-d-1$ cannot inform any new vertex during step i, as they are using the reverse of the arcs activated at time $i-d-1$ (through which they have been informed).

Similarly, it is easy to check that if d is odd, either the vertices informed during round $i-d-2$ or during round $i-d-4$ cannot inform any new vertex during step i. Since the first case is worse, in every case $se(r,i) \geq se(r,i-1) + se(r,i-3) + \cdots + se(r,i-d+2) + se(r,i-d-2) + \cdots + se(r,i-2d-1) - un(i)$.

Therefore, the following theorem holds.

THEOREM 4.5. *With probability* $1 - o(1)$, $g(G) \leq (\log_\xi n) + O(\log\log n)$, *where* ξ *is the largest root of the equation* $\gamma^{2d+1} - \gamma^{2d} - \gamma^{2d-2} - \cdots - \gamma^d - \gamma^{d-4} - \cdots - \gamma^2 - 1 = 0$ *if* d *is even, and* $\gamma^{2d+1} - \gamma^{2d} - \gamma^{2d-2} - \cdots - \gamma^{d+1} - \gamma^{d-3} - \cdots - \gamma^2 - 1 = 0$ *if* d *is odd.*

The above half-duplex protocol is $(2d + 2)$-systolic and does not give a matching upper bound for systolic half-duplex protocols on general topologies. This holds because the protocol has at the same time a restriction due to the fact that the graph has fixed parameter and one deriving from the fact that it is systolic. In fact, if every incident arc is activated in some round, for every incoming arc activated there is a successive round in which the reverse of the same arc is activated, and in such a round the information communicated by the incoming arc cannot be forwarded to any new vertex. However, if we don't care about the vertex degree and consider the symmetric digraph obtained by the construction of gossiping in directed graphs by adding for each directed matching a reverse matching (never activated), we get a half-duplex s-systolic protocol on an undirected graph with $s = 2d$ and with the same optimal gossiping time of Theorem 4.4. Hence, even if we don't put a separate claim, again

we have a matching upper bound for systolic half-duplex protocols.

5. Concluding remarks. In this paper we have proved the optimality of the general lower bounds known for bounded-degree broadcasting, bounded-degree gossiping (only full-duplex) and systolic gossiping (full-duplex, directed, and half-duplex). This has been accomplished by showing the existence of networks with a matching broadcasting or gossiping time.

Similar results have been proved also for bounded-degree gossiping in the directed and half-duplex modes, but they don't match the corresponding lower bounds of [9]. However, we strongly conjecture that even in this case our bounds are optimal and there do not exist networks with a lower gossiping time, so that the gap can be eliminated only by improving [9].

Finally, our proofs are only existential and the effective construction of graphs attaining such lower bounds is another research direction worthy of consideration.

REFERENCES

[1] A. Bar-Noy and S. Kipnis, *Designing broadcasting algorithms in the postal model for message passing systems*, Math. Systems Theory, 27 (1994), pp. 431–452.

[2] J.-C. Bermond, L. Gargano, A. A. Rescigno, and U. Vaccaro, *Fast gossiping by short messages*, SIAM J. Comput., 27 (1998), pp. 917–941.

[3] J.-C. Bermond, P. Hell, A. Liestman, and J. G. Peters, *Broadcasting in bounded degree graphs*, SIAM J. Discrete Math., 5 (1992), pp. 10–24.

[4] J. Bermond, X. Munoz, and A. Marchetti-Spaccamela, *Induced broadcasting algorithms in iterated line digraphs*, in Proceedings of the 2nd International Euro-Par Conference, Lecture Notes in Comput. Sci. 1123, Springer-Verlag, Berlin, 1996, pp. 313–324.

[5] B. Bollobás and F. R. K. Chung, *The diameter of a cycle plus a random matching*, SIAM J. Discrete Math., 1 (1988), pp. 328–333.

[6] J. de Rumeur, *Communication dans les réseaux de processeurs*, Collection Etudes et Recherches en Informatique, Masson, Paris, 1994.

[7] S. Even and B. Monien, *On the number of rounds necessary to disseminate information*, in Proceedings of the 1st ACM Symposium on Parallel Algorithms and Architectures (SPAA), Santa Fe, NM, 1989, pp. 318–327.

[8] M. Flammini and S. Pérennès, *Lower bounds on systolic gossip*, Inform. and Comput., to appear.

[9] M. Flammini and S. Pérennès, *Lower Bounds on the Broadcasting and Gossiping Time of Restricted Protocols*, Tech. Report 3612, INRIA Sophia-Antipolis, France, 1999.

[10] P. Fraigniaud and E. Lazard, *Methods and problems of communication in usual networks*, Discrete Appl. Math., 53 (1994), pp. 79–133.

[11] S. T. Hedetniemi, S. M. Hedetniemi, and A. Liestman, *A survey of gossiping and broadcasting in communication networks*, Networks, 18 (1986), pp. 319–349.

[12] J. Hromkovič, R. Klasing, E. Stohr, and H. Wagener, *Gossiping in vertex-disjoint paths mode in d-dimensional grids and planar graphs*, Inform. Comput., 123 (1995), pp. 17–28.

[13] J. Hromkovič, R. Klasing, W. Unger, and H. Wagener, *Optimal algorithms for broadcast and gossip in the edge-disjoint path modes*, Inform. Comput., 133 (1997), pp. 1–33.

[14] J. Hromkovič, R. Klasing, B. Monien, and R. Peine, *Dissemination of information in interconnection networks (broadcasting and gossiping)*, in Combinatorial Network Theory, Ding-Zhu Du and D. Frank Hsu, eds., Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995, pp. 125–212.

[15] J. Hromkovič, R. Klasing, D. Pardubská, W. Unger, and H. Wagener, *The complexity of systolic dissemination of information in interconnection networks*, RAIRO Inform. Théor. Appl., 28 (1994), pp. 303–342.

[16] M. Jerrum and S. Skyum, *Families of fixed degree graphs for processor interconnection*, IEEE Trans. Comput., 33 (1984), pp. 190–194.

[17] R. Klasing, B. Monien, R. Peine, and E. Stohr, *Broadcasting in butterfly and de Bruijn networks*, Discrete Appl. Math., 53 (1994), pp. 183–197.

[18] G. Kortsarz and D. Peleg, *Traffic-light scheduling on the grid*, Discrete Appl. Math., 53 (1994), pp. 211–234.

[19] D. W. Krumme, G. Cybenko, and K. N. Venkataraman, *Gossiping in minimal time*, SIAM J. Comput., 21 (1992), pp. 111–139.
[20] R. Labahn, S. Hedetniemi, and R. Laskar, *Periodic gossiping on trees*, Discrete Appl. Math., 53 (1994), pp. 235–246.
[21] R. Labahn and I. Warnke, *Quick gossiping by multi-telegraphs*, in Topics in Combinatorics and Graph Theory, Physica, Heidelberg, Germany, 1990, pp. 451–458.
[22] A. L. Liestman and J. G. Peters, *Broadcast networks of bounded degree*, SIAM J. Discrete Math., 1 (1988), pp. 531–540.
[23] A. Liestman and D. Richards, *Network communication in edge-colored graphs: Gossiping*, IEEE Trans. Par. Distr. Syst., 4 (1993), pp. 438–445.
[24] A. Liestman and D. Richards, *Perpetual gossiping*, Parallel Process. Lett., 3 (1993), pp. 347–355.
[25] S. Pérennès, *Communications dans les réseaux d'interconnexion*, Ph.D. thesis, Université de Nice-Sophia Antipolis, Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis CNRS URA 1376, 1996.
[26] S. Pérennès, *Lower bounds on broadcasting time of de Bruijn networks*, in Proceedings of the 2nd International Euro-Par Conference, Lecture Notes in Comput. Sci. 1123, Springer-Verlag, Berlin, 1996, pp. 325–332.
[27] V. Sunderam and P. Winkler, *Fast information sharing in a complete network*, Discrete Appl. Math., 42 (1993), pp. 75–86.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 283–285

CONSTRUCTION OF FERRERO PAIRS OF ALL POSSIBLE ORDERS*

TIM BOYKETT†

Abstract. The structure of Ferrero pairs in terms of the sizes of the groups involved is investigated and an explicit condition is obtained. For every pair of numbers satisfying this criterion, a Ferrero pair is constructed explicitly. Thus the determination of interesting Ferrero pairs cannot be determined from their direct numerical properties.

Key words. Ferrero pairs, construction, nearrings

AMS subject classifications. 16Y30, 20B25, 20D45

PII. S0895480198338736

1. Introduction.

The use of planar nearrings to construct balanced incomplete block designs (BIBDs) dates back to Ferrero's initial paper [2]. One can say that the complete structure of the planar nearring and the BIBD is held within the structure of the Ferrero pair.

We look at the class of Ferrero pairs that can be constructed, characterizing them by a simple parameter pair, and we see that all possible parameter pairs can be constructed explicitly using the field generated Ferrero pairs as building blocks.

A *Ferrero pair* (N, Φ) is an additively written (but not necessarily abelian) group N and a group Φ of fixed point free (also known as regular or semiregular) automorphisms of N. We consider only finite N. The *parameters* of a Ferrero pair are $n = |N|$ and $t = |\Phi|$. Ferrero pairs are closely related to planar nearrings [1].

Ferrero pairs can be constructed from finite fields in a rather elementary manner. This construction dates back to Ferrero's original construction. Given a finite field K and some multiplicative subgroup $\Phi \leq K^*$, we see that Φ acts upon the additive group K as a group of fixed point free automorphisms. Thus we obtain a Ferrero pair (K, Φ). Such Ferrero pairs are known as *field generated Ferrero pairs.*

The main result of this paper is the following.

THEOREM 1. *Let n, t be positive integers, $n = \Pi_i p_i^{e_i}$ the prime factorization of n; then there exists a Ferrero pair (N, Φ) with $n = |N|$ and $t = |\Phi|$ iff t divides $p_i^{e_i} - 1$ for all i.*

We will go about obtaining this result by looking at what numbers n, t can exist, then explicitly constructing examples where these numbers are realized.

2. Obtaining Ferrero pairs.

It would be of interest to be able to find suitable Φ for a given additive group N such that (N, Φ) is a Ferrero pair. In particular, we would like to know, for a given $n = |N|$, which values of $t = |\Phi|$ are theoretically possible and which can be explicitly constructed.

In this section we will see that a Ferrero pair can be decomposed into a collection of Ferrero pairs with n a prime power. We obtain some simple restrictions upon the

*Received by the editors May 13, 1998; accepted for publication (in revised form) March 6, 2001; published electronically May 22, 2001. This research was funded by FWF Project P11486-TEC, the Johannes–Kepler Hochschulefonds, and the scientific research funding of the government of Upper Austria.

http://www.siam.org/journals/sidma/14-3/33873.html

†Time's Up, Industriezeile 33b, A-4020 Linz, Austria; also, Mathematik, Uni Linz, A-4040 Linz, Austria (tim@timesup.org).

values of t that are feasible. Then we will see an explicit construction for all pairs (n, t) that satisfy these constraints.

PROPOSITION 2 (see 1.3 in [3]). *If (N, Φ) is a Ferrero pair with $N = N_1 \oplus N_2$ and Φ fixing the subgroups N_1 and N_2 as sets, then there are $\Phi_1 \cong \Phi_2 \cong \Phi$ such that (N_1, Φ_1) and (N_2, Φ_2) are Ferrero pairs.*

Proof. Let $\Phi_i := \Phi|_{N_i}$. For $\phi \in \Phi$, write $\phi_i := \phi|_{N_i}$. Suppose Φ_1 was not isomorphic to Φ. Then since Φ_1 is a homomorphic image of Φ, there is some nonidentity $\phi \in \Phi$ such that ϕ_1 is the identity on N_1. Then $\phi(n_1, 0) = (\phi_1 n_1, \phi_2 0) = (n_1, 0)$ so $(n_1, 0)$ is a fixed point of ϕ. However, ϕ is fixed point free, so $\Phi_i \cong \Phi$ for all i.

Suppose there is some $\phi_1 \in \Phi_1$ with a fixed point $n_1 \in N_1$, i.e., $\phi_1 n_1 = n_1$. Take ϕ the isomorph to ϕ_1 in Φ, ϕ_2 the isomorph to ϕ in Φ_2. Then $\phi(n_1, 0) = (\phi_1 n_1, \phi_2 0) = (n_1, 0)$, so $(n_1, 0)$ is a fixed point of ϕ. However, (N, Φ) is a Ferrero pair, so this is not possible; thus there are no fixed points for any $\phi_1 \in \Phi_1$. The proof applies similarly for Φ_2 and we see that (N_i, Φ_i) is a Ferrero pair for each i. □

The restriction to two summands in the above is of course unnecessary; the result can easily be extended to many summands. The distinction of the Φ_i is also unnecessary since they are all isomorphic.

COROLLARY 3. *If (N, Φ) is a Ferrero pair and $N = \bigoplus_i N_i$ with each N_i Φ-invariant, then (N_i, Φ) is a Ferrero pair for each summand N_i.*

This becomes most interesting when we consider a Ferrero pair (N, Φ) and notice that because N is nilpotent by Thompson's theorem (see, e.g., [4, p. 306]), then it can be expressed as a direct product of its Sylow p-subgroups. Since an automorphism cannot take an element outside of its p-group, each Sylow subgroup is closed with respect to Φ. Thus Corollary 3 applies, and we can split the Ferrero pair up into a collection of noninteracting pairs.

COROLLARY 4. *A Ferrero pair is a direct sum of prime power Ferrero pairs with isomorphic automorphism groups.*

What is even more interesting is that the reverse construction also works.

PROPOSITION 5 (see Theorem 5.42 in [1]). *Given a family (N_i, Φ_i) of Ferrero pairs with $\Phi_i \cong \Phi_j$ for all $i, j = 1, \ldots, k$, then (N, Φ) is also a Ferrero pair, where $N := \bigoplus_i N_i$ and $\Phi \cong \Phi_i$.*

The following two results form the proof of Theorem 1.

COROLLARY 6. *Given a Ferrero pair (N, Φ), with $n = |N| = \prod_i p_i^{e_i}$ as the prime factorization, then $t = |\Phi|$ divides $p_i^{e_i} - 1$ for each i.*

Proof. Take one of the direct summands N_i of N as a sum of its Sylow subgroups and note that every orbit of Φ upon N_i except the 0-orbit has order t. No orbit can be larger, and if some orbit is smaller, then some $\phi \in \Phi$ has a fixed point in the orbit. Thus t divides $|N_i| - 1 = p_i^{e_i} - 1$ for each i and we are done. □

PROPOSITION 7. *Given a number $n = \prod_i p_i^{e_i}$ and some number t such that t divides $p_i^{e_i} - 1$ for all i, there is a Ferrero pair (N, Φ) with $|N| = n$ and $|\Phi| = t$.*

Proof. Let $K_i := GF(p_i^{e_i})$ and $N_i := (K_i, +)$. Let ω_i be a generator of the multiplicative group K_i^*. Since t divides $p_i^{e_i} - 1$, let m_i be the number such that $m_i t = p_i^{e_i} - 1$. Then let $\Phi_i :=< \omega_i^{m_i} >$, a cyclic subgroup of the multiplicative group K_i^* of order t. Then (N_i, Φ_i) is a field generated Ferrero pair.

Take $N := \bigoplus_i N_i$, Φ the cyclic group of order t and by Proposition 5, (N, Φ) is a Ferrero pair with parameters (n, t). □

Thus we obtain Theorem 1.

3. Conclusion. In this note we have seen that there is an explicit construction method for Ferrero pairs of all possible orders. Thus the determination of the

"interesting" Ferrero pairs cannot be based upon their numerical properties.

It is unknown whether the construction presented here of glueing Ferrero pairs together via their automorphism groups has a parallel in design theory.

Acknowledgments. I am grateful for resources offered by the Mathematics Department at the University of Western Australia and funding from the Johannes–Kepler University Hochschulefonds financing, as well as support from the scientific research funding of the government of Upper Austria.

REFERENCES

[1] J. R. Clay, *Nearrings: Geneses and Applications*, Oxford University Press, Oxford, UK, 1992.
[2] G. Ferrero, *Stems planari e BIB-desegni*, Riv. Math. Univ. Parma (2), 11 (1970), pp. 79–96.
[3] W.-F. Ke and H. Keichle, *Automorphisms of certain design groups*, J. Algebra, 167 (1994), pp. 488–500.
[4] D. J. S. Robinson, *A Course in the Theory of Groups*, Springer, New York, 1996.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 286–290

MONOCHROMATIC PARTITIONS OF COMPLETE UNIFORM HYPERGRAPHS*

KRZYSZTOF BRYŚ† AND ZBIGNIEW LONC†

Abstract. Let H be a complete n-uniform hypergraph, the edges of which are colored with c colors. A subhypergraph of H is called *monochromatic* if all its edges are colored with the same color. We prove that, for fixed n, c, and k, the problem of deciding whether H admits a partition of its vertex set into subsets inducing complete monochromatic subhypergraphs of order at least k is polynomial.

Key words. complete uniform hypergraph, vertex partition, computational complexity, Ramsey theorem

AMS subject classifications. 05C65, 05D10

PII. S0895480199357716

1. Introduction.

Let n, c, and k be positive integers. In this paper we deal with partitions of the vertex set of a complete n-uniform hypergraph whose edges are colored with c colors into subsets inducing complete monochromatic subhypergraphs of order at least k. (A subhypergraph is *monochromatic* if all its edges have the same color.)

Denote by $\mathcal{P}_n(X)$ the set of n-element subsets of a finite set X. Let $H = (X, \mathcal{E})$, $\mathcal{E} \subseteq \mathcal{P}_n(X)$, be an n-uniform hypergraph and let $Y \subseteq X$. By a hypergraph *induced* in H by the set Y we mean a hypergraph $(Y, \{E \in \mathcal{E} : E \subseteq Y\})$. Define $K_n^m = (X, \mathcal{P}_n(X))$, $|X| = m$, to be a complete n-uniform hypergraph.

Let us consider the following problem.

Problem $\mathcal{P}_{n,c,k}$.

Instance. A complete n-uniform hypergraph H, the edges of which are colored with c colors.

Question. Can the vertex set of H be partitioned into subsets inducing monochromatic complete subhypergraphs of order at least k?

The following theorem is the main result of this paper.

THEOREM 1. *Let c, k, and n be any fixed positive integers. The problem $\mathcal{P}_{n,c,k}$ is polynomial time solvable.*

The above problem has already been considered mainly in the case of $n = c = 2$, i.e., for graphs whose edges are colored with two colors. Notice that in this case our problem can be reformulated as follows.

Problem $\mathcal{P}_{2,2,k}$.

Instance. A graph G.

Question. Can the vertex set of G be partitioned into subsets inducing complete subgraphs and/or their complements of order at least k?

A variation (say, $\mathcal{P}'_{2,2,k}$) of this problem, which can be obtained from $\mathcal{P}_{2,2,k}$ by substituting the words "at least k" by the words "exactly k," has been considered earlier.

*Received by the editors June 1, 1999; accepted for publication (in revised form) January 29, 2001; published electronically May 22, 2001.
http://www.siam.org/journals/sidma/14-3/35771.html

†Institute of Mathematics, Warsaw University of Technology, Pl. Politechniki 1, 00-661 Warsaw, Poland (brys@alpha.mini.pw.edu.pl, zblonc@alpha.mini.pw.edu.pl).

Favaron, Lonc, and Truszczyński [4] proved a result which immediately implies polynomiality of the problem $\mathcal{P}'_{2,2,3}$ if the instance graph G is a line graph. Lonc [7] showed that $\mathcal{P}'_{2,2,k}$ is polynomial time solvable for any fixed k if the instance graph G is a line graph. Finally, Bryś and Lonc [1] gave a proof that $\mathcal{P}'_{2,2,k}$ is polynomial time solvable for any k and an arbitrary instance graph G. Clearly Theorem 1 contains all of these results as special cases.

It has to be mentioned here that for a fixed $k \geq 3$, the problem of existence of a partition of the vertex set of an instance graph into k-element sets inducing complete subgraphs is NP-complete (see Garey and Johnson [3]). Hell and Kirkpatrick [6] proved that the problem of existence of a partition of a vertex set into subsets inducing complete subgraphs of order at least k, for a fixed $k \geq 3$, is NP-complete. Note that for $k = 2$ this problem is equivalent to the problem of existence of a partition of the vertex set of a graph into subsets inducing complete subgraphs of order 2 and/or 3. Hell and Kirkpatrick [6] showed that this problem is polynomial time solvable. They gave a characterization of graphs admitting partition of the vertex set into subsets inducing graphs isomorphic to K_2 and/or K_3.

Let X be an m-element set and let a be a fixed element in X. Define $L_n^m = (X, \mathcal{P}_n(X \setminus \{a\}))$. For an n-uniform hypergraph $H = (X, \mathcal{E})$, by $\overline{H}$ we mean the complement of H, i.e., the hypergraph $(X, \mathcal{P}_n(X) \setminus \mathcal{E})$.

Clearly, partitions of the vertex set of a complete n-uniform hypergraph K_m^n whose edges are colored with two colors are equivalent to partitions of the vertex set of an arbitrary hypergraph H into subsets inducing complete subhypergraphs and/or their complements. Clearly not all hypergraphs admit such partitions; however, the following theorem by Lonc and Truszczyński [8] (which was one of the motivations of our research) holds.

THEOREM. *There is an integer $S(n,k)$ such that if an n-uniform hypergraph H has at least $S(n,k)$ vertices, then the vertex set of H can be partitioned into subsets inducing subhypergraphs isomorphic to one of the following hypergraphs:*

$$K_n^m, \overline{K}_n^m, L_n^m, \overline{L}_n^m,$$

where $m \geq k$.

For a hypergraph H we denote by $V(H)$ its vertex set.

2. Results. Since our problem $\mathcal{P}_{n,c,k}$ is trivial for $n = 1$, $c = 1$, or $k = 1$, we shall assume in what follows that $n > 1$, $c > 1$, and $k > 1$.

Let H be a complete n-uniform hypergraph, the edges of which are colored with c colors. Define an (n,c,k)-*partition* of H to be a partition of the vertex set of H into subsets inducing monochromatic complete subhypergraphs of order at least k.

By an (n,c,k)-*covering* of a vertex set $R \subseteq V(H)$ we mean an (n,c,k)-partition of any set T such that $R \subseteq T \subseteq V(H)$ and each member of the partition has at least one vertex in R.

Let $r_{n,c}(k)$ be the smallest integer such that any complete n-uniform hypergraph of order at least $r_{n,c}(k)$ whose edges are colored with c colors contains a monochromatic subhypergraph of order k. Existence of $r_{n,c}(k)$ follows from the Ramsey theorem for uniform hypergraphs (see [5]).

For $i = 1, 2, 3, \ldots$, we define the following sequence $r_{n,c}^i(k)$. Let $r_{n,c}^1(k) = r_{n,c}(k)$ and $r_{n,c}^i(k) = r_{n,c}(r_{n,c}^{i-1}(k))$ for $i > 1$.

For a graph G and $A \subseteq V(G)$ denote by $N_G(A)$ the set of neighbors in G of vertices of A.

LEMMA 2. *Let H be a complete n-uniform hypergraph, the edges of which are colored with c colors. If a set $S \subseteq V(H)$ has an (n,c,k)-covering, then there exists an (n,c,k)-covering π of S such that*

$$\left|\left(\bigcup_{F\in\pi} F\right) - S\right| < (k-1)^2 \sum_{i=1}^{k-1} r_{n,c}^i(k).$$

Proof. Let π be an (n,c,k)-covering of S which minimizes the number $|(\bigcup_{F\in\pi} F)-S|$.

Define

$$\pi_i = \{F \in \pi : |F \cap S| = i\}$$

for $i = 1,2,\dots,k-1$. By minimality of π, we can assume that $|F| = k$ for every $F \in \pi_i$. Let

$$S_1 = S \cap \bigcup_{F\in\pi_1} F.$$

Clearly there is exactly one vertex from each member of π_1 in S_1. Notice that if $|S_1| \geq r_{n,c}(k)$, then S_1 contains a set F_1 of k vertices inducing a monochromatic complete hypergraph. There are exactly k members $G_1, G_2, \dots, G_k$ of π_1 such that $F_1 \cap G_j \neq \emptyset$ for $j = 1,\dots,k$. The set $\pi' = (\pi - \{G_1,\dots,G_k\}) \cup \{F_1\}$ is an (n,c,k)-covering of S such that

$$\left|\left(\bigcup_{F\in\pi'} F\right) - S\right| \leq \left|\left(\bigcup_{F\in\pi} F\right) - S\right| - k(k-1) < \left|\left(\bigcup_{F\in\pi} F\right) - S\right|,$$

a contradiction with the minimality of π. We have shown that $|S_1| < r_{n,c}(k)$.

For $i = 2,3,\dots,k-1$, define

$$S_i = S \cap \bigcup_{F\in\pi_i} F.$$

There are exactly i vertices of each member of π_i in S_i. Divide S_i into sets $A_1,\dots,A_i$ by choosing to each of them one vertex from each member of π_i. Let $\mathcal{G}_i$ be an i-partite graph with vertex classes $A_1,\dots,A_i$. Two vertices are joined by an edge in $\mathcal{G}_i$ if and only if they belong to the same member of π_i. Clearly, $\mathcal{G}_i$ is isomorphic to the disjoint union of $|\pi_i|$ copies of complete graphs on i vertices, $|A_1| = |A_2| = \dots = |A_i| = |\pi_i| = \frac{|S_i|}{i}$, and each vertex from A_j (for $j = 1,\dots,i$) is joined in $\mathcal{G}_i$ with exactly one vertex from A_h for $h \neq j$. If $|A_i| = \frac{|S_i|}{i} \geq r_{n,c}^i(k)$, then A_i contains a set of vertices, L_i, say, of order $r_{n,c}^{i-1}(k)$ which induces a complete monochromatic subhypergraph of H. The set $N_{\mathcal{G}_i}(L_i) \cap A_{i-1}$ contains a set L_{i-1} of order $r_{n,c}^{i-2}(k)$ inducing a complete monochromatic subhypergraph of H. Similarly we construct the sets L_j of order $r_{n,c}^j(k)$ for $j = i-2,\dots,2$. Finally, $|N_{\mathcal{G}_i}(L_2) \cap A_1| = r_{n,c}(k)$ since $|L_2| = r_{n,c}(k)$, so $N_{\mathcal{G}_i}(L_2) \cap A_1$ contains a k-element set L_1 inducing a complete monochromatic subhypergraph of H.

Note that the sets $B_1 = L_1$ and $B_j = N_{\mathcal{G}_i}(L_1) \cap A_j$, for $j = 2,\dots,i$, induce complete monochromatic subhypergraphs of H of order k.

Let $H_1, H_2, \ldots, H_k$ be the members of π_i whose vertex sets intersect B_1. Clearly, they also intersect B_j for $j = 2, 3, \ldots, i$. Hence $\pi' = (\pi - \{H_1, \ldots, H_k\}) \cup \{B_1, \ldots, B_i\}$ is an (n, c, k)-covering of S such that

$$\left|\left(\bigcup_{F\in\pi'} F\right) - S\right| \le \left|\left(\bigcup_{F\in\pi} F\right) - S\right| - k^2 + ki < \left|\left(\bigcup_{F\in\pi} F\right) - S\right|,$$

a contradiction with the minimality of π. Hence

$$|A_i| = \frac{|S_i|}{i} < r^i_{n,c}(k),$$

for $i = 2, 3, \ldots, k-1$, so $|S_i| < ir^i_{n,c}(k) \le (k-1)r^i_{n,c}(k)$. Consequently,

$$\left|\left(\bigcup_{F\in\pi} F\right) - S\right| = \sum_{i=1}^{k-1}(k-i)|S_i| \le \sum_{i=1}^{k-1}(k-1)|S_i| < (k-1)^2 \sum_{i=1}^{k-1} r^i_{n,c}(k). \qquad \square$$

Define $r(n, c, k) = (k-1)^2 \sum_{i=1}^{k-1} r^i_{n,c}(k)$ and $t(n, c, k) = r_{n,c}(r(n, c, k) + k)$. In the proof of the next theorem we follow some ideas of Erdös, Tuza, and Valtr [2].

THEOREM 3. *For any complete n-uniform hypergraph H whose edges are colored with c colors, H admits an (n, c, k)-partition if and only if for every subset $S \subseteq V(H)$, $|S| < t(n, c, k)$, there exists in H an (n, c, k)-covering of S.*

Proof. Since the "only if" part of the theorem is trivial, we pass on to the proof of the "if" part. Let $M = r(n, c, k)+k$. By the Ramsey theorem for uniform hypergraphs (see [5]) we can partition the set of vertices of H into subsets $S(H)$ and $T(H)$ such that $|S(H)| < t(n, c, k) = r_{n,c}(M)$ and the set $T(H)$ induces a hypergraph H' admitting an (n, c, M)-partition. (Just delete from $V(H)$ subsets inducing complete monochromatic subhypergraphs of order at least M as many times as possible. Denote by $S(H)$ the set of vertices that remain and by $T(H)$ the set of deleted vertices.) Let us consider any (n, c, M)-partition of H' and denote it by α. By our assumptions, there is an (n, c, k)-covering of $S(H)$. It follows from Lemma 2 that there exists such an (n, c, k)-covering of $S(H)$, say, π, that the set $|(\bigcup_{F\in\pi} F) - S(H)|$ consists of less than $r(n, c, k)$ elements. After deleting from the vertex set of H all subsets belonging to π, we obtain a hypergraph H'' which is a subhypergraph of $H - S(H)$. Since $M = r(n, c, k) + k$, for every $G \in \alpha$,

$$\left|G - \bigcup_{F\in\pi} F\right| \ge M - \left|\left(\bigcup_{F\in\pi} F\right) - S(H)\right| > M - r(n, c, k) = k,$$

so H'' admits an (n, c, k)-partition, β, say, consisting of subsets of elements of α. The set $\beta \cup \pi$ is an (n, c, k)-partition of H. $\square$

Proof of Theorem 1. We claim that the following algorithm solves the problem $\mathcal{P}_{n,c,k}$ in polynomially many steps.

ALGORITHM.

1. Delete from the vertex set of H subsets inducing complete monochromatic subhypergraphs of order at least $M = r(n, c, k)+k$ as many times as possible. Denote the set of remaining vertices by $S(H)$.
2. Check if there exists a subset T of $V(H) \setminus S(H)$ of order smaller than $r(n, c, k)$ such that the subhypergraph induced by $S(H) \cup T$ in H has an (n, c, k)-partition. If the answer is YES, then STOP (H admits an (n, c, k)-partition); otherwise STOP (H does not admit an (n, c, k)-partition).

Denote by π the set of subsets deleted in part 1 of the algorithm. Clearly π is an (n, c, M)-partition of the hypergraph $H - S(H)$.

If the answer to the question in part 2 is YES, then denote by π' an (n, c, k)-partition of $S(H) \cup T$. For an arbitrary set $F \in \pi$, we get

$$|F - (S(H) \cup T)| = |F - T| > M - r(n, c, k) = k,$$

so the family $\pi' \cup \{F - (S(H) \cup T) : F \in \pi\}$ is an (n, c, k)-partition of H.

If the answer to the question in part 2 is NO, then by Lemma 2, $S(H)$ does not admit an (n, c, k)-covering, and, consequently, by Theorem 3 (since $|S(H)| < r_{n,c}(M) = t(n, c, k)$) H does not admit an (n, c, k)-partition. We have shown the correctness of the algorithm.

To prove polynomiality of the algorithm observe first that both deciding if H contains a complete monochromatic subhypergraph of cardinality M and finding such a subhypergraph, if it exists, can be done in a constant time (with respect to the order of H). Indeed, if $|V(H)| < r_{n,c}(M)$, then we just check all possible M-element subsets of $V(H)$. If $|V(H)| \geq r_{n,c}(M)$, then by Ramsey's theorem in any $r_{n,c}(M)$-element subset Y of $V(H)$ there is a complete monochromatic subgraph of cardinality M. We find it again by checking all M-element subsets of Y. Therefore we need only linear with respect to the order of H number of steps to complete part 1 of the algorithm.

Since the cardinalities of the sets $S(H) \cup T$ appearing in part 2 of the algorithm are smaller than $t(n, c, k)+r(n, c, k)$ (a constant with respect to the order of H again), by a brute force method of checking all possibilities we are able to complete part 2 of the algorithm in time polynomial with respect to the order of H. □

REFERENCES

[1] K. Bryś and Z. Lonc, *Clique and anticlique partition of graphs*, Discrete Math., 185 (1998), pp. 41–49.

[2] P. Erdös, Z. Tuza, and P. Valtr, *Ramsey-remainder*, European J. Combin., 17 (1996), pp. 519–532.

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Francisco, 1979.

[4] O. Favaron, Z. Lonc, and M. Truszczyński, *Decomposition of graphs into graphs with three edges*, Ars Combin., 20 (1985), pp. 125–146.

[5] R. L. Graham, B. L. Rothschild, and J. H. Spencer, *Ramsey Theory*, Wiley, New York, 1990.

[6] D. G. Kirkpatrick and P. Hell, *On the complexity of general graph factor problems*, SIAM J. Comput., 12 (1983), pp. 601–609.

[7] Z. Lonc, *Delta-system decompositions of graphs*, Discrete Appl. Math., 164 (1997), pp. 221–224.

[8] Z. Lonc and M. Truszczyński, *Decomposition of large uniform hypergraphs*, Order, 1 (1985), pp. 345–350.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 291–298

AN EFFICIENT ALGORITHM FOR THE RING LOADING PROBLEM WITH INTEGER DEMAND SPLITTING*

YOUNG-SOO MYUNG†

Abstract. In the ring loading problem, traffic demands are given for each pair of nodes in an undirected ring network and a flow is routed in either of two directions, clockwise and counterclockwise. The load of an edge is the sum of the flows routed through the edge and the objective of the problem is to minimize the maximum load on the ring. Myung [*J. Korean OR and MS Society*, 23 (1998), pp. 49–62 (in Korean)] has presented an efficient algorithm for solving a problem where flow is restricted to integers. However, the proof for the validity of the algorithm in their paper is long and complicated and as the paper is written in Korean, its accessibility is very limited. In this paper, we slightly modify their algorithm and provide a simple proof for the correctness of the proposed algorithm.

Key words. ring loading problem, integer programming, polynomial algorithm

AMS subject classifications. 90B10, 90C27

PII. S0895480199358709

1. Introduction.

The ring loading problem (RLP) is defined on an undirected ring network $R = (V, L)$ with a node set $V = \{1, 2, \ldots, n\}$ and an edge set $L = \{(1,2), (2,3), \ldots, (n-1, n), (n, 1)\}$. We will also refer to edge $(i, i+1)$ as edge i except that $(n, 1)$ is referred to as edge n. Let K be the index set of the selected origin-destination pairs of nodes. For each $k \in K$, we are given r_k units of flow requirements (demands) and let $o(k)$ and $d(k)$, where $o(k) < d(k)$, denote its origin and destination nodes, respectively. The demand between $o(k)$ and $d(k)$ can be routed in either of the two directions, clockwise and counterclockwise. We say that a flow is routed in the *clockwise* (*counterclockwise*) direction if a flow passes through the node sequence $\{o(k), o(k)+1, \ldots, d(k)-1, d(k)\}$ ($\{o(k), o(k)-1, \ldots, 1, n, \ldots, d(k)+1, d(k)\}$). Since all the constituent edges of a ring must have the same capacity, the capacity of a ring is determined by the maximum of the traffic loads imposed on its edges. Therefore, the ring capacity depends on how to route each flow requirement on the ring. The objective of the RLP is then to find an optimal routing which minimizes the maximum edge load.

The RLP arises when designing synchronous optical network (SONET) bidirectional self-healing rings (SHRs). For more details on SONET SHRs, we refer to Wu [11] and Cosares et al. [1]. In a bidirectional SHR, demands may or may not be allowed to be split between both directions. We thus have two kinds of the RLP: with and without demand splitting. For example, in the RLP without demand splitting (RLPWO), each demand must be entirely routed in either of the two directions, while in the RLP with demand splitting (RLPW), each demand can be split between the two directions. In some cases of the RLPW, the flow of each demand in each direction may be restricted to integers. For example, the capacity of an edge is usually determined as the multiples of unit capacity. We will refer to the RLP with integer demand splitting (RLPWI) as this integer version of the RLPW while the RLPW is

*Received by the editors July 12, 1999; accepted for publication (in revised form) February 2, 2001; published electronically May 22, 2001.
http://www.siam.org/journals/sidma/14-3/35870.html
†Department of Business Administration, Dankook University, Cheonan, Chungnam 330-714, Korea (myung@dankook.ac.kr).

referred to as the problem which allows fractional flow. To avoid a trivial case, we assume integer demands in the RLPWI. Even when the demands are integers, the optimal solution of the RLPWI, in general, may have noninteger (but half-integer) components [4, 9].

Due to its practical significance, the RLP has been considered in many researches. Cosares and Saniee [2] have presented several heuristics for the RLPWO. Myung, Kim, and Tcha [5] have developed an $O(n|K|)$ algorithm for the RLPW and an efficient approximation procedure for the RLPWO. Shyur, Wu, and Chen [8] have dealt with the RLPWI and tested a heuristic. Lee and Chang [4] have developed an approximation algorithm for the RLPWI which produces an approximate solution whose objective value is at most one unit higher than the optimal value. Vachani, Shulman, and Kubat [10] have developed two $O(n^3)$ algorithms: one for solving the RLPW and the other for the RLPWI. Schrijver, Seymour, and Winkler [9] have also developed an $O(n^2|K|)$ algorithm which solves the RLPW and an approximation heuristic for the RLPWO. They also noted that their heuristic for the RLPWO can be applied to finding an integer optimal solution for the RLPWI if all demands are equal to 1 and multiple demands are allowed to the same origin-destination pair. A direct application of their heuristic results in a pseudopolynomial algorithm for a general instance of the RLPWI. Myung [6] has developed an $O(n|K|)$ algorithm for solving the RLPWI. Although this algorithm is efficient, their proof for showing the validity of the algorithm is rather long and complicated. Moreover, the accessibility of the paper is quite limited since it has been written in Korean.

The main contribution of this paper is to provide a simple proof for the correctness of the modified version of Myung's algorithm in [6]. The paper is organized as follows. Section 2 introduces the notation and develops theorems which simplify the proof of validity of the modified algorithm. In section 3, we present the modified solution procedure for the RLPWI and show its validity. Finally, section 4 provides concluding remarks.

2. Notation and theorems. For each $k \in K$, let $L_k^+ = \{(i, i+1) \in L | o(k) \leq i < d(k)\}$ and $L_k^- = L \setminus L_k^+$. Then $L_k^+ (L_k^-)$ denotes the set of edges contained in the clockwise (counterclockwise) direction path from $o(k)$ to $d(k)$. Note that edge n is contained in L_k^- for all $k \in K$. For each $l \in L$, let $K_l^+ = \{k \in K | l \in L_k^+\}$ and $K_l^- = \{k \in K | l \in L_k^-\}$. Then $K_l^+ (K_l^-)$ is the index set of origin-destination pairs whose clockwise (counterclockwise) direction path contains l. Note that $K_l^- = K \setminus K_l^+$.

For each $k \in K$, let's define variable x_k which denotes the amount of the total demand between $o(k)$ and $d(k)$ routed in the clockwise direction. Let $X = \{\mathbf{x} \in R^{|K|} | 0 \leq x_k \leq r_k \text{ for each } k \in K\}$ and for a given $\mathbf{x} \in X$, let

$$g(\mathbf{x}, l) = \sum_{k \in K_l^+} x_k + \sum_{k \in K_l^-} (r_k - x_k) \quad \text{for each} \quad l \in L.$$

Then $g(\mathbf{x}, l)$ denotes the sum of the flows routed through edge l. Let $F(\mathbf{x}) = \max_{l \in L} g(\mathbf{x}, l)$. For a given $\mathbf{x} \in X$, we will call an edge $l \in L$ the maximum load edge (with respect to $\mathbf{x}$) if $g(\mathbf{x}, l) = F(\mathbf{x})$ and we let $L(\mathbf{x})$ denote the set of the maximum load edges, i.e., $L(\mathbf{x}) = \{l \in L | g(\mathbf{x}, l) = \max_{i \in L} g(\mathbf{x}, i)\}$. The RLPW can be represented as follows:

$$(\bar{P}) \qquad z_{LP} = \min_{\mathbf{x} \in X} F(\mathbf{x}).$$

Let $X_I = X \cap Z^{|K|}$; then the RLPWI can be represented as follows:

$$(P) \qquad z_I = \min_{\mathbf{x} \in X_I} F(\mathbf{x}).$$

Any pair of distinct edges i and j constitute a cut and we define D_{ij} for each cut $\{i, j\}$ as

$$D_{ij} = \sum \{r_k : i \in L_k^+ \text{ and } j \in L_k^-, \text{ or } i \in L_k^- \text{ and } j \in L_k^+\}.$$

D_{ij} can be interpreted as the total demand across the cut $\{i, j\}$ and can be expressed with respect to a given $\mathbf{x} \in X$ as follows.

Remark 1. For any $\mathbf{x} \in X$ and two distinct edges i and j,

$$D_{ij} = g(\mathbf{x}, i) + g(\mathbf{x}, j) - 2 \sum_{k \in K_i^+ \cap K_j^+} x_k - 2 \sum_{k \in K_i^- \cap K_j^-} (r_k - x_k). \tag{2.1}$$

Therefore, for any pair of two distinct edges i and j, $g(\mathbf{x}, i) + g(\mathbf{x}, j) \geq D_{ij}$ and $D_{ij} = g(\mathbf{x}, i) + g(\mathbf{x}, j)$ if and only if i and j satisfy the following two conditions:

(C1) $x_k = 0$ for each $k \in K_i^+ \cap K_j^+$ and
(C2) $x_k = r_k$ for each $k \in K_i^- \cap K_j^-$.

Let D_{max} be the maximum value of D_{ij} for all cuts. We say that a cut $\{i, j\}$ is *tight* if $D_{ij} = D_{max}$. Remark 1 indicates that $F(\mathbf{x}) \geq D_{max}/2$ for any $\mathbf{x} \in X$. On the other hand, there exists a feasible solution $\mathbf{x} \in X$ such that $F(\mathbf{x}) \leq D_{max}/2$. This fact can be shown as a special case of Okamura and Seymour's theorem [7] and the proofs of this specialized version were also provided by Vachani, Shulman, and Kubat [10] and Schrijver, Seymour, and Winkler [9] independently. Therefore, $\mathbf{x} \in X$ is an optimal solution of $(\bar{P})$ if and only if $F(\mathbf{x}) = D_{max}/2$. Therefore, the following statement is straightforward.

LEMMA 1. *For any $\mathbf{x} \in X$, if $g(\mathbf{x}, i) = g(\mathbf{x}, j) = F(\mathbf{x})$ for some cut $\{i, j\}$ and $\mathbf{x}$ satisfies* (C1) *and* (C2)*, then $\mathbf{x}$ is an optimal solution of $(\bar{P})$ and the cut $\{i, j\}$ is tight.*

Myung, Kim, and Tcha [5] have developed an algorithm which is called EXACT and finds an optimal solution of the RLPW in $O(n|K|)$ time. For later use, we briefly explain the algorithm EXACT and the properties of the solutions produced by the algorithm.

ALGORITHM EXACT.

(Step 1) Reorder the indices of K as follows: if $o(k_1) < o(k_2)$, then $k_1 < k_2$, and if $o(k_1) = o(k_2)$ and $d(k_1) > d(k_2)$, then $k_1 < k_2$.

(Step 2) Initially, all demands are routed in the clockwise direction.

(Step 3) For each $k \in K$ according to the increasing order of K, reroute all or a part of demand k in the counterclockwise direction if rerouting would decrease the ring capacity.

In Step 1, the ordering of the origin-destination pairs of $|K|$ can be done within $O(|K| \log |K|)$ time and the sequence of the indices in K plays an important role in constructing an optimal solution. The rerouting procedure in Step 3 is very simple and requires $O(n|K|)$ time. For any $k \in K$, if $\max_{l \in L_k^+} g(\mathbf{x}, l) > \max_{l \in L_k^-} g(\mathbf{x}, l)$, that is, $L(\mathbf{x}) \subseteq L_k^+$, rerouting demand k in the counterclockwise direction decreases

the resulting ring capacity. Since demand k is examined at the kth iteration of the rerouting procedure, we use the same index k for indicating both an origin-destination pair and an iteration step. Let $\mathbf{x}^k$ denote the solution obtained after the rerouting step is performed for $k \in K$ and $\mathbf{x}^0 = \{r_1, \ldots, r_{|K|}\}$. Then $\mathbf{x}^{|K|}$ is the solution which EXACT finally produces. At iteration k, demand k is rerouted until either all the demand is routed in the counterclockwise direction or the resulting solution satisfies $\max_{l \in L_k^+} g(\mathbf{x}^k, l) = \max_{l \in L_k^-} g(\mathbf{x}^k, l)$.

Now, we introduce our two main theorems on the properties of $\mathbf{x}^{|K|}$, based on which we develop the algorithm for the RLPWI which we will present in the next section. Before presenting the theorems, we introduce some preliminary results that have appeared in Myung, Kim, and Tcha [5]. Let $K^0 = \{k \in K : x_k^{|K|} = 0\}$, $K^r = \{k \in K : x_k^{|K|} = r_k\}$, and $K^b = \{k \in K : 0 < x_k^{|K|} < r_k\}$. Then the following lemma is straightforward.

LEMMA 2 (Myung, Kim, and Tcha [5]). *For each $k = 1, 2, \ldots, |K|$, the following relations hold:*

(i) $L(\mathbf{x}^{k-1}) \subseteq L(\mathbf{x}^k)$;

(ii) $L(\mathbf{x}^{k-1}) \subseteq L_k^+$ *if and only if* $k \in K^0 \cup K^b$; *and*

(iii) *if* $k \in K^r \cup K^b$, *then* $L(\mathbf{x}^k) \setminus L_k^+ \neq \emptyset$.

Let $l_0 = \min L(\mathbf{x}^0)$ and $l_{max} = \max L(\mathbf{x}^{|K|})$. In other words, l_0 is the edge having the smallest index among the maximum load edges with respect to the solution before rerouting starts, and l_{max} is the edge having the largest index among maximum load edges with respect to the solution obtained after rerouting is completed. Note that l_0 remains as a maximum load edge with respect to $\mathbf{x}^{|K|}$ but may or may not be the one with the smallest index among the maximum load edges with respect to $\mathbf{x}^{|K|}$.

THEOREM 3. *For any maximum load edge l_1 with respect to $\mathbf{x}^{|K|}$ such that $l_1 < l_0$, there exists a maximum load edge with respect to $\mathbf{x}^{|K|}$ corresponding to l_1, say, l_2, such that $l_2 \geq l_0$ and $g(\mathbf{x}^{|K|}, l_1) + g(\mathbf{x}^{|K|}, l_2) = D_{l_1,l_2}$, i.e., $\{l_1, l_2\}$ constitutes a tight cut.*

Proof. We will show that the following selection of l_2 corresponding to l_1 satisfies the condition described in the theorem. If $K_{l_1}^+ = \emptyset$, then set $l_2 = \max L(\mathbf{x}^0)$; otherwise, set $l_2 = \max\{l \in L | l \in L(\mathbf{x}^k)$ for some k such that $k \in K_{l_1}^+\}$. Obviously, $l_0 \leq l_2$. Now, we show that $\mathbf{x}^{|K|}$ and $\{l_1, l_2\}$ satisfy (C1) and (C2) in Remark 1 which means that $g(\mathbf{x}^{|K|}, l_1) + g(\mathbf{x}^{|K|}, l_2) = D_{l_1,l_2}$. We first prove that l_1 and l_2 satisfy (C1). Suppose that $k \in K_{l_1}^+ \cap K_{l_2}^+$ for some $k \in K$, i.e., $\{l_1, l_2\} \subseteq L_k^+$. We claim that $L(\mathbf{x}^k) \subseteq L_k^+$. If our claim is true, (C1) holds because $k \in K^0$ by Lemma 2. To prove our assertion, we first show that $l_1 \leq \min L(\mathbf{x}^k) \leq \max L(\mathbf{x}^k) \leq l_2$. Note that $K_{l_1}^+ \neq \emptyset$. By the definition of l_2, $\max L(\mathbf{x}^k) \leq l_2$. Since $l_1 < \min L(\mathbf{x}^0)$, l_1 must have become a maximum load edge after at least one rerouting iteration was performed. Therefore, there must exist $k' > 0$ such that $\min L(\mathbf{x}^{k'}) \leq l_1 < \min L(\mathbf{x}^{k'-1})$ and $l_1 < \min L_{k'}^+$. Since $l_1 \in L_k^+$, $o(k) < o(k')$ which means that $k < k'$. Therefore, $l_1 < \min L(\mathbf{x}^k)$ since $L(\mathbf{x}^k) \subseteq L(\mathbf{x}^{k'-1})$. From the fact that $l_1 \leq \min L(\mathbf{x}^k) \leq \max L(\mathbf{x}^k) \leq l_2$ and $\{l_1, l_2\} \subseteq L_k^+$, $L(\mathbf{x}^k) \subseteq L_k^+$.

Second, we prove that l_1 and l_2 satisfy (C2). Suppose that $k \in K_{l_1}^- \cap K_{l_2}^-$, i.e., $\{l_1, l_2\} \subseteq L_k^-$, and also suppose that $x_k^{|K|} < r_k$, i.e., $k \in K^0 \cup K^b$. Since $k \in K^0 \cup K^b$, $l_0 \in L_k^+$ by Lemma 2. Therefore, $l_1 < \min L_k^+ \leq \max L_k^+ < l_2$. By our definition of l_2, there is a $k' \in K_{l_1}^+$ such that $l_2 \in L(\mathbf{x}^{k'})$. Therefore, $k' < k$, and hence $l_2 \in L(\mathbf{x}^{k-1})$. Therefore, $L(\mathbf{x}^{k-1}) \not\subseteq L_k^+$ which contradicts our assumption that $x_k^{|K|} < r_k$ by (ii) of Lemma 2. □

Theorem 3 implies the correctness of algorithm EXACT and has also appeared in Myung [6]. However, here we have provided a slightly simplified proof for completeness. Along with Theorem 3, Theorem 4 is useful to prove the validity of the algorithm for the RLPWI which will be presented in the next section.

THEOREM 4. *If $l_0 \neq l_{max}$ and there exists no $k \in K^+_{l_0} \cap K^+_{l_{max}}$ such that $x_k^{|K|} > 0$, then $\{l_0, l_{max}\}$ is a tight cut.*

Proof. Suppose that $l_0 \neq l_{max}$. By definition of l_0 and l_{max}, $g(\mathbf{x}^{|K|}, l_0) = g(\mathbf{x}^{|K|}, l_{max}) = F(\mathbf{x}^{|K|})$. Since $l_0 \in L_k^+$ for all $k \in K^0 \cup K^b$, we know that $x_k^{|K|} = r_k$ for all $k \in K^-_{l_0} \cap K^-_{l_{max}}$. Therefore, $\{l_0, l_{max}\}$ is a tight cut by Lemma 1. □

3. An $O(n|K|)$ algorithm for the RLPWI. In this section, we present an $O(n|K|)$ algorithm which solves the RLPWI. This algorithm is similar to the one presented in Myung [6]. However, here we give a simple proof for the validity of the proposed algorithm using the theorems developed in section 2. Our algorithm starts from $\mathbf{x}^{|K|}$, an optimal solution of $(\bar{P})$ produced by EXACT. If $\mathbf{x}^{|K|}$ is integral, it is also an optimal solution of the RLPWI. Suppose that $\mathbf{x}^{|K|}$ is not integral. Let $K^f = \{k_1, \ldots, k_s\}$ be the index set of the origin-destination pairs in K for which $x_k^{|K|}$ has a fractional value. Then, $K^f \subseteq K^b$ and by our assumption on the ordering of the indices of K and (iii) of Lemma 2, $o(k_1) < o(k_2) < \cdots < o(k_s) < d(k_1) < \cdots < d(k_s)$. Let's partition L into the following $2s+1$ subsets as follows: $L_0 = \{l \in L : 1 \leq l < o(k_1)\}$, $L_1 = \{l \in L : o(k_1) \leq l < o(k_2)\}$, …, $L_s = \{l \in L : o(k_s) \leq l < d(k_1)\}$, …, $L_{2s} = \{l \in L : d(k_s) \leq l \leq n\}$. Note that L_0 may be an empty set and that $L(\mathbf{x}^0) \subseteq L_s$ because $L(\mathbf{x}^0) \subseteq L_k^+$ for all $k \in K^f$. So $l_0 \in L_s$.

If we consider the rerouting procedure of EXACT, it is not difficult to know that the fractional part of $x_k^{|K|}$ for each $k \in K^f$ is equal to 0.5. So, if we reroute each demand $k \in K^f$ by 0.5 in either a clockwise or a counterclockwise direction, we can obtain an integer solution. Moreover, we will show that if rerouting is done carefully, an integer optimal solution of the RLPWI can be obtained. As we already mentioned, our algorithm starts from $\mathbf{x}^{|K|}$ produced by EXACT. If $\mathbf{x}^{|K|}$ is not integral, i.e., $K^f \neq \emptyset$, we reroute each demand $k \in K^f$ by 0.5 in either a clockwise or a counterclockwise direction. We define the two different rerouting methods, *method A* and *method B*. Both methods reroute each demand $k \in K^f$ by the amount of 0.5, in the increasing order of $k \in K^f$ and in either of the two directions alternatingly, one after another. The difference of the two methods is that method A starts iteration by rerouting the first flow in the clockwise direction while method B starts rerouting in the counterclockwise direction. More precisely, let $\mathbf{x}^*$ be the integer solution obtained by rerouting the fractional flows of $\mathbf{x}^{|K|}$ using either method A or B. Let $z_{k_t} = x^*_{k_t} - x^{|K|}_{k_t}$ for each $k_t \in K^f$. If $\mathbf{x}^*$ is the result of method A, then $z_{k_t} = 0.5$ if t is odd and $z_{k_t} = -0.5$ if t is even. If x^* is the result of method B, then $z_{k_t} = -0.5$ if t is odd and $z_{k_t} = 0.5$ otherwise. Note that for each $l \in L$

$$g(\mathbf{x}^*, l) = g(\mathbf{x}^{|K|}, l) + \sum_{k \in K_l^+ \cap K^f} z_k - \sum_{k \in K_l^- \cap K^f} z_k.$$

Now we present an algorithm which produces an optimal integral solution.

ALGORITHM INTEGER.

(Step 1) Implement EXACT. If $K^f \neq \emptyset$, go to Step 2.

(Step 2) If $|K^f|$ is odd, reroute $\mathbf{x}^{|K|}$ using either method A or B. Otherwise, go to Step 3.

(Step 3) If no maximum load edge belongs to $L_1 \cup L_3 \cup \cdots \cup L_{s-1}$, reroute $\mathbf{x}^{|K|}$ using method A. Otherwise, go to Step 4.

(Step 4) Reroute $\mathbf{x}^{|K|}$ using method B. If $l_{max} \notin L_{2s}$, reroute one unit of demand k_s in the counterclockwise direction. If $l_{max} \in L_{2s}$ and there exists some $k \in K_{l_0}^+ \cap K_{l_{max}}^+$ such that $x_k^{|K|} > 0$, then reroute one unit of demand k in the counterclockwise direction.

The following observation is useful to prove the validity of the algorithm.

Remark 2. (a) If $|K^f|$ is odd and $\mathbf{x}^*$ is the integral solution obtained by rerouting $\mathbf{x}^{|K|}$ using either method A or B, then for each $l \in L$,

$$g(\mathbf{x}^{|K|}, l) - 0.5 \le g(\mathbf{x}^*, l) \le g(\mathbf{x}^{|K|}, l) + 0.5.$$

(b) If $|K^f|$ is even and $\mathbf{x}^*$ is the result of method A,

$$g(\mathbf{x}^*, l) = \begin{cases} g(\mathbf{x}^{|K|}, l) + 1 & \text{if } l \in L_1 \cup L_3 \cup \cdots \cup L_{s-1}, \\ g(\mathbf{x}^{|K|}, l) & \text{if } l \in L_0 \cup L_2 \cup \cdots \cup L_s \cup \cdots \cup L_{2s}, \\ g(\mathbf{x}^{|K|}, l) - 1 & \text{if } l \in L_{s+1} \cup L_{s+3} \cup \cdots \cup L_{2s-1}. \end{cases}$$

(c) If $|K^f|$ is even and $\mathbf{x}^*$ is the result of method B,

$$g(\mathbf{x}^*, l) = \begin{cases} g(\mathbf{x}^{|K|}, l) + 1 & \text{if } l \in L_{s+1} \cup L_{s+3} \cup \cdots \cup L_{2s-1}, \\ g(\mathbf{x}^{|K|}, l) & \text{if } l \in L_0 \cup L_2 \cup \cdots \cup L_s \cup \cdots \cup L_{2s}, \\ g(\mathbf{x}^{|K|}, l) - 1 & \text{if } l \in L_1 \cup L_3 \cup \cdots \cup L_{s-1}. \end{cases}$$

When $|K^f|$ is even, the following two lemmas are useful to get information on an optimal integral solution. We say that a cut $\{i, j\}$ is odd if $D_{max} - D_{ij}$ is odd. We also say that a ring network satisfies the *parity condition* if for every odd cut $\{i, j\}$, at least one of i and j does not belong to any tight cut. Then the next lemma is the direct consequence of Theorem 2.2 in Frank et al. [3] and Theorem 6.1 in Schrijver, Seymour, and Winkler [9].

LEMMA 5. *Suppose that $|K^f|$ is even. If a ring network satisfies the parity condition, then $z_I = z_{LP}$; otherwise, $z_I = z_{LP} + 1$.*

Based on the above lemma, we can show that the following lemma holds.

LEMMA 6. *Suppose that $|K^f|$ is even. For any pair of edges i and j such that $i \in L_t$ for some odd index $t \le s$, and $j \in L_t$ for some even index $t \le s$, if both edges belong to some tight cut, then the cut $\{i, j\}$ violates the parity condition.*

Proof. By our assumption, $g(\mathbf{x}^{|K|}, i) + g(\mathbf{x}^{|K|}, j) = D_{max}$, and from (2.1) in Remark 1, $g(\mathbf{x}^{|K|}, i) + g(\mathbf{x}^{|K|}, j) - D_{ij} = 2\sum_{k \in K_i^+ \cap K_j^+} x_k^{|K|} + 2\sum_{k \in K_i^- \cap K_j^-} (r_k - x_k^{|K|})$. For each $k \in K^f$, $x_k^{|K|} = \alpha_k + 0.5$ for some integer α_k and one of $|(K_i^+ \cap K_j^+) \cap K^f|$ and $|(K_i^- \cap K_j^-) \cap K^f|$ is odd while the other is even. Therefore, $g(\mathbf{x}^{|K|}, i) + g(\mathbf{x}^{|K|}, j) - D_{ij}$ is odd. □

THEOREM 7. *The algorithm INTEGER produces an optimal solution for the RLPWI.*

Proof. If $|K^f|$ is odd, $g(\mathbf{x}^{|K|}, l)$ has a fractional value for each $l \in L$ and its fractional part is equal to 0.5. Therefore, z_{LP} also has a fractional value whose fractional part is equal to 0.5. Since $z_I \ge z_{LP} + 0.5$, rerouting by either method A or B produces an optimal solution. From now on, we assume that $s = |K^f|$ is even and show why Steps 3 and 4 are guaranteed to provide an optimal integer solution. If no maximum load edge belongs to $L_1 \cup L_3 \cup \cdots \cup L_{s-1}$, (b) of Remark 2 implies that

the integral solution obtained by rerouting $\mathbf{x}^{|K|}$ via method A has the objective value equal to z_{LP}. Suppose that we are in Step 4 and that we have some maximum load edge, say, l_1,which belongs to $L_1 \cup L_3 \cup \cdots \cup L_{s-1}$. Then exactly one of the following three cases happens.

Case 1. $l_{max} \notin L_{2s}$.

Let $\mathbf{x}^*$ be the integral solution obtained by first rerouting $\mathbf{x}^{|K|}$ via method B and additionally rerouting one unit of demand k_s in the counterclockwise direction. Then $g(\mathbf{x}^*, l) = g(\mathbf{x}^{|K|}, l) + 1$ if $l \in L_0 \cup L_2 \cup \cdots \cup L_{s-2} \cup L_{2s}$ and $g(\mathbf{x}^*, l) \leq g(\mathbf{x}^{|K|}, l)$ otherwise. Obviously, either $F(\mathbf{x}^*) = F(\mathbf{x}^{|K|})$ or $F(\mathbf{x}^*) = F(\mathbf{x}^{|K|}) + 1$. If the latter case happens, at least one edge in $L_0 \cup L_2 \cup \cdots \cup L_{s-2}$ is a maximum load edge with respect to $\mathbf{x}^{|K|}$. That edge belongs to some tight cut by Theorem 3. Since l_1 also belongs to some tight cut by Theorem 3, this violates the parity condition along with l_1 by Lemma 6.

Case 2. $l_{max} \in L_{2s}$ and there exists no $k \in K^+_{l_0} \cap K^+_{l_{max}}$ such that $x^{|K|}_k > 0$.

In this case, $\{l_0, l_{max}\}$ is a tight cut by Theorem 4 and l_1 also belongs to some tight cut by Theorem 3. Therefore, $\{l_1, l_0\}$ violates the parity condition and $z_I = z_{LP} + 1$. So, the integral solution obtained by rerouting via method B is optimal to RLPWI.

Case 3. $l_{max} \in L_{2s}$ and there exists $k \in K^+_{l_0} \cap K^+_{l_{max}}$ such that $x^{|K|}_k > 0$.

Since $x^{|K|}_k > 0$, $k \in K^b \cup K^r$. Notice that $d(k) > d(k_s)$ because $l_{max} \in L_{2s}$ and $l_{max} \in L^+_k$. In addition, we can know that $o(k) > o(k_s)$. Otherwise, $k < k_s$ by our assumption on the ordering of the origin-destination pairs of $|K|$, and $L^+_{k_s} \subseteq L^+_k$. Since $k \in K^b \cup K^r$, $L(\mathbf{x}^k) \setminus L^+_k \neq \emptyset$ by (iii) of Lemma 2. Therefore, $L(\mathbf{x}^k) \not\subseteq L^+_{k_s}$ which contradicts the fact that $k_s \in K^f \subseteq K^b$ by (ii) of Lemma 2. Let $L^1 = \{l \in L : o(k_s) \leq l < o(k)\}$ and $L^2 = \{l \in L : d(k) \leq l \leq n\}$. Note that L^2 doesn't contain any maximum load edge. Let $\mathbf{x}^*$ be the integral solution obtained by first rerouting $\mathbf{x}^{|K|}$ via method B and additionally rerouting one unit of demand k in the counterclockwise direction. Then $g(\mathbf{x}^*, l) = g(\mathbf{x}^{|K|}, l) + 1$ if $l \in L_0 \cup L_2 \cup \cdots \cup L_{s-2} \cup L^1 \cup L^2$ and $g(\mathbf{x}^*, l) \leq g(\mathbf{x}^{|K|}, l)$ otherwise. By the same way as we used in Case 1, we can show that if $F(\mathbf{x}^*) = F(\mathbf{x}^{|K|}) + 1$, the ring network violates the parity condition. □

The complexity of our algorithm for solving the RLPWI stays $O(n|K|)$ because the extra steps for deriving an integer optimal solution from $\mathbf{x}^{|K|}$ doesn't increase the complexity of EXACT.

4. Conclusion. This paper considered the RLP in which the flow of demand is allowed to be split but is restricted to integers. This problem arises in the design of SONET bidirectional rings, where the capacity of edge is selected as the multiples of unit capacity. We have developed an efficient algorithm which optimally solves the RLPWI in $O(n|K|)$ time which is faster than any other algorithms developed so far. The RLP is only a subproblem within the comprehensive planning tool, which has to be solved multiple times in practical applications. Therefore, the proposed algorithm is useful when designing a real-world, large-scale SONET broadband network.

REFERENCES

[1] S. Cosares, D. N. Deutsch, I. Saniee, and O. J. Wasem, *SONET toolkit: A decision support system for designing robust and cost-effective fiber-optic networks*, Interfaces, 25 (1995), pp. 20–40.

[2] S. Cosares and I. Saniee, *An optimization problem related to balancing loads on SONET rings*, Telecommunication Systems, 3 (1994), pp. 165–181.

[3] A. Frank, T. Nishizeki, N. Saito, H. Suzuki, and E. Tardos, *Algorithms for routing around a rectangle*, Discrete Appl. Math., 40 (1992), pp. 363–378.
[4] C. Y. Lee and S. G. Chang, *Balancing loads on SONET rings with integer demand splitting*, Comput. Oper. Res., 24 (1997), pp. 221–229.
[5] Y.-S. Myung, H.-G. Kim, and D.-W. Tcha, *Optimal load balancing on SONET bidirectional rings*, Oper. Res., 45 (1997), pp. 148–152.
[6] Y.-S. Myung, *Optimal load balancing on SONET rings with integer demand splitting*, J. Korean OR and MS Society, 23 (1998), pp. 49–62 (in Korean).
[7] H. Okamura and P. D. Seymour, *Multicommodity flows in planar graphs*, J. Combin. Theory Ser. B, 31 (1981), pp. 75–81.
[8] C. C. Shyur, Y. M. Wu, and C. H. Chen, *A capacity comparison for SONET self-healing ring networks*, IEEE GLOBECOM, 3 (1993), pp. 1574–1578.
[9] A. Schrijver, P. Seymour, and P. Winkler, *The ring loading problem*, SIAM J. Discrete Math., 11 (1998), pp. 1–14.
[10] R. Vachani, A. Shulman, and P. Kubat, *Multicommodity flows in ring networks*, INFORMS J. Comput., 8 (1996), pp. 235–242.
[11] T. H. Wu, *Fiber Network Service Survivability*, Artech House, Boston, MA, 1992.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 299–311

ON LOWER BOUNDS FOR SELECTING THE MEDIAN*

DORIT DOR†, JOHAN HÅSTAD‡, STAFFAN ULFBERG‡, AND URI ZWICK†

Abstract. We present a reformulation of the $2n + o(n)$ lower bound of Bent and John [*Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, 1985, pp. 213–216] for the number of comparisons needed for selecting the median of n elements. Our reformulation uses a weight function. Apart from giving a more intuitive proof for the lower bound, the new formulation opens up possibilities for improving it. We use the new formulation to show that any *pair-forming* median finding algorithm, i.e., a median finding algorithm that starts by comparing $\lfloor n/2 \rfloor$ disjoint pairs of elements must perform, in the worst case, at least $2.01n + o(n)$ comparisons. This provides strong evidence that selecting the median requires at least $cn + o(n)$ comparisons for some $c > 2$.

Key words. median selection, comparison algorithms, lower bounds

AMS subject classifications. 68Q25, 68R05, 06A07

PII. S0895480196309481

1. Introduction. Sorting and selection problems have received extensive attention by computer scientists and mathematicians for a long time. Comparison based algorithms for solving these problems work by performing pairwise comparisons between the elements until the relative order of all elements is known, in the case of sorting, or until the ith largest element among the n input elements is found, in the case of selection.

Sorting in a comparison based computational model is quite well understood. Any deterministic algorithm can be modeled by a decision tree in which all internal nodes represent a comparison between two elements; every leaf represents a result of the computation. Since there must be at least as many leaves in the decision tree as there are possible reorderings of n elements, all algorithms that sort n elements use at least $\lceil \log n! \rceil \geq n \log n - n \log e + o(n) \approx n \log n - 1.44n + o(n)$ comparisons in the worst case. (All logarithms in this paper are base 2 logarithms.) The best known sorting method, called *merge insertion* by Knuth [9], is due to Ford and Johnson [7]. It sorts n elements using at most $n \log n - 1.33n + o(n)$ comparisons. Thus, the gap between the upper and lower bounds is very narrow in that the error in the second order term is bounded by $0.11n$.

The problem of finding the median is the special case of selecting the ith largest in an ordered set of n elements when $i = \lceil n/2 \rceil$. Although much effort has been put into finding the exact number of required comparisons, there is still an annoying gap between the best upper and lower bounds currently known.

Knowing how to sort, we could select the median by first sorting and then selecting the middlemost element; it is quite evident that we could do better but how much better? This question received a somewhat surprising answer when Blum et al. [3] showed in 1973 how to determine the median in linear time using at most $5.43n$ comparisons. This result was improved upon in 1976 when Schönhage, Paterson, and

*Received by the editors September 18, 1996; accepted for publication (in revised form) January 30, 2001; published electronically June 5, 2001.
http://www.siam.org/journals/sidma/14-3/30948.html
†School of Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel (dorit@checkpoint.com, zwick@post.tau.ac.il).
‡Department of Numerical Analysis and Computing Science, Royal Institute of Technology, 100 44 Stockholm, Sweden (johanh@nada.kth.se, staffanu@nada.kth.se).

Pippenger [13] presented an algorithm that uses only $3n + o(n)$ comparisons. Their main invention was the use of *factories* which mass-produce certain partial orders that can easily be merged with each other.

This remained the best algorithm for almost 20 years until Dor and Zwick [5] pushed down the number of comparisons a little bit further to $2.95n + o(n)$ by adding *green factories* that recycle debris from the merging process used in the algorithm of [13].

The first nontrivial lower bound for the problem was also presented in 1973 by Blum et al. [3] using an adversary argument. Their $1.5n$ lower bound was subsequently improved to $1.75n + o(n)$ by Pratt and Yao [12] in 1973. Then Yap [14], and later Munro and Poblete [10], improved it to $\frac{38}{21}n + O(1)$ and $\frac{79}{43}n + O(1)$, respectively. The proofs of these last two bounds are long and complicated.

In 1979, Fussenegger and Gabow [8] proved a $1.5n + o(n)$ lower bound for the median using a new proof technique. Bent and John [2] used the same basic ideas when they gave a short proof in 1985 that improved the lower bound to $2n + o(n)$, which is currently the best available. Thus, the uncertainty in the coefficient of n is larger for finding the median than it is for sorting, even though the linear term is the second order term in the case of sorting.

Since our methods are based on the proof by Bent and John, let us describe it in some detail. Given the decision tree of a comparison based algorithm, they invented a method to prune it that yields a collection of pruned trees. Then, lower bounds for the number of pruned trees and for their number of leaves are obtained. A final argument saying that the leaves of the pruned trees are almost disjoint then gives a lower bound for the size of the decision tree.

In section 2 we reformulate the proof by Bent and John by assigning weights to each node in the decision tree. The weight of a node v corresponds to the total number of leaves in subtrees with root v in all pruned trees where v occurs in the proof by Bent and John. The weight of the root is approximately 2^{2n}; we show that every node v in the decision tree has a child whose weight is at least half the weight of v and that the weights of all the leaves are small.

When the proof is formulated in this way, it becomes more transparent, and one can more easily study individual comparisons to rule out some as being bad from the algorithm's point of view.

For many problems, such as finding the maximal or the minimal element of an ordered set, and finding the maximal *and* minimal element of an ordered set, there are optimal algorithms that start by making $\lfloor n/2 \rfloor$ pairwise comparisons between singleton elements. We refer to algorithms that start in this way as being *pair-forming*. It has been discussed whether there are optimal pair-forming algorithms for all partial orders, and in particular this question was posed as an open problem by Aigner [1]. Some examples were then found by Chen [4], showing that pair-forming algorithms are not always optimal.

It is interesting to note that the algorithms in [5] and [13] are both pair-forming. It is still an open problem whether there are optimal pair-forming algorithms for finding the median.

In section 3 we use our new approach to prove that any pair-forming algorithm uses at least $2.01227n + o(n)$ comparisons to find the median.

Dor and Zwick [6] have recently been able to extend the ideas described here and obtain a $(2+\epsilon)n$ lower bound, for some tiny $\epsilon > 0$, on the number of comparisons performed, in the worst case, by any median selection algorithm.

2. Bent and John revisited. Bent and John [2] proved that $2n+o(n)$ comparisons are required for selecting the median. Their result, in fact, is more general and provides a lower bound for the number of comparisons required for selecting the ith largest element for any $1 \le i \le n$. We concentrate here on median selection, although our results, like those of Bent and John, can be extended to general i.

Although the proof given by Bent and John is relatively short and simple, we present a reformulation here. There are two reasons for this: the first is that the proof gets more transparent; the second is that this formulation makes it easier to study the effect of individual comparisons.

THEOREM 2.1 (Bent and John [2]). *Finding the median requires $2n + o(n)$ comparisons.*

Proof. Any deterministic algorithm for finding the median can be represented by a decision tree T, in which each internal node v is labeled by a comparison $a : b$. The two children of such a node, $v_{a<b}$ and $v_{a>b}$, represent the outcomes $a < b$ and $a > b$, respectively. We assume that decision trees do not contain redundant comparisions between elements whose relative order has already been established.

We consider a universe U containing n elements. For every node v in T and subset C of U we make the following definitions:

$$\max_v(C) = \left\{ a \in C \;\middle|\; \begin{array}{l} \text{every comparison } a : b \text{ above } v \\ \text{with } b \in C \text{ had outcome } a > b \end{array} \right\},$$

$$\min_v(C) = \left\{ a \in C \;\middle|\; \begin{array}{l} \text{every comparison } a : b \text{ above } v \\ \text{with } b \in C \text{ had outcome } a < b \end{array} \right\}.$$

Before we proceed with the proof that selecting the median requires $2n + o(n)$ comparisons, we present a proof of a somewhat weaker result. We assume that U contains $n = 2m$ elements and show that selecting the two middlemost elements requires $2n + o(n)$ comparisons. The proof in this case is slightly simpler, yet it demonstrates the main ideas used in the proof of the theorem.

We define a weight function on the nodes of T. This weight function satisfies the following three properties: (i) the weight of the root is $2^{2n+o(n)}$; (ii) each internal node v has a child whose weight is at least half the weight of v; (iii) the weight of each leaf is small.

For every node v in the decision tree, we keep track of subsets A of size m which may contain the m largest elements with respect to the comparisons already made. Let $\mathcal{A}(v)$ contain all such sets which are called *upper half compatible* with v. The A's are assigned weights which estimate how far from a solution the algorithm is, assuming that the elements in A are the m largest. The weight of every $A \in \mathcal{A}(v)$ is defined as

$$w_v^1(A) = 2^{|\min_v(A)|+|\max_v(\bar{A})|},$$

and the weight of a node v is defined as

$$w(v) = \sum_{A \in \mathcal{A}(v)} w_v^1(A).$$

The superscript 1 in $w_v^1(A)$ is used as, we shall shortly have to define, a second weight function $w_v^2(B)$.

TABLE 2.1
The weight of a set $A \in \mathcal{A}(v)$ in the children of a node v relative to its weight in v.

case		$w^1_{v_{a<b}}(A)$	$w^1_{v_{a>b}}(A)$
$a \in A$	$b \in A$	$\frac{1}{2}$ or 1	$\frac{1}{2}$ or 1
$a \in A$	$b \in \bar{A}$	0	1
$a \in \bar{A}$	$b \in A$	1	0
$a \in \bar{A}$	$b \in \bar{A}$	$\frac{1}{2}$ or 1	$\frac{1}{2}$ or 1

In the root r of T, all subsets of size m of U are upper half compatible with r so that $|\mathcal{A}(r)| = \binom{2m}{m}$. Also, each $A \in \mathcal{A}(r)$ has weight 2^{2m}, and we find, as promised, that

$$w(r) = 2^{2m}\binom{2m}{m} = 2^{2n+o(n)}.$$

Consider the weight $w^1_v(A)$ of a set $A \in \mathcal{A}(v)$ at a node v labeled by the comparison $a : b$. What are the weights of A in v's children? This depends on which of the elements a and b belongs to A (and on which of them is minimal in A or maximal in $\bar{A}$). The four possible cases are considered in Table 2.1. The weights given there are relative to the weight $w^1_v(A)$ of A at v. A zero indicates that A is no longer compatible with this child and thus does not contribute to its weight. The weight $w^1_{v_{a<b}}(A)$, when $a, b \in A$, for example, is $\frac{1}{2}w^1_v(A)$, if $b \in \min_v(A)$, and is otherwise $w^1_v(A)$. As can be seen, v always has at least one child in which the weight of A is at least half its weight at v. Furthermore, in each one of the four cases, $w^1_{v_{a<b}}(A) + w^1_{v_{a>b}}(A) \geq w^1_v(A)$.

Each leaf v of the decision tree corresponds to a state of the algorithm in which the two middlemost elements were found. There is therefore only one set A left in $\mathcal{A}(v)$. Since we have identified the minimum element in A and the maximum element in $\bar{A}$, we get that $w^1_v(A) = 4$. Therefore, if we follow a path from the root of the tree and repeatedly descend to the child with the largest weight, we will, when we eventually reach a leaf, have performed at least $2n + o(n)$ comparisons.

We now prove that selecting the median also requires at least $2n + o(n)$ comparisons. To make the median well defined we assume that $n = 2m - 1$. The problem that arises in the above argument is that the weights of the leaves in T, when the selection of the median, and not the two middlemost elements, is considered, are not necessarily small enough: it is possible to know the median without knowing any relations between elements in $\bar{A}$ (which now contains $m - 1$ elements); this is remedied as follows.

In a node v, where the algorithm is close to determining the minimum element in A, we essentially force it to determine the largest element in $\bar{A}$ instead. This is done by moving an element a_0 out of A and creating a set $B = \bar{A} \cup \{a_0\}$. This set is *lower half compatible* with v and the median is the maximum element in B. By a suitable choice of a_0, most of $\max_v(\bar{A})$ is in $\max_v(B)$. A set B is lower half compatible with v if $|B| = m$ and it may contain the m smallest elements in U. We keep track of B's in the *multiset* $\mathcal{B}(v)$.

For the root r of T, we let $\mathcal{A}(r)$ contain all subsets of size m of U as before and let $\mathcal{B}(r)$ be empty. We exchange some A's for B's as the algorithm proceeds. The

TABLE 2.2
The weight of a set $B \in \mathcal{B}(v)$ in the children of a node v relative to its weight in v.

case		$w^2_{v_{a<b}}(B)$	$w^2_{v_{a>b}}(B)$
$a \in B$	$b \in B$	$\frac{1}{2}$ or 1	$\frac{1}{2}$ or 1
$a \in B$	$b \in \bar{B}$	1	0
$a \in \bar{B}$	$b \in B$	0	1
$a \in \bar{B}$	$b \in \bar{B}$	1	1

weight of a set B is defined as

$$w^2_v(B) = 2^{|\max_v(B)|}.$$

The weight of B estimates how far the algorithm is from a solution, assuming that the elements in B are the m smallest elements. The weight of a node v is now defined to be

$$w(v) = \sum_{A \in \mathcal{A}(v)} w^1_v(A) + 2^{4\sqrt{n}} \sum_{B \in \mathcal{B}(v)} w^2_v(B).$$

In the beginning of an algorithm (in the upper part of the decision tree), the weight of a node is still the sum of the weights of all A's, and therefore $w(r) = 2^{2n+o(n)}$.

We now define $\mathcal{A}(v)$ and $\mathcal{B}(v)$ for the rest of T more exactly. For any node v in T, except the root, simply copy $\mathcal{A}(v)$ and $\mathcal{B}(v)$ from the parent node and remove all sets that are not upper or lower half compatible with v, respectively. We ensure that the weight of every leaf is small by doing the following: If, for some $A \in \mathcal{A}(v)$ we have $|\min_v(A)| = \lceil 2\sqrt{n} \rceil$, we select an element $a_0 \in \min_v(A)$ which has been compared to the fewest number of elements in $\bar{A}$; we then remove the set A from $\mathcal{A}(v)$ and add the set $B = \bar{A} \cup \{a_0\}$ to $\mathcal{B}(v)$.

Note that at the root, $|\min_r(A)| = m$ for all $A \in \mathcal{A}(r)$ and that this quantity decreases by at most one for each comparison until a leaf is reached. In a leaf v the median is known; thus, $\mathcal{A}(v)$ is empty.

LEMMA 2.2. *Let $\mathcal{A}(v)$ and $\mathcal{B}(v)$ be defined by the rules described above. Then, every internal node v (labeled $a : b$) in T has a child with at least half the weight of v, i.e., $w(v_{a<b}) \geq w(v)/2$ or $w(v_{a>b}) \geq w(v)/2$.*

Proof. Table 2.1 gives the weights of a set $A \in \mathcal{A}(v)$ at v's children relative to the weight $w^1_v(A)$ of A at v. Similarly, Table 2.2 gives the weights of a set $B \in \mathcal{B}(v)$ in v's children, relative to the weight $w^2_v(v)$ of B at v. As $w^1_{v_{a<b}}(A) + w^1_{v_{a>b}}(A) \geq w^1_v(A)$ and $w^2_{v_{a<b}}(B) + w^2_{v_{a>b}}(B) \geq w^2_v(B)$, for every $A \in \mathcal{A}(v)$ and $B \in \mathcal{B}(v)$, all that remains to be checked is that the weight does not decrease when a lower half compatible set B replaces an upper half compatible set A. This is covered by Lemma 2.3. □

LEMMA 2.3. *If A is removed from $\mathcal{A}(v)$ and B is added in its place to $\mathcal{B}(v)$, and if fewer than $4n$ comparisons have been performed on the path from the root to v, then $2^{4\sqrt{n}} w^2_v(B) > w^1_v(A)$.*

Proof. A set $A \in \mathcal{A}(v)$ is replaced by a set $B = \bar{A} \cup \{a_0\} \in \mathcal{B}(v)$ only when $|\min_v(A)| = \lceil 2\sqrt{n} \rceil$. The element a_0, in such a case, is an element of $\min_v(A)$ that has been compared to the fewest number of elements in $\bar{A}$. If a_0 was compared to at least $2\sqrt{n}$ elements in $\bar{A}$, we get that each element of $\min_v(A)$ was compared to at

least $2\sqrt{n}$ elements in $\bar{A}$, and at least $4n$ comparisons have been performed on the path from the root to v, a contradiction. We therefore get that a_0 was compared to fewer than $2\sqrt{n}$ elements of $\bar{A}$ and thus $|\max_v(B)| > |\max_v(\bar{A})| - 2\sqrt{n}$. As a consequence, we get that $4\sqrt{n} + |\max_v(B)| > |\min_v(A)| + |\max_v(\bar{A})|$ and thus $2^{4\sqrt{n}} w_v^2(B) > w_v^1(A)$, as required. □

We now know that the weight of the root is large and that the weight does not decrease too fast; what remains to be shown is that the weights of the leaves are relatively small. This is established in the following lemma.

LEMMA 2.4. *For a leaf v (in which the median is known), $w(v) \leq 2m2^{4\sqrt{n}}$.*

Proof. Clearly, the only sets compatible with a leaf of T are the set A containing the m largest elements and the set B containing the m smallest elements. Since $|\min_v(A)| = |\max_v(B)| = 1$, we get that $w_v^2(B) = 2$ and $A \notin \mathcal{A}(v)$.

Since there are exactly m elements that can be removed from B to obtain a corresponding $\bar{A}$, there can be at most m copies of B in $\mathcal{B}(v)$. □

Let T be a comparison tree that corresponds to a median finding algorithm. If the height of T is at least $4n$, we are done. Otherwise, by starting at the root and repeatedly descending to a child whose weight is at least half the weight of its parent, we trace a path whose length is at least $2n + o(n)$ and Theorem 2.1 follows. □

Let us see how the current formalism gives room for improvement that did not exist in the original proof. The $2n + o(n)$ lower bound is obtained by showing that each node v in a decision tree T that corresponds to a median finding algorithm has a child whose weight is at least half the weight of v. Consider the nodes $v_0, v_1, \ldots, v_\ell$ along the path obtained by starting at the root of T and repeatedly descending to the child with the larger weight until a leaf is reached. If we could show that sufficiently many nodes on this path have weights strictly larger than half the weights of their parents, we would obtain an improved lower bound for median selection. If $w(v_i) \geq \frac{1}{2}(1 + \delta_i) \cdot w(v_{i-1})$, for every $1 \leq i \leq \ell$, then the length of this path, and therefore the depth of T, is at least $2n + \sum_{i=1}^{\ell} \log_2(1 + \delta_i) + o(n)$.

3. An improved lower bound for pair-forming algorithms. Let v be a node of a comparison tree. An element x is a *singleton* at v if it was not compared above v with any other element. Two elements x and y form a *pair* at v if the elements x and y were compared to each other above v, but neither of them was compared to any other element.

A pair-forming algorithm is an algorithm that starts by constructing $\lfloor n/2 \rfloor = m - 1$ pairs. By concentrating on comparisons that involve elements that are part of pairs, we obtain a better lower bound for pair-forming algorithms.

THEOREM 3.1. *A pair-forming algorithm for finding the median must perform, in the worst case, at least $2.00691n + o(n)$ comparisons.*

Proof. It is easy to see that a comparison involving two singletons can be delayed until just before one of them is to be compared for the second time. We can therefore restrict our attention to comparison trees in which the partial order corresponding to each node contains at most two pairs. Allowing only one pair is not enough, as algorithms should be allowed to construct two pairs $\{a, b\}$ and $\{a', b'\}$ and then compare an element from $\{a, b\}$ with an element from $\{a', b'\}$.

We focus our attention on nodes in the decision tree in which an element of a pair is compared for the second time and in which the number of nonsingletons is at most ϵm for some $\epsilon < 1$. If v is a node in which the number of nonsingletons is at most ϵm, for some $\epsilon < 1$, then $\mathcal{B}(v)$ is empty and thus $w(v) = \sum_{A \in \mathcal{A}(v)} w_v^1(A)$ and we do not have to consider Table 2.2 for the rest of the section.

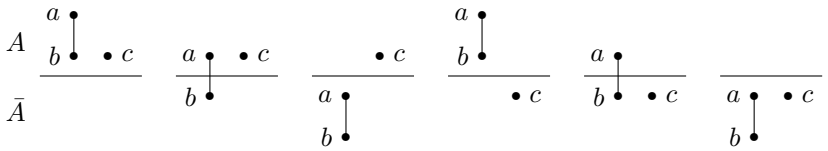


FIG. 3.1. *The six possible ways that a, b, and c may be divided between A and Ā. Note that c is not necessarily a singleton element; it may be part of a larger partial order.*

Recall that $\mathcal{A}(v)$ denotes the collection of subsets of U size m that are upper half compatible with v. If $H, L \subseteq U$ are subsets of U, of arbitrary size, we let

$$\mathcal{A}_{H/L}(v) = \{A \in \mathcal{A}(v) \mid H \subseteq A \text{ and } L \subseteq \bar{A}\}.$$

We let $w_{H/L}(v)$ be the contribution of the sets of $\mathcal{A}_{H/L}(v)$ to the weight of v, i.e.,

$$w_{H/L}(v) = \sum_{A \in \mathcal{A}_{H/L}(v)} w_v^1(A).$$

For brevity, we write $\mathcal{A}_{h_1 \ldots h_r/l_1 \ldots l_s}(v)$ for $\mathcal{A}_{\{h_1,\ldots,h_r\}/\{l_1,\ldots,l_s\}}(v)$ and $w_{h_1 \ldots h_r/l_1 \ldots l_s}(v)$ for $w_{\{h_1,\ldots,h_r\}/\{l_1,\ldots,l_s\}}(v)$.

Before proceeding, we describe the intuition that lies behind the rest of the proof. Consider Table 2.1 from the last section. If, in a node v of the decision tree, the two cases $a \in A$, $b \in \bar{A}$ and $a \in \bar{A}$, $b \in A$ are not equally likely, or more precisely, if the contributions $w_{a/b}(v)$ and $w_{b/a}(v)$ of these two cases to the total weight of v are not equal, there must be at least one child of v whose weight is greater than half the weight of v. The difficulty in improving the lower bound of Bent and John therefore lies at nodes in which the contributions of the two cases $a \in A$, $b \in \bar{A}$ and $a \in \bar{A}$, $b \in A$ are almost equal. This fact is not so easily seen when looking at the original proof given in [2].

Suppose now that v is a node in which an element a of a pair $\{a, b\}$ is compared with an arbitrary element c and that the number of nonsingletons in v is at most ϵm. We assume, without loss of generality, that $a > b$. The weights of a set $A \in \mathcal{A}(v)$ in v's children depend on which of the elements a, b, and c belongs to A and on whether c is minimal in A or maximal in $\bar{A}$. The six possible ways of dividing the elements a, b, and c between A and $\bar{A}$ are shown in Figure 3.1. The weights of the set A in v's children, relative to the weight $w_v^1(A)$ of A at v, in each one of these six cases are given in Table 3.1. Table 3.1 is similar to Table 2.1 with c playing the role of b. There is one important difference, however. If $a, b, c \in A$, as in the first row of Table 3.1, then the weight of A in $v_{a>c}$ is equal to the weight of A in v. The weight is not halved, as may be the case in the first row of Table 2.1. If the contribution $w_{abc/}(v)$ of the case $a, b, c \in A$ to the weight of v is not negligible, there must again be at least one child of v whose weight is greater than half the weight of v.

The improved lower bound is obtained by showing that if the contributions of the cases $a \in A$, $b \in \bar{A}$ and $a \in \bar{A}$, $b \in A$ are roughly equal, and if most elements in the partial order are singletons, then the contribution of the case $a, b, c \in A$ is nonnegligible. The larger the number of singletons in the partial order, the larger is the relative contribution of the weight $w_{abc/}(v)$ to the weight $w(v)$ of v. Thus, whenever an element of a pair is compared for the second time, we make a small gain. The above intuition is made precise in the following lemma.

TABLE 3.1
The weight of a set $A \in \mathcal{A}(v)$ in the children of a node v, relative to its weight in v, when the element a of a pair $a > b$ is compared with an arbitrary element c.

case			$w^1_{v_{a<c}}(A)$	$w^1_{v_{a>c}}(A)$
$a \in A$	$b \in A$	$c \in A$	$\frac{1}{2}$ or 1	1
$a \in A$	$b \in \bar{A}$	$c \in A$	$\frac{1}{2}$ or 1	$\frac{1}{2}$
$a \in \bar{A}$	$b \in \bar{A}$	$c \in A$	1	0
$a \in A$	$b \in A$	$c \in \bar{A}$	0	1
$a \in A$	$b \in \bar{A}$	$c \in \bar{A}$	0	1
$a \in \bar{A}$	$b \in \bar{A}$	$c \in \bar{A}$	$\frac{1}{2}$	$\frac{1}{2}$ or 1

LEMMA 3.2. *If v is a node in which an element a of a pair $a > b$ is compared with an element c, and if the number of singletons in v is at least $m + 2\sqrt{n}$, then*

$$w(v_{a<c}) \geq \tfrac{1}{2}w(v) + \tfrac{1}{2}(w_{c/a}(v) - w_{a/c}(v)) ,$$
$$w(v_{a>c}) \geq \tfrac{1}{2}w(v) + \tfrac{1}{2}(w_{a/c}(v) - w_{c/a}(v) + w_{abc/}(v)) .$$

Proof. Both inequalities follow easily by considering the entries in Table 3.1. To obtain the second inequality, for example, note that $w(v_{a>c}) \geq \frac{1}{2}(w(v) + w_{abc/}(v) - w_{c/ab}(v) + w_{ab/c}(v) + w_{a/bc}(v))$. As $w_{c/ab}(v) = w_{c/a}(v)$ and $w_{ab/c}(v) + w_{a/bc}(v) = w_{a/c}(v)$, the second inequality follows. □

It is worth pointing out that in Table 3.1 and in Lemma 3.2, we need only to assume that $a > b$; we do not use the stronger condition that $a > b$ is a pair. This stronger condition is crucial, however, in what follows, especially in Lemma 3.4.

To make use of Lemma 3.2 we need bounds on the relative contributions of the different cases. The following lemma is a useful tool for determining such bounds.

LEMMA 3.3. *Let $G = (V_1, V_2, E)$ be a bipartite graph. Let δ_1 and δ_2 be the minimal degree of the vertices of V_1 and V_2, respectively. Let Δ_1 and Δ_2 be the maximal degree of the vertices of V_1 and V_2, respectively. Assume that a positive weight function w is defined on the vertices of G such that $w(v_1) = r \cdot w(v_2)$ whenever $v_1 \in V_1$, $v_2 \in V_2$, and $(v_1, v_2) \in E$. Let $w(V_1) = \sum_{v_1 \in V_1} w(v_1)$ and $w(V_2) = \sum_{v_2 \in V_2} w(v_2)$. Then,*

$$r\frac{\delta_2}{\Delta_1} \cdot w(V_2) \;\leq\; w(V_1) \;\leq\; r\frac{\Delta_2}{\delta_1} \cdot w(V_2).$$

Proof. Let $v_1(e) \in V_1$ and $v_2(e) \in V_2$ denote the two vertices connected by the edge e. We then have

$$\delta_1 \sum_{v_1 \in V_1} w(v_1) \leq \sum_{e \in E} w(v_1(e)) = r \sum_{e \in E} w(v_2(e)) \leq r\Delta_2 \sum_{v_2 \in V_2} w(v_2).$$

The other inequality follows by exchanging the roles of V_1 and V_2. □

Using Lemma 3.3 we obtain the following basic inequalities.

LEMMA 3.4. *If v is a node in which $a > b$ is a pair and the number of nonsingletons in v is at most ϵm, then*

$$\begin{array}{rcccl}
\frac{1}{2}(1-\epsilon)\cdot w_{ac/b}(v) & \leq & w_{abc/}(v) & \leq & \frac{1}{2(1-\epsilon)}\cdot w_{ac/b}(v) \ , \\
2(1-\epsilon)\cdot w_{c/ab}(v) & \leq & w_{ac/b}(v) & \leq & \frac{2}{1-\epsilon}\cdot w_{c/ab}(v) \ , \\
\frac{1}{2}(1-\epsilon)\cdot w_{a/bc}(v) & \leq & w_{ab/c}(v) & \leq & \frac{1}{2(1-\epsilon)}\cdot w_{a/bc}(v) \ , \\
2(1-\epsilon)\cdot w_{/abc}(v) & \leq & w_{a/bc}(v) & \leq & \frac{2}{1-\epsilon}\cdot w_{/abc}(v) \ .
\end{array}$$

Each one of these inequalities relates a weight, such as $w_{abc/}(v)$, to a weight, such as $w_{ac/b}(v)$, obtained by moving one of the elements of the pair $a > b$ from A to $\bar{A}$. In each inequality we "lose" a factor of $1-\epsilon$. When the elements a and b are joined together, a factor of 2 is introduced. When the elements a and b are separated, a factor of $\frac{1}{2}$ is introduced.

Proof. We present a proof of the inequality $w_{abc/}(v) \leq \frac{1}{2(1-\epsilon)}\cdot w_{ac/b}(v)$. The proof of all the other inequalities is almost identical.

Construct a bipartite graph $G = (V_1, V_2, E)$ whose vertex sets are $V_1 = \mathcal{A}_{abc/}(v)$ and $V_2 = \mathcal{A}_{ac/b}(v)$. Define an edge $(A_1, A_2) \in E$ between $A_1 \in \mathcal{A}_{abc/}(v)$ and $A_2 \in \mathcal{A}_{ac/b}(v)$ if and only if there is a singleton $d \in \bar{A}_1$ such that $A_2 = A_1 \setminus \{b\} \cup \{d\}$. Suppose that (A_1, A_2) is such an edge. As $a \notin \min_v(A_1)$ but $a \in \min_v(A_2)$, while all other elements are extremal with respect to A_1 if and only if they are extremal with respect to A_2 (note that $b \in \min_v(A_1)$ and $b \in \max_v(\bar{A}_2)$), we get that $w^1_v(A_1) = \frac{1}{2}\cdot w^1_v(A_2)$.

For every set A of size m, the number of singletons in A is at least $(1-\epsilon)m$ and at most m. We therefore get that the minimal degrees of the vertices of V_1 and V_2 are $\delta_1, \delta_2 \geq (1-\epsilon)m$ and the maximal degrees of V_1 and V_2 are $\Delta_1, \Delta_2 \leq m$. The inequality $w_{abc/}(v) \leq \frac{1}{2(1-\epsilon)}\cdot w_{ac/b}(v)$ therefore follows from Lemma 3.3. □

Using these basic inequalities we obtain the following lemma.

LEMMA 3.5. *If v is a node in which $a > b$ is a pair and the number of nonsingletons is at most ϵm, for some $\epsilon < 1$, then*

$$\begin{aligned}
w_{abc/}(v) &\geq \tfrac{(1-\epsilon)^2}{(2-\epsilon)^2}\cdot w_{c/}(v) \ , \\
w_{a/c}(v) &\geq \tfrac{(1-\epsilon)(3-\epsilon)}{(2-\epsilon)^2}\cdot w_{/c}(v) \ , \\
w_{c/a}(v) &\leq \tfrac{1}{(2-\epsilon)^2}\cdot w_{c/}(v) \ .
\end{aligned}$$

Proof. We present the proof of the first inequality. The proof of the other two inequalities is similar. Using inequalities from Lemma 3.4 we get that

$$\begin{aligned}
w_{c/}(v) &= w_{abc/}(v) + w_{ac/b}(v) + w_{c/ab}(v) \\
&\leq w_{abc/}(v) + \tfrac{2}{1-\epsilon}\cdot w_{abc/}(v) + \tfrac{1}{(1-\epsilon)^2}\cdot w_{abc/}(v) \\
&= \tfrac{(2-\epsilon)^2}{(1-\epsilon)^2}\cdot w_{abc/}(v)
\end{aligned}$$

and the first inequality follows. □

We are now ready to show that if v is a node in which an element of a pair is compared for the second time, then v has a child whose weight is greater than half the weight of v. Combining Lemmas 3.2 and 3.5, we get that

$$\begin{aligned}
\tfrac{1}{2}\cdot(w(v_{a<c}) + w(v_{a>c})) &\geq \tfrac{1}{2}\cdot w(v) + \tfrac{(1-\epsilon)^2}{4(2-\epsilon)^2}\cdot w_{c/}(v) \ , \\
w(v_{a>c}) &\geq \tfrac{1}{2}\cdot w(v) - \tfrac{\epsilon}{2(2-\epsilon)}\cdot w_{c/}(v) + \tfrac{(1-\epsilon)(3-\epsilon)}{2(2-\epsilon)^2}\cdot w_{/c}(v) \ .
\end{aligned}$$

Let $\alpha = w_{c/}(v)/w(v)$ and $1-\alpha = w_{/c}(v)/w(v)$. We get that

$$\begin{aligned}\tfrac{1}{2}\cdot(w(v_{a<c}) + w(v_{a>c})) &\geq \left(\tfrac{1}{2} + \tfrac{(1-\epsilon)^2}{4(2-\epsilon)^2}\alpha\right)\cdot w(v)\ ,\\ w(v_{a>c}) &\geq \left(\tfrac{1}{2} - \tfrac{\epsilon}{2(2-\epsilon)}\alpha + \tfrac{(1-\epsilon)(3-\epsilon)}{2(2-\epsilon)^2}(1-\alpha)\right)\cdot w(v)\\ &= \left(\tfrac{1}{2} - \tfrac{3-2\epsilon}{2(2-\epsilon)^2}\alpha + \tfrac{(1-\epsilon)(3-\epsilon)}{2(2-\epsilon)^2}\right)\cdot w(v)\ .\end{aligned}$$

As a consequence, we get that

$$\max\{w(v_{a<c}), w(v_{a>c})\} \geq \max\left\{\tfrac{1}{2} + \tfrac{(1-\epsilon)^2}{4(2-\epsilon)^2}\alpha,\ \tfrac{1}{2} - \tfrac{3-2\epsilon}{2(2-\epsilon)^2}\alpha + \tfrac{(1-\epsilon)(3-\epsilon)}{2(2-\epsilon)^2}\right\}\cdot w(v)\ .$$

The coefficient of $w(v)$, on the right-hand side, is minimized when the two expressions whose maximum is taken are equal. This happens when $\alpha = \frac{2(3-4\epsilon+\epsilon^2)}{7-6\epsilon+\epsilon^2}$. Plugging this value of α into the two expressions, we get that

$$\max\{w(v_{a<c}), w(v_{a>c})\} \geq \tfrac{1}{2}(1 + f_1(\epsilon))\cdot w(v)\ ,$$

where

$$f_1(\epsilon) = \frac{(3-\epsilon)(1-\epsilon)^3}{(2-\epsilon)^2(7-6\epsilon+\epsilon^2)}\ .$$

It is easy to check that $f_1(\epsilon) > 0$ for $\epsilon < 1$.

A *pair-forming* comparison is a comparison in which two singletons are compared to form a pair. A *pair-touching* comparison is a comparison in which an element of a pair is compared for the second time. In a pair-forming algorithm, the number of singletons is decreased only by pair-forming comparisons. Each pair-forming comparison decreases the number of singletons by exactly two. As explained above, pair-forming comparisons can always be delayed so that a pair-forming comparison $a : b$ is immediately followed by a comparison that touches the pair $\{a, b\}$, or by a pair-forming comparison $a' : b'$ and then by a comparison that touches both pairs $\{a, b\}$ and $\{a', b'\}$.

Consider again the path traced from the root by repeatedly descending to the child with the larger weight. As a consequence of the above discussion, we get that when the ith pair-touching comparison along this path is performed, the number of nonsingletons in the partial order is at most $4i$. It therefore follows from the remark made at the end of the previous section that the depth of the comparison tree corresponding to any pair-forming algorithm is at least

$$\begin{aligned}&2n + \sum_{i=1}^{m/4} \log_2\left(1 + f_1\left(\frac{4i}{m}\right)\right) + o(n)\\ &= 2n + \frac{n}{8}\cdot\int_0^1 \log_2(1 + f_1(t))dt + o(n) \ \approx\ 2.00691n + o(n)\ .\end{aligned}$$

This completes the proof of Theorem 3.1. □

The worst case in the proof above is obtained when the algorithm converts all the elements into *quartets*. A quartet is a partial order obtained by comparing elements contained in two disjoint pairs. In the proof above, we analyzed cases in which an element a of a pair $a > b$ is compared with an arbitrary element c. If the element c is

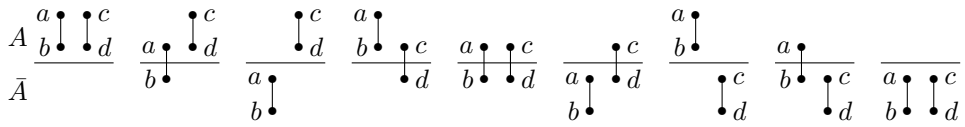


FIG. 3.2. *The nine possible ways that a, b, c, and d may be divided between A and Ā.*

TABLE 3.2

The weight of a set $A \in \mathcal{A}(v)$ in the children of a node v, relative to its weight in v, when the element a of a pair $a > b$ is compared with an element of a pair $c > d$.

case	$w^1_{v_{a<c}}(A)$	$w^1_{v_{a>c}}(A)$	$w^1_{v_{a<d}}(A)$	$w^1_{v_{a>d}}(A)$
$A \in \mathcal{A}_{abcd/}$	1	1	$\frac{1}{2}$	1
$A \in \mathcal{A}_{acd/b}$	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
$A \in \mathcal{A}_{cd/ab}$	1	0	1	0
$A \in \mathcal{A}_{abc/d}$	$\frac{1}{2}$	1	0	1
$A \in \mathcal{A}_{ac/bd}$	$\frac{1}{2}$	$\frac{1}{2}$	0	1
$A \in \mathcal{A}_{c/abd}$	1	0	$\frac{1}{2}$	$\frac{1}{2}$
$A \in \mathcal{A}_{ab/cd}$	0	1	0	1
$A \in \mathcal{A}_{a/bcd}$	0	1	0	1
$A \in \mathcal{A}_{/abcd}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1

also part of a pair, a tighter analysis is possible. By performing this analysis we can improve Theorem 3.1.

THEOREM 3.6. *A pair-forming algorithm for finding the median must perform, in the worst case, at least* $2.01227n + o(n)$ *comparisons.*

Proof. Consider comparisons in which the element from a pair $a > b$ is compared with an element of a pair $c > d$. The nine possible ways of dividing the elements a, b, c, and d among A and $\bar{A}$ are depicted in Figure 3.2. We may assume, without loss of generality, that the element a is compared with either c or with d.

Let v be a node of the comparison tree in which $a > b$ and $c > d$ are pairs and which one of the comparions $a : c$ or $a : d$ is performed. Let $A \in \mathcal{A}(v)$. The weights of a set A in v's children, relative to the weight $w^1_v(A)$ of A at v, in each one of these nine cases are given in Table 3.2. The two possible comparisons $a : c$ and $a : d$ are considered separately. The following equalities are easily verified.

LEMMA 3.7. *If $a > b$ and $c > d$ are pairs in v, then*

$$\begin{aligned} w_{acd/b}(v) &= w_{abc/d}(v) , \\ w_{cd/ab}(v) &= w_{ab/cd}(v) , \\ w_{c/abd}(v) &= w_{a/bcd}(v) , \\ w_{ac/bd}(v) &= 4 \cdot w_{ab/cd}(v) . \end{aligned}$$

The following inequalities are analogous to the inequalities of Lemma 3.4.

LEMMA 3.8. *If $a > b$ and $c > d$ are pairs in v and if the number of nonsingletons in v is at most ϵm, for some $\epsilon < 1$, then*

$$\begin{array}{rcccl}
\frac{1}{2}(1-\epsilon)w_{abc/d}(v) & \leq & w_{abcd/}(v) & \leq & \frac{1}{2(1-\epsilon)}w_{abc/d}(v)\ , \\
2(1-\epsilon)w_{ab/cd}(v) & \leq & w_{abc/d}(v) & \leq & \frac{2}{1-\epsilon}w_{ab/cd}(v)\ , \\
\frac{1}{2}(1-\epsilon)w_{a/bcd}(v) & \leq & w_{ab/cd}(v) & \leq & \frac{1}{2(1-\epsilon)}w_{a/bcd}(v)\ , \\
2(1-\epsilon)w_{/abcd}(v) & \leq & w_{a/bcd}(v) & \leq & \frac{2}{1-\epsilon}w_{/abcd}(v)\ .
\end{array}$$

First consider the comparison $a : c$. By examining Table 3.2 and using the equalities of Lemma 3.7, we get that

$$\begin{array}{rcl}
\frac{w(v_{a<c})+w(v_{a>c})}{2} = & & w_{abcd/}(v) + \frac{3}{4}w_{acd/b}(v) + \frac{1}{2}w_{cd/ab}(v) + \frac{3}{4}w_{abc/d}(v) + \frac{1}{2}w_{ac/bd}(v) \\
& & \quad + \frac{1}{2}w_{c/abd}(v) + \frac{1}{2}w_{ab/cd}(v) + \frac{1}{2}w_{a/bcd}(v) + \frac{1}{2}w_{/abcd}(v) \\
& = & w_{abcd/}(v) + \frac{3}{2}w_{abc/d}(v) + 3w_{ab/cd}(v) + w_{a/bcd}(v) + \frac{1}{2}w_{/abcd}(v).
\end{array}$$

Minimizing this expression, subject to the equalities of Lemma 3.7, the inequalities of Lemma 3.8, and the fact that the weights of the nine cases sum up to $w(v)$, amounts to solving a linear program. By solving this linear program we get that

$$\frac{w(v_{a<c}) + w(v_{a>c})}{2w(v)} \geq \frac{1}{2}(1 + f_2(\epsilon)) \cdot w(v)\ ,$$

where

$$f_2(\epsilon) = \frac{(3-\epsilon)(1-\epsilon)^3}{(2-\epsilon)^4}\ .$$

It seems intuitively clear that the comparison $a : d$ is a bad comparison from the algorithm's point of view. The adversary will most likely answer with $a > d$. Indeed, by solving the corresponding linear program, we get that

$$\begin{array}{rcl}
w(v_{a>d}) & = & w_{abcd/}(v) + \frac{1}{2}w_{acd/b}(v) + w_{abc/d}(v) + w_{ac/bd}(v) \\
& & \quad + \frac{1}{2}w_{c/abd}(v) + w_{ab/cd}(v) + w_{a/bcd}(v) + w_{/abcd}(v) \\
& = & w_{abcd/}(v) + \frac{3}{2}w_{abc/d}(v) + 5w_{ab/cd}(v) + \frac{3}{2}w_{a/bcd}(v) + w_{/abcd}(v) \geq \frac{3}{4}\ .
\end{array}$$

As $\frac{1}{2}(1 + f_2(\epsilon)) \leq \frac{3}{4}$, for every $0 \leq \epsilon \leq 1$, we may disregard the comparison $a : d$ from any further consideration.

It is easy to verify that $(1 + f_1(\epsilon))^2 \geq 1 + f_2(\epsilon)$. As a result, we get a lower bound of

$$2n + \frac{n}{8} \cdot \int_0^1 \log_2(1 + f_2(t))dt + o(n) \ \approx\ 2.01227n + o(n)\ .$$

This completes the proof of Theorem 3.6. □

4. Concluding remarks. We presented a reformulation of the $2n + o(n)$ lower bound of Bent and John for the number of comparisons needed for selecting the median of n elements. Using this new formulation we obtained an improved lower bound for pair-forming median finding algorithms. As mentioned, Dor and Zwick [6] have recently extended the ideas described here and obtained a $(2+\epsilon)n$ lower bound for general median finding algorithms for some tiny $\epsilon > 0$.

We believe that the lower bound for pair-forming algorithms obtained here can be substantially improved. Such an improvement seems to require, however, some new ideas. Obtaining an improved lower bound for pair-forming algorithms may be an important step towards obtaining a lower bound for general algorithms which is significantly better than the lower bound of Bent and John [2].

Paterson [11] conjectures that the number of comparisons required for selecting the median is about $(\log_{4/3} 2)\cdot n \approx 2.41n$.

REFERENCES

[1] M. Aigner, *Producing posets*, Discrete Math., 35 (1981), pp. 1–15.

[2] S. W. Bent and J. W. John, *Finding the median requires* $2n$ *comparisons*, in Proceedings of the 17th Annual ACM Symposium on Theory of Computing, 1985, pp. 213–216.

[3] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan, *Time bounds for selection*, J. Comput. System Sci., 7 (1973), pp. 448–461.

[4] J. Chen, *Partial Order Productions*, Ph.D. thesis, Lund University, Lund, Sweden, 1993.

[5] D. Dor and U. Zwick, *Selecting the median*, SIAM J. Comput., 28 (1999), pp. 1722–1758.

[6] D. Dor and U. Zwick, *Median selection requires* $(2+\epsilon)n$ *comparisons*, SIAM J. Discrete Math., 14 (2001), pp. 312–325.

[7] L. R. Ford and S. M. Johnson, *A tournament problem*, Amer. Math. Monthly, 66 (1959), pp. 387–389.

[8] F. Fussenegger and H. N. Gabow, *A counting approach to lower bounds for selection problems*, J. Assoc. Comput. Mach., 26 (1979), pp. 227–238.

[9] D. E. Knuth, *The Art of Computer Programming, Vol.* 3*, Searching and Sorting*, Addison-Wesley, Reading, MA, 1973.

[10] I. Munro and P.V. Poblete, *A Lower Bound for Determining the Median*, Technical report CS-82-21, University of Waterloo, Waterloo, Canada, 1982.

[11] M. S. Paterson, *Progress in selection*, in Proceedings of the Fifth Scandinavian Workshop on Algorithm Theory, Reykjavík, Iceland, 1996, pp. 368–379.

[12] V. R. Pratt and F. F. Yao, *On lower bounds for computing the ith largest element*, in Proceedings of the 14th Annual Symposium on Switching and Automata Theory, Iowa City, IA, 1973, pp. 70–81.

[13] A. Schönhage, M. Paterson, and N. Pippenger, *Finding the median*, J. Comput. System Sci., 13 (1976), pp. 184–199.

[14] C. K. Yap, *New Lower Bounds for Medians and Related Problems*, Computer science report 79, Yale University, New Haven, CT, 1976.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 312–325

MEDIAN SELECTION REQUIRES $(2+\epsilon)N$ COMPARISONS*

DORIT DOR† AND URI ZWICK†

Abstract. Improving a long standing result of Bent and John [*Proceedings of the* 17*th Annual ACM Symposium on Theory of Computing*, Providence, RI, 1985, pp. 213–216], and extending a recent result of Dor, Håstad, Ulfberg, and Zwick [*SIAM J. Discrete Math.*, 14 (2001), pp. 299–311], we obtain a $(2+\epsilon)n$ lower bound (for some fixed $\epsilon > 0$) on the number of comparisons required, in the worst case, for selecting the *median* of n elements.

Key words. median selection, comparison algorithms, lower bounds

AMS subject classifications. 68Q25, 68R05, 06A07

PII. S0895480199353895

1. Introduction.

The *selection problem* is defined as follows: Given a set X containing n distinct elements and given a number $1 \leq t \leq n$, find the tth largest element of X, i.e., the element of X smaller than exactly $t-1$ elements of X and larger than the other $n-t$ elements of X. The *median* of X is the $\lceil n/2 \rceil$th largest element of X.

The selection problem is one of the most fundamental problems of computer science. Although considerable effort was made to determine the exact comparison complexity of it, the problem is still far from being solved. The history of the upper and lower bounds obtained for the problem of median selection is summarized in Table 1.1.

Blum et al. [BFP+73] obtained the first linear upper bound and the first nontrivial lower bound for the problem. Schönhage, Paterson, and Pippenger [SPP76] improved the upper bound to $3n$. Extending the work of Schönhage, Paterson, and Pippenger[SPP76], we recently lowered the upper bound to about $2.95n$ [DZ99].

The lower bound of Blum et al. [BFP+73] is obtained using a simple adversary argument. Progressively more elaborate adversary arguments were used by Pratt and Yao [PY73], Kirkpatrick [Kir81], Yap [Yap76], and Munro and Poblete [MP82] to push the lower bound to $79n/43 \sim 1.837n$. To obtain the last lower bound in this list, hundreds of cases had to be checked. Therefore, it seems that the type of adversary arguments used in these works is unlikely to yield substantially better results.

A different approach altogether was used by Fussenegger and Gabow [FG78]. Although their lower bound seems to be identical to the lower bound of Blum et al. [BFP+73], their result is in fact stronger. Fussenegger and Gabow show that any comparison tree for finding the median must have at least $2^{1.5n-o(n)}$ leaves. As comparison trees are binary trees (we assume that all the elements are distinct), it follows that each median finding comparison tree must have a leaf at depth at least $1.5n - o(n)$. Fussenegger and Gabow also present an alternative formulation of their lower bound using an adversary based on a weight function which they call *chaos*.

Bent and John [BJ85] (see also John [Joh88]) extended the work of Fussenegger and Gabow and showed that every comparison tree for finding the median of n ele-

*Received by the editors April 12, 1999; accepted for publication January 30, 2001; published electronically June 5, 2001.

http://www.siam.org/journals/sidma/14-3/35389.html

†School of Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel (dorit@checkpoint.com, zwick@post.tau.ac.il).

TABLE 1.1

Upper and lower bounds on the number of comparisons required, in the worst case, for selecting the median of n elements.

UPPER BOUNDS

AUTHORS	REF.	BOUND
Blum, Floyd, Pratt, Rivest, Tarjan	[BFP+73]	$5.43 \cdot n$
Schönhage, Paterson, Pippenger	[SPP76]	$3 \cdot n$
Dor, Zwick	[DZ99]	$2.95 \cdot n$

LOWER BOUNDS

AUTHORS	REF.	BOUND
Blum, Floyd, Pratt, Rivest, Tarjan	[BFP+73]	$1.5 \cdot n$
Fusseneger, Gabow	[FG78]	$1.5 \cdot n$
Pratt, Yao	[PY73]	$1.75 \cdot n$
Kirkpatrick	[Kir81]	$1.75 \cdot n$
Yap	[Yap76]	$1.81 \cdot n$
Munro, Poblete	[MP82]	$1.83 \cdot n$
Bent, John	[BJ85]	$2 \cdot n$
Dor, Zwick	*	$(2+\epsilon) \cdot n$

ments must either have a depth which is much larger than $2n$ or must have at least $2^{2n-o(n)}$ leaves. As a consequence, each such comparison tree must have a leaf at depth at least $2n - o(n)$. Descriptions of this lower bound can also be found in Paterson [Pat96] (using a *multitude of adversaries*), in Knuth [Knu98] (using a *randomized adversary*) and in Dor et al. [DHUZ01] (using a *weight function*).

Dor et al. [DHUZ01] recently showed that any *pair-forming* median selecting algorithm must perform, in the worst-case, at least $2.01n$ comparisons. A pair-forming algorithm is an algorithm that begins with $\lfloor n/2 \rfloor$ comparisons of $\lfloor n/2 \rfloor$ disjoint pairs of elements. Although the best current selection algorithms are pair-forming, it is not yet known whether there is an optimal median selection algorithm which is pair-forming.

In this work, we extend the result of Dor et al. [DHUZ01] and show that *any* median selection algorithm must perform, in the worst-case, at least $(2 + \epsilon)n$ comparisons for some fixed $\epsilon > 0$. Unfortunately, the ϵ for which we are currently able to prove this result is tiny. We present here a proof of this result with $\epsilon = 2^{-80}$. No attempt was made to optimize ϵ as it seems that new ideas will be required to obtain a significant improvement. We believe that the significance of the new lower bound is in showing that median selection requires *more than* $2n + o(n)$ comparisons.

Preliminary versions of this work appears in the Ph.D. thesis of the first author [Dor95] and in [DZ96].

2. Discussion. Median selection is one of the most fundamental problems of computer science and it is important, therefore, to determine its exact complexity.

The comparison complexity of many comparison problems is exactly known. It is clear, for example, that exactly $n - 1$ comparisons are needed to find the maximum of n elements. Exactly $n + \lceil \log n \rceil - 2$ comparisons are needed, in the worst case, to find the second largest element (Schreier [Sch32], Kislitsyn [Kis64]). All logarithms in the paper are taken to base 2. Exactly $n + \lceil \log \frac{n-1}{4} \rceil + \lceil \log \frac{n-1}{5} \rceil + 1$ comparisons are needed to find the third largest element of n elements when $n \geq 50$ (Kirkpatrick [Kir74]). Exactly $\lceil 3n/2 \rceil - 2$ comparisons are needed to find both the maximum and

the minimum of n elements (Pohl [Poh72]). Exactly $2n-1$ comparisons are needed to merge two sorted lists each of length n (Stockmeyer and Yao [SY80]). Exactly $\lceil \log \frac{7}{12}(n+1) \rceil + \lceil \log \frac{14}{17}(n+1) \rceil$ comparisons are needed to merge a sorted list of n elements with a sorted list of two elements (Hwang and Lin [HL71]).

The comparison complexity of sorting is almost exactly known. At least $\lceil \log n! \rceil \geq n \log n - 1.44n$ comparisons are needed. At most $\sum_{k=2}^{n} \lceil \log \frac{3}{4}k \rceil \leq n \log n - 1.33n$ comparisons are needed, as shown by Ford and Johnson [FJ59] (see also Knuth [Knu98]).

It is even known that the average case complexity of selecting the median is $1.5n + o(n)$. The upper bound is by Floyd and Rivest [FR75]. The lower bound is by Cunto and Munro [CM89].

Determining the exact number of comparisons needed, in the worst case, for selecting the median seems to be a much harder and a much more challenging problem. A gap of about $0.95n$ still remains between the best upper and lower bounds. We know much more about the second order term of sorting than we know about the leading term of median selection. We believe that even small steps towards closing the gap are significant if they are obtained using new ideas or break what seems to be a natural boundary.

Paterson [Pat96] conjectures that the number of comparisons required for selecting the median is about $(\log_{4/3} 2) \cdot n \approx 2.41n$.

3. Comparison trees. The problem of finding the median can be formulated as a game between an *algorithm* and an *adversary*. The algorithm is supposed to find the median of the n distinct input elements. The algorithm is only allowed to issue comparison queries of the form $a : b$, where a and b are input elements. The adversary has to answer each such comparison with either $a < b$ or $a > b$. Each answer of the adversary has to be consistent with all her previous answers. The algorithm tries to find the median using as few comparisons as possible. The adversary tries to force the algorithm to make as many comparisons as possible.

A median finding algorithm in this model can be described using a *comparison tree*. A comparison tree T is a binary tree in which each internal vertex v is labeled by a comparison $a : b$. The two edges emanating from v are labeled by the two possible outcomes $a < b$ and $a > b$ of the comparison. The vertices to which these edges lead are denoted, respectively, by $v_{a<b}$ and $v_{a>b}$. The algorithm starts at the root r of the tree and makes the comparison labeling it. The algorithm then proceeds along the edge labeled by the reply of the adversary, makes the comparison labeling the vertex reached, and so forth. The operation of the algorithm ends when a leaf is encountered. The information gathered by the algorithm when it reaches a vertex v of the tree amounts to a partial order P_v on the n input elements. A comparison tree T describes a median finding algorithm if for every leaf w of T, the partial order P_w contains enough information to determine the median. Throughout the paper, we assume that the algorithm never makes a comparison whose outcome is already known to it.

The worst case complexity of a median finding algorithm described by a comparison tree T is simply the *depth* $D(T)$ of T. The adversary, who knows T, can force the algorithm to take a path that leads to a leaf of maximal depth.

The number of leaves of a comparison tree T is denoted by $L(T)$. As each comparison tree is a binary tree, we immediately get that $L(T) \leq 2^{D(T)}$ or equivalently that $D(T) \geq \log_2 L(T)$. A lower bound on the number of leaves contained in every median finding comparison tree therefore supplies a lower bound on the number or comparisons that have to be made, in the worst case, by any median finding algorithm.

This is the so-called *leaf counting* approach.

4. An overview of the new lower bound. The main idea used to obtain the improved lower bound is fairly simple. Let T be a comparison tree that corresponds to a median finding algorithm. To each vertex v of T we attach a weight $w(v)$. We then show that the weights attached to the vertices satisfy the following three properties:

P1. $w(\mathit{root}) \geq 2^{2n-o(n)}$.
P2. $w(\mathit{leaf}) \leq 2^{o(n)}$ for every leaf *leaf*.
P3. $w(v_{a>b}) + w(v_{a<b}) \geq w(v)$ for every internal vertex v.

Property P3 implies that each internal vertex v has a child u such that $w(u) \geq \frac{1}{2}w(v)$. By starting at the root of the tree and descending each time to the child with the larger weight, until a leaf is reached, we trace a path whose length is at least $\log_2(w(\mathit{root})/w(\mathit{leaf})) = 2n - o(n)$. This gives us an alternative formulation of the lower bound of Bent and John [BJ85].

To get an improved lower bound we have to go one step further. Let $v_0, v_1, \ldots, v_k$ be the path traced above. Let δ_i be such that $w(v_i) = \frac{1}{2}(1+\delta_i)\cdot w(v_{i-1})$. The length of the path is then guaranteed to be at least $2n + \sum_{i=1}^{k} \log_2(1+\delta_i) - o(n)$. If we could show that in enough cases the weight of a child is strictly greater than half the weight of its parent, so that $\sum_{i=1}^{k} \log_2(1+\delta_i) = \Omega(n)$, we would get our improved lower bound.

Dor et al. [DHUZ01] obtain an improved lower bound for pair-forming median algorithms using this approach. Obtaining an improved lower bound for general median finding algorithms seems to be a harder task. To achieve it, we have to bend the rules of the game a little bit. The weight function used to obtain the lower bound violates property P3, so that $\delta_i < 0$ for some values of i. We can still show, however, that $\sum_{i=1}^{k} \log_2(1+\delta_i) = \Omega(n)$, which is the important property.

5. Weight functions. A suitable weight function is an essential ingredient in the plan laid down in the previous section. The weight function used to obtain the lower bound is a modification of the weight function used in [DHUZ01]. We shortly describe the weight function used in [DHUZ01] and then describe the weight function used by us here.

Let P be a partial order on the elements of a set X. Let $A \subset X$ be a subset of X. The set A is said to be an *up-set* of P if whenever $a <_P b$ and $a \in A$, then $b \in A$. In other words, A is an up-set of P if none of the elements of A is known to be smaller than an element from $\bar{A} = X \setminus A$ (see Trotter [Tro92, p. 27]). We let $\min_P(A)$ be the set of minimal elements, with respect to P, in A. We let $\max_P(\bar{A})$ be the set of maximal elements, with respect to P, in $\bar{A}$. If P_v is the partial order associated with a vertex v in T, we let $\min_v(A) = \min_{P_v}(A)$ and $\max_v(\bar{A}) = \max_{P_v}(\bar{A})$. We let $\mathit{up\text{-}set}_t(v)$ be the set of up-sets of P_v of size t.

To simplify the description of the weight functions, we assume that comparisons involving two singletons are always delayed for as long as possible, i.e., until just before one of these two singletons is to be compared for a second time. (A *singleton* is an element that was not yet compared to any other element.) It is easy to see that we can therefore restrict our attention to comparison trees in which the partial order corresponding to each vertex contains at most two pairs. (Two elements a and b form a *pair* if the comparison $a : b$ is the only comparison in which these two elements participated.) Allowing only one pair in each partial order is not enough as algorithms should be allowed to build two pairs $\{a, b\}$ and $\{a', b'\}$ and then compare an element from $\{a, b\}$ with an element from $\{a', b'\}$ to form a quartet.

If v is a vertex of a comparison tree in which the number of singletons is greater than $n/2+2\sqrt{n}$, then the definition of the weight function $w(v)$ of v used in [DHUZ01] is particularly simple. Let $t = \lceil n/2 \rceil$. The weight $w_A(v)$ of v with respect to a set $A \subset X$ of size t is defined as follows:

$$w_A(v) = \begin{cases} 2^{|\min_v(A)| + |\max_v(\bar{A})|} & \text{if } A \in \textit{up-set}_t(v)\ , \\ 0 & \text{otherwise.} \end{cases}$$

The weight of v is then defined to be

$$w(v) = \sum_A w_A(v),$$

where the sum is over all subsets A of X of size t.

If the number of singletons in v is at most $n/2 + 2\sqrt{n}$, then the definition of the weight function $w(v)$ used in [DHUZ01] is a bit more complicated. As we concentrate here on the early stages of the algorithm in which the number of singletons is still very large, we do not repeat this definition here. The interested reader may find it in [DHUZ01].

Before describing the weight function $\hat{w}(v)$ used by us here to obtain the new lower bound, we describe a slightly different way of looking at the weight function $w(v)$ used in Dor et al. [DHUZ01].

Let $v = a : b$ be a vertex of a median finding tree. The outcome $a < b$ is said to be *critical* with respect to an up-set A if either $a, b \in A$ and $b \in \min_v(A)$ or $a, b \in \bar{A}$ and $a \in \max_v(\bar{A})$. In other words, an outcome is critical if it removes an element from the set of minimal elements of A or the set of maximal elements of $\bar{A}$. Note that a critical outcome is an outcome of an *internal* comparison, i.e., a comparison between two elements of A, or two elements of $\bar{A}$. An outcome $a < b$ of an internal comparison which is not critical is said to be *noncritical*.

Let $\text{int}_v(A)$ be the number of internal comparisons with respect to A above v. Let $\textit{crit}_v(A)$ be the number of critical outcomes with respect to A above v and let $\textit{noncrit}_v(A)$ be the number of noncritical outcomes with respect to A above v. Clearly $\text{int}_v(A) = \textit{crit}_v(A) + \textit{noncrit}_v(A)$. The number of extremal elements with respect to A, i.e., $|\min_v(A)| + |\max_v(\bar{A})|$ is equal to $n - \textit{crit}_v(A)$. An alternative formulation of the weight function $w_A(v)$ is therefore

$$w_A(v) = \begin{cases} 2^{n - \textit{crit}_v(A)} & \text{if } A \in \textit{up-set}_t(v)\ , \\ 0 & \text{otherwise}\ . \end{cases}$$

So far, we classified the outcome of each internal comparison as either critical or noncritical. In each vertex v we now also classify the noncritical outcomes with respect to A obtained above v as either *marked* or *unmarked*. If an outcome $a < b$ is marked, with respect to A, at v, then it will also be marked at all the vertices that lie below v. Playing the role of the adversary, we may decide, however, to mark a noncritical outcome $a < b$ in v, with respect to A, although it was not marked at v's parent. Let $\textit{unmark}_v(A)$ be the number of noncritical outcomes with respect to A, obtained above v, which are still unmarked at v. The new weight function is defined as follows:

$$\hat{w}_A(v) = \begin{cases} 2^{n - \textit{crit}_v(A) - \textit{unmark}_v(A)} & \text{if } A \in \textit{up-set}_t(v)\ , \\ 0 & \text{otherwise.} \end{cases}$$

Another difference in the definition of the new weight functions is that the score $\hat{w}(v)$ is *not* obtained by summing over all up-sets A. Some of the up-sets are deliberately excluded from the summation. For each vertex v there is a set $drop(v)$ such that

$$\hat{w}(v) = \sum_{A \notin drop(v)} \hat{w}_A(v) \ .$$

If $A \in drop(v)$, so that A is excluded at v, then A is also excluded at every descendant of v.

The definition of $\hat{w}(v)$ given above is used if the number of singletons in v is at least $n/2 + 2\sqrt{n}$. If the number of singletons in v is less than $n/2 + 2\sqrt{n}$, we revert to the definition used in [DHUZ01] and take $\hat{w}(v) = w(v)$. It is easy to see that for every vertex v we have $\hat{w}(v) \leq w(v)$.

The definition of the weight function $\hat{w}(v)$ is not complete without specifying the rules used to mark noncritical outcomes and to exclude up-sets . These rules will be specified in the next section.

6. An improved lower bound for general median finding algorithms. In this section we obtain the improved lower bound for general median finding algorithms. We use the weight function $\hat{w}(v)$ defined in the previous section.

Let T be a median finding tree. We start again at the root at the tree and then repeatedly descend to the child with the larger weight. As we descend the tree, we decide which noncritical outcomes to mark and which up-sets to drop, thereby determining the weights of the vertices we encounter. No outcomes are marked and no up-sets are excluded at the root. Suppose that we are currently at vertex v. Let v_1 and v_2 be the two children of v. Outcomes that are marked at v must also be marked at v_1 and v_2 and up-sets that are excluded at v must also be excluded at v_1 and v_2. We may, however, decide to mark some additional noncritical outcomes, thus increasing the weight, or to exclude some up-sets, thus decreasing the weight. After making these decisions, we can compute the weights $\hat{w}(v_1)$ and $\hat{w}(v_2)$. We then descend to the child with the larger weight. We denote this child by v'.

Let $v_0, v_1, v_2, \ldots, v_k$ be the path traced from the root. We say that the move $v_i \to v_{i+1}$ is *ν-good* if $\hat{w}(v_{i+1}) \geq \frac{1}{2}(1+\nu)\hat{w}(v_i)$. We say that the move is *regular* if $\hat{w}(v_{i+1}) \geq \frac{1}{2}\hat{w}(v_i)$. We say that the move is *ν-bad* if $\hat{w}(v_{i+1}) \geq \frac{1}{2}(1-\nu)\hat{w}(v_i)$. Our goal is to show that the path traced contains many good moves and only a few bad ones.

We concentrate our attention on comparisons in which at least one of the elements is compared for the first or second time. The good and the bad moves correspond to such comparisons. An element becomes *active* when it is compared for the first time. If the first comparison of an element yields a good move, the element immediately becomes inactive. The second comparison involving an active element yields, in most cases, a good move. In some less frequent cases, however, such a comparison yields a bad move. In either case, the element becomes inactive. In some cases an element becomes inactive before it is compared for the second time. We say that two elements are *neighbors* if they were directly compared to each other. We will maintain the following two conditions:

1. An active element has exactly one neighbor.
2. An element has at most one active neighbor.

Each comparison $a : b$ falls into one of the following four types:

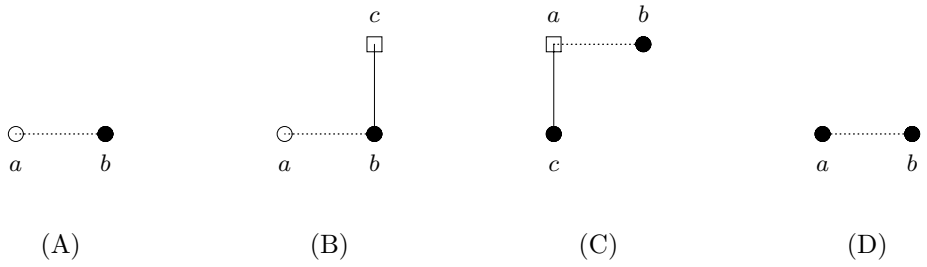


FIG. 6.1. *Comparisons of Types A,B,C, and D. Empty circles denote singletons. Squares denote active elements. The filled circles in Type A,B, and D comparisons denote elements that are neither singletons nor active. The dotted line denotes the comparison to be made.*

Type A. Comparison between a singleton and an element that does not have an active neighbor.

Type B. Comparison between a singleton and an element that has an active neighbor.

Type C. Comparison of an active element and another element.

Type D. Other comparison.

Comparisons of Type A and B are comparisons in which an element is compared for the first time. Comparisons of Type C are comparisons in which an active element is compared for the second time. The four types of comparisons are shown schematically in Figure 6.1. We now describe the actions taken by the adversary at a vertex v labeled by the comparison $a : b$. We consider four different cases, corresponding to the four types of comparisons. We let v_1 and v_2 denote the children of v, and we let v' denote the child of v with the largest weight (ties are broken arbitrarily). In the following we let $\nu = 2^{-70}$, $\Delta = 65$, and $\epsilon = 2^{-80}$.

Case A. a is a singleton. b is not active and does not have an active neighbor.

No new outcomes are marked; no new up-sets are dropped at v_1 and v_2. If $v \to v'$ is a ν-good move, then a immediately becomes inactive. Otherwise, if $v \to v'$ is not a ν-good move, then a becomes active. If b is also a singleton, then b also becomes active. The move $v \to v'$ is a regular move. (It is easy to see that if b is also a singleton, then the move $v \to v'$ is necessarily a regular move.)

Case B. a is a singleton. b is not active but has an active neighbor c.

Assume that $b < c$. The case $b > c$ is dealt with analogously. Mark the outcome $a > b$ at $v_{a>b}$ with respect to all the up-sets A, such that $a, b \in \bar{A}$, with respect to which the outcome $a > b$ is noncritical. We show below (Lemma 6.2) that $v \to v'$ is a ν-good move. The element a immediately becomes inactive. The element c becomes inactive. The element b remains without an active neighbor.

Case C. a is active and c is the unique neighbor of a.

Assume that $a > c$. The case $a < c$ is dealt with analogously. In $v_{a<b}$ and $v_{a>b}$, if $x > c$ is a noncritical outcome and A is an up-set such that $x, c \in \bar{A}$, then mark $x > c$ with respect to A. In $v_{a>b}$, if A is an up-set such that $a, b \in A$ and the outcome $a > b$ is noncritical with respect to A, then mark $a > b$ with respect to A. If the move $v \to v'$ is not ν-good, then we exclude all the up-sets A for which $c \in A$ at v_1 and v_2 (i.e., $\mathit{drop}(v_{a<b}) = \mathit{drop}(v_{a>b}) = \mathit{drop}(v) \cup \{A \mid c \in A\}$). We then recompute the weights $\hat{w}(v_1)$ and $\hat{w}(v_2)$. We show below (Lemma 6.3) that the degree of c in this case is at least Δ and that $\hat{w}(v_{a<b}) \geq \frac{1}{2}(1 - 3\nu) \cdot \hat{w}(v)$. The move $v \to v_{a<b}$ is

therefore 3ν-bad. In both cases, a becomes inactive. If b or c were active, then they also become inactive. If b was a singleton before the comparison $a : b$, then it becomes inactive.

Case D. The comparison $a : b$ is not of one of the types considered in Cases A, B, and C.

No new outcomes are marked; no new up-sets are dropped. The move $v \to v'$ is regular.

At most one element becomes inactive in Case A. Exactly two elements become inactive in Case B. At most three elements become inactive in Case C. It should be noted, however, that three elements become inactive in Case C only if a and c formed a pair in v. No elements become inactive in Case D.

Up-sets added to $drop(v)$ and therefore excluded from the definition of $\hat{w}(v)$ are always up-sets that include, or that do not include, a specific element. We say that an element a is *forced* in v if all the up-sets that include a, or all the up-sets that do not include a, belong to $drop(v)$.

It is important to observe that while marking noncritical outcomes in Cases B and C, we maintain the following property: if we mark an outcome $x < y$ with respect to an up-set A for which $x, y \in A$, then we deactivate the active neighbor of y, if there was one. Similarly, if we mark an outcome $x < y$ with respect to an up-set A for which $x, y \in \bar{A}$, then we deactivate the active neighbor of x, if there were one.

The new lower bound follows from the following three lemmas, dealing with Cases A, B, and C.

LEMMA 6.1. *If the comparison $a : b$ labeling v is of type A and b is forced at v, then $v \to v'$ is ν-good.*

LEMMA 6.2. *If the comparison $a : b$ labeling v is of type B, then $v \to v'$ is ν-good.*

LEMMA 6.3. *If the comparison $a : b$ labeling v is of type C and $v \to v'$ is not ν-good, then*

(i) *the degree of c, the neighbor of a, in P_v is at least Δ;*

(ii) *the move $v \to v_{a<b}$ is 3ν-bad.*

The (technical) proofs of these three lemmas will be given later. We first show that the lemmas imply the new lower bound.

THEOREM 6.4. *The depth of every median finding comparison tree is at least $(2+\epsilon)n - o(n)$, where $\epsilon \geq 2^{-80}$.*

Proof. The adversary traces a path from the root. Consider the number of comparisons performed until $N = n/4$ elements become involved in comparisons. If this number is at least $(2+4\epsilon)N$, then the depth of the tree must be at least $(2+\epsilon)n$. This is because the adversary may decide that $N/2$ of the compared elements are the largest elements in the set, and that the remaining $N/2$ elements are the smallest elements in the set, in which case the algorithm would have to find the median of the remaining $n - N$ elements. This requires at least $2(n - N) - o(n)$ more comparisons by the lower bound of Bent and John [BJ85]. The total number of comparisons performed would therefore be at least $(2 + 4\epsilon)N + 2(n - N) - o(n) = (2 + \epsilon)n - o(n)$.

Therefore, assume that the number of comparisons performed until $N = n/4$ elements are involved in comparisons is at most $(2 + 4\epsilon)N$. Let v^* be the first vertex on the path traced by the adversary in which N elements are involved in comparisons. The partial order P_{v^*} contains exactly $n - N$ singletons and N nonsingletons. The number of elements whose degree in P_{v^*} is at least Δ is at most $2(2 + 4\epsilon)N/\Delta$. Lemma 6.1 (Case A) implies that a forced element will never have an active neighbor. As each bad move forces an element, we get, using Lemma 6.3 (Case C), that the

number of bad moves is at most $2(2+4\epsilon)N/\Delta$. Each good or bad move causes at most three elements to become inactive. The number of good and bad moves is therefore at least a third of the number of elements that become inactive. As explained, we assume that the formation of pairs is delayed until just before the time they are used. We may therefore assume that each partial order P_v contains at most two pairs. Each element has at most one active neighbor. The number of active elements at v^* is therefore at most $(N-4)/2+4 = N/2+2$. It therefore follows that the number of good and bad moves is at least $(N/2-2)/3 \geq N/6-1$ and that the number of good moves is at least $N/6 - 2(2+4\epsilon)N/\Delta - 1$. As mentioned, a good or bad move in which three elements become inactive can result only from a comparison in which an element of a pair is compared for the second time. It is easy to show, using the arguments used in [DHUZ01], that the move that results from such a comparison is much better than two ν-good moves. We may therefore assume that the number of good and bad moves is at least $(N/2-2)/2 \geq N/4-1$ and that the number of good moves is at least $N/4 - 2(2+4\epsilon)N/\Delta - 1$. We get therefore that

$$\sum_{i=1}^{k} \log_2(1+\delta_i) \;\geq\; \Big(\frac{N}{4} - \frac{2(2+4\epsilon)N}{\Delta} - 1\Big)\log_2(1+\nu) \;+\; \frac{2(2+4\epsilon)N}{\Delta}\cdot\log_2(1-3\nu) \;\geq\; \epsilon n\;,$$

as required. □

Before proving Lemmas 6.1–6.3 we introduce the following notation, also used in [DHUZ01]. Given two sets of elements $H, L \subseteq X$, we let

$$\mathcal{A}_{H/L}(v) \;=\; \{A \in \textit{up-set}_t(v) \mid H \subseteq A, L \subseteq \bar{A}\}\;.$$

In other words, $\mathcal{A}_{H/L}(v)$ is the set of all up-sets A of size $t = \lceil n/2 \rceil$ at v such that all the elements of H belong to A while none of the elements of L belongs to A. We also let

$$\hat{w}_{H/L}(v) \;=\; \sum_{A \in \mathcal{A}_{H/L}(v) \setminus \textit{drop}(v)} \hat{w}_A(v)\;.$$

For brevity, we write $\hat{w}_{a/b}(v)$ and $\hat{w}_{ab/}(v)$, etc. instead of $\hat{w}_{\{a\}/\{b\}}(v)$ and $\hat{w}_{\{a,b\}/\phi}(v)$.

We also need the following lemma which is an adaptation of Lemmas 7 and 8 of [DHUZ01].

LEMMA 6.5. *If for every $A \in \mathcal{A}_{H/L,x}(v)$ and every singleton $y \in A \setminus H$ we have*

$$s \cdot \hat{w}_A(v) \;\leq\; \hat{w}_{A'}(v) \;\leq\; r \cdot \hat{w}_A(v)\;,$$

where $A' = A \cup \{x\} \setminus \{y\}$, and if the number of nonsingletons in v is at most $\frac{n}{4} - |H|$, then

$$\frac{s}{2} \cdot \hat{w}_{H/L,x}(v) \;\leq\; \hat{w}_{H,x/L}(v) \;\leq\; 2r \cdot \hat{w}_{H/L,x}(v)\;.$$

Proof. Consider the bipartite graph $G = (V_1, V_2, E)$, where $V_1 = \mathcal{A}_{H/L,x}(v)$, $V_2 = \mathcal{A}_{H,x/L}(v)$, and $(A, A') \in E$ if and only if $A \in V_1$, $A' \in V_2$, and there is a singleton $y \in A \setminus H$ such that $A' = A \cup \{x\} \setminus \{y\}$. For every $A \in V_1$, the number of singletons in $A \setminus H$ is at least $(\frac{n}{2} - |H|) - (\frac{n}{4} - |H|) = \frac{n}{4}$ and at most $\frac{n}{2}$. For an up-set $A \in V_1 \cup V_2$, we let $w(A) = \hat{w}_A(v)$. Given an edge $e = (v_1, v_2) \in E$, we let $v_1(e) = v_1$ and $v_2(e) = v_2$. We now have

$$\frac{n}{4} \cdot \sum_{v_2 \in V_2} w(v_2) \;\leq\; \sum_{e \in E} w(v_2(e)) \;\leq\; r \cdot \sum_{e \in E} w(v_1(e)) \;\leq\; r \cdot \frac{n}{2} \cdot \sum_{v_1 \in V_1} w(v_1)\;.$$

As $\hat{w}_{H/L,x}(v) = \sum_{v_1 \in V_1} w(v_1)$ and $\hat{w}_{H,x/L}(v) = \sum_{v_2 \in V_2} w(v_2)$, we get the right inequality in the statement of the lemma. The left inequality is obtained in a similar manner. □

We now complete the presentation of the lower bound by giving proofs of Lemmas 6.1–6.3. We note that the arguments used in the proofs of Lemmas 6.1 and 6.2 are similar to arguments used in [DHUZ01]. Most of the new ideas are used in the proof of Lemma 6.3.

Proof of Lemma 6.1. Suppose, without loss of generality, that $drop(A) \supseteq \{A \in up\text{-}set_t(v) \mid b \in A\}$ so that b is forced *not* to belong to all the up-sets A that participate in the calculation of the score $\hat{w}(v)$ of v. We therefore get

$$\hat{w}(v) = \hat{w}_{a/b}(v) + \hat{w}_{/ab}(v) \ ,$$
$$\hat{w}(v_{a>b}) \geq \hat{w}_{a/b}(v) + \frac{1}{2}\hat{w}_{/ab}(v) \ .$$

It is easy to see that if $A \in \mathcal{A}_{/ab}(v)$, $s \in A$ is a singleton, and $A' = A \cup \{a\} \setminus \{s\}$, then $\hat{w}_{A'}(v) = \hat{w}_A(v)$. Using Lemma 6.5, we get that $\hat{w}_{/ab}(v) \leq 2\hat{w}_{a/b}(v)$. We therefore immediately get that $\hat{w}_{a/b}(v) \geq \frac{1}{3}\hat{w}(v)$. Putting this all together, we get that

$$\hat{w}(v_{a>b}) \ \geq \ \hat{w}_{a/b}(v) + \tfrac{1}{2}\hat{w}_{/ab}(v) \ \geq \ \tfrac{1}{2}[\hat{w}(v) + \hat{w}_{a/b}(v)] \ \geq \ \tfrac{1}{2}(1+\tfrac{1}{3})\hat{w}(v) \ .$$

Thus, the move $v \to v_{a>b}$ is in fact a $\frac{1}{3}$-good move. □

Proof of Lemma 6.2. Assume, without loss of generality, that $b < c$. It is easy to see that

$$\begin{aligned}
\hat{w}(v) &= \hat{w}_{abc/}(v) + \hat{w}_{a/bc}(v) + \hat{w}_{ac/b}(v) + \hat{w}_{bc/a}(v) + \hat{w}_{c/ab}(v) + \hat{w}_{/abc}(v) \ , \\
\hat{w}(v_{a>b}) &= \tfrac{1}{2}\hat{w}_{abc/}(v) + \hat{w}_{a/bc}(v) + \hat{w}_{ac/b}(v) + \tfrac{1}{2}\hat{w}_{c/ab}(v) + \hat{w}_{/abc}(v) \ , \\
\hat{w}(v_{a<b}) &= \tfrac{1}{2}\hat{w}_{abc/}(v) + \hat{w}_{bc/a}(v) + \tfrac{1}{2}\hat{w}_{c/ab}(v) + \tfrac{1}{2}\hat{w}_{/abc}(v) \ .
\end{aligned}$$

(The term $\hat{w}_{/abc}(v)$ appears in $\hat{w}(v_{a>b})$ with a coefficient of 1 as we mark the non-critical outcome $a > b$ at $v_{a>b}$ with respect to all the up-sets of $\mathcal{A}_{/ab}(v)$.) As a consequence, we get that

$$\hat{w}(v_{a>b}) \geq \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{2}[\hat{w}_{a/b}(v) - \hat{w}_{b/a}(v) + \hat{w}_{/abc}(v)] \ ,$$
$$\hat{w}(v_{a<b}) \geq \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{2}[\hat{w}_{b/a}(v) - \hat{w}_{a/b}(v)] \ .$$

By adding these two inequalities, we also get that

$$\tfrac{1}{2}[\hat{w}(v_{a>b}) + \hat{w}(v_{a<b})] \geq \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{4}\hat{w}_{/abc}(v) \ .$$

To proceed, we need the following inequalities.

PROPOSITION 6.6. *If a is a singleton at v, and c, where $b < c$, is the active neighbor of b at v, then*

(i) $\hat{w}_{/abc}(v) \geq \frac{1}{15}\hat{w}_{/b}(v)$;
(ii) $\hat{w}_{a/b}(v) \leq \frac{2}{3}\hat{w}_{/b}(v)$;
(iii) $\hat{w}_{b/a}(v) \geq \frac{1}{3}\hat{w}_{b/}(v)$.

Proof. The proof of these inequalities is similar to the proof of the inequalities of Lemma 9 of [DHUZ01]. We describe here the proof of (i). The proofs of the other two inequalities is similar. Using Lemma 6.5, it is easy to see that

$$\hat{w}_{a/bc}(v) \leq 2\hat{w}_{/abc}(v), \quad \hat{w}_{ac/b}(v) \leq 8\hat{w}_{/abc}(v), \quad \hat{w}_{c/ab}(v) \leq 4\hat{w}_{/abc}(v) \ .$$

Thus,

$$\begin{aligned}\hat{w}_{/b}(v) = \ & \hat{w}_{a/bc}(v) + \ \hat{w}_{ac/b}(v) + \ \hat{w}_{c/ab}(v) + \hat{w}_{/abc}(v) \\ & \leq 2\hat{w}_{/abc}(v) + 8\hat{w}_{/abc}(v) + 4\hat{w}_{/abc}(v) + \hat{w}_{/abc}(v) \\ & \leq 15\hat{w}_{/abc}(v) \ ,\end{aligned}$$

as required. □

We thus have

$$\begin{aligned}\tfrac{1}{2}[\hat{w}(v_{a>b}) + \hat{w}(v_{a<b})] \ &\geq \ \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{4}\hat{w}_{/abc}(v) \ \geq \ \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{60}\hat{w}_{/b}(v) \ , \\ \hat{w}(v_{a<b}) \ &\geq \ \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{2}\hat{w}_{b/a}(v) - \tfrac{1}{2}\hat{w}_{a/b}(v) \ \geq \ \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{6}\hat{w}_{b/}(v) - \tfrac{1}{3}\hat{w}_{/b}(v) \ .\end{aligned}$$

As $\hat{w}(v) = \hat{w}_{b/}(v) + \hat{w}_{/b}(v)$, and $\hat{w}(v') \geq \max\{\hat{w}(v_{a<b}), \frac{1}{2}[\hat{w}(v_{a>b}) + \hat{w}(v_{a<b})]\}$, a simple calculation shows that $\hat{w}(v') \geq \frac{1}{2}(1 + \frac{1}{93})\hat{w}(v)$. The move $v \to v'$ is therefore a $\frac{1}{93}$-good move. □

Proof of Lemma 6.3. We start with the proof of part (ii) that claims that if v is labeled by the comparison $a : b$, where a is active and c is the neighbor of a, and if $v \to v'$ is not ν-good, then $\hat{w}(v_{a<b}) \geq \frac{1}{2}(1 - 3\nu)\cdot\hat{w}(v)$. We assume, without loss of generality, that $a > c$.

Consider at first the weights $\hat{w}(v_{a<b})$ and $\hat{w}(v_{a>b})$ *after* marking all the noncritical outcomes involving a and c but *before* dropping any up-sets. Using arguments similar to those used in the proof of Lemma 6.2, and noting that $a > c$, we get that

$$\begin{aligned}\hat{w}(v_{a>b}) &\geq \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{2}[\hat{w}_{a/b}(v) - \hat{w}_{b/a}(v) + \hat{w}_{abc/}(v)], \\ \hat{w}(v_{a<b}) &\geq \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{2}[\hat{w}_{b/a}(v) - \hat{w}_{a/b}(v)] \ .\end{aligned}$$

As none of the moves $v \to v_{a<b}$ and $v \to v_{a>b}$ is a ν-good move, we get that

$$\begin{aligned}-\nu\cdot\hat{w}(v) \ \leq \ \hat{w}_{b/a}(v) - \hat{w}_{a/b}(v) \ &\leq \ \nu\cdot\hat{w}(v) \ , \\ \hat{w}_{abc/}(v) \ &\leq \ 2\nu\cdot\hat{w}(v) \ .\end{aligned}$$

We now consider the weight $\hat{w}(v_{a<b})$ *after* dropping all the the up-sets that contain c. If A is an up-set and $c \in A$, then $a \in A$. (Recall that $a > c$.) The excluded up-sets are therefore the up-sets of $\mathcal{A}_{abc/}(v) \cup \mathcal{A}_{ac/b}(v)$. As the up-sets of $\mathcal{A}_{ac/b}(v)$ are not up-sets at $v_{a<b}$, we get that

$$\begin{aligned}\hat{w}(v_{a<b}) &\geq \tfrac{1}{2}\hat{w}(v) + \tfrac{1}{2}(\hat{w}_{b/a}(v) - \hat{w}_{a/b}(v) - \hat{w}_{abc/}(v)) \\ &\geq \tfrac{1}{2}(1 - 3\nu)\cdot\hat{w}(v) \ .\end{aligned}$$

The move $v \to v_{a<b}$ is therefore a 3ν-bad move, as required.

We now prove part (i) of Lemma 6.3. All the weights considered now are *before* we mark the additional noncritical outcomes and *before* we drop any up-sets. Let $B(c)$ denote the set of elements that were compared to c, on the path to the vertex v, and were found to be larger than c. Note that $a \in B(c)$. Consider the three following sets of up-sets (see Figure 6.2):

$$\begin{aligned}\mathcal{B}_1 &= \{A \mid B(c) \cap \bar{A} \neq \{a\}, \ a \in \bar{A}, \ b \in A\} \ , \\ \mathcal{B}_2 &= \{A \mid B(c) \cap \bar{A} \neq \emptyset, \ a \in A, \ b \in A\} \ , \\ \mathcal{B}_3 &= \{A \mid B(c) \subseteq A, \ c \in \bar{A}, b \in A\} \ .\end{aligned}$$

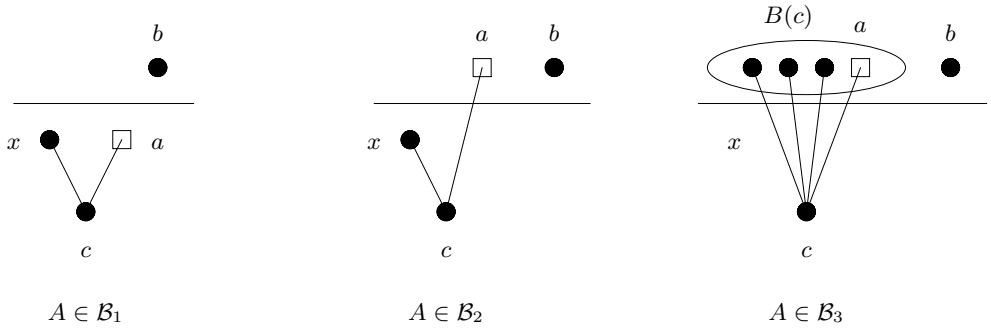


FIG. 6.2. *Up-sets from the collections $\mathcal{B}_1$,$\mathcal{B}_2$ and $\mathcal{B}_3$. Elements above the horizontal line belong to A. Elements below the horizontal line belong to $\bar{A}$.*

The crux of the lower bound proof is the fact that with respect to each up-set of $\mathcal{B}_1$ there is at least one noncritical outcome which is still unmarked at v. Indeed, let $A \in \mathcal{B}_1$. As $B(c) \cap \bar{A} \neq \{a\}$, there is an element $x \neq a$ such that $x > c$ and $x \in \bar{A}$ (note the x may depend on A). As $a, x \in \bar{A}$ and $a > c$ and $x > c$, one of the outcomes $a > c$ and $x > c$ is noncritical. Noncritical outcomes are marked only in Cases B and C. When we mark a noncritical outcome $y > z$ with respect to an up-set A such that $y, z \in \bar{A}$, we deactivate the active neighbor of z, if there were one. As c currently has an active neighbor, namely, a, we get that either the comparison $a : c$ took place after the comparison $x : c$, in which case $a > c$ is an unmarked noncritical outcome, or that the comparison $a : c$ took place before the comparison $x : c$, in which case $x > c$ is still an unmarked noncritical outcome. By marking these noncritical outcomes, the weight of v increases by at least $\hat{w}_{\mathcal{B}_1}(v)$. As the move $v \to v'$ is not ν-good, we get that $\hat{w}_{\mathcal{B}_1}(v) \leq \nu \cdot \hat{w}(v)$.

We need the following three propositions. In each one of them we assume that the number of nonsingletons in v is at most $N = n/4$.

PROPOSITION 6.7. *In Case C, if the move $v \to v'$ is not ν-good, then $\hat{w}_{\mathcal{B}_2}(v) \leq 4\nu \cdot \hat{w}(v)$.*

Proof. If $A \in \mathcal{B}_1$, s is a singleton in A, and $A' = A \cup \{a\} \setminus \{s\}$, then $A' \in \mathcal{B}_2$ and $\hat{w}_A(v) = \frac{1}{2} \cdot \hat{w}_{A'}(v)$. Using Lemma 6.5 we therefore get that $\frac{1}{4} \cdot \hat{w}_{\mathcal{B}_2}(v) \leq \hat{w}_{\mathcal{B}_1}(v)$. As $\hat{w}_{\mathcal{B}_1}(v) \leq \nu \cdot \hat{w}(v)$, we get the required inequality. □

PROPOSITION 6.8. *In Case C, if the move $v \to v'$ is not ν-good, and if the degree of c, the neighbor of a, is δ, then $\hat{w}_{abc/}(v) \geq 2^{-(\delta+1)} \cdot \hat{w}_{\mathcal{B}_3}(v)$.*

Proof. If $A \in \mathcal{B}_3$, s is a singleton in A, and $A' = A \cup \{c\} \setminus \{s\}$, then $A' \in \mathcal{A}_{abc/}(v)$. As the degree of c is δ, we get that $\hat{w}_{A'}(v) \geq 2^{-\delta} \cdot \hat{w}_A(v)$ (as all the outcomes $x > c$, where $x \in B(c)$ may be critical with respect to A'). Using Lemma 6.5 we get the required inequality. □

PROPOSITION 6.9. *In Case C, if the move $v \to v'$ is not ν-good, then $\hat{w}_{ab/}(v) > \frac{1}{4}(1 - 2\nu) \cdot \hat{w}(v)$.*

Proof. By moving a from A to $\bar{A}$, or vice versa, we get, using Lemma 6.5, that $\hat{w}_{b/a}(v) \leq \hat{w}_{ab/}(v)$ and $\hat{w}_{/ab}(v) \leq \hat{w}_{a/b}(v)$. Since $v \to v'$ is not a ν-good move, we get that $\hat{w}_{a/b}(v) \leq \hat{w}_{b/a}(v) + \nu \cdot \hat{w}(v)$. Therefore,

$$\hat{w}(v) = \hat{w}_{ab/}(v) + \hat{w}_{/ab}(v) + \hat{w}_{b/a}(v) + \hat{w}_{a/b}(v)$$

$$\leq 2\cdot\hat{w}_{ab/}(v) + 2\cdot\hat{w}_{a/b}(v)$$
$$\leq 4\cdot\hat{w}_{ab/}(v) + 2\nu\cdot\hat{w}(v),$$

as required. □

We now return to the proof of Lemma 6.3(i). It is easy to verify that

$$\hat{w}_{ab/}(v) = \hat{w}_{\mathcal{B}_2}(v) + \hat{w}_{\mathcal{B}_3}(v) + \hat{w}_{abc/}(v)\ .$$

Let δ be the degree of c. We have already shown, in the proof of part (ii), that if $v \rightarrow v_{a>b}$ is not a ν-good move, then $\hat{w}_{abc/}(v) \leq 2\nu\cdot\hat{w}(v)$. Combining this with Proposition 6.8 we get that $\hat{w}_{\mathcal{B}_3}(v) \leq \nu 2^{\delta+2}\cdot\hat{w}(v)$. Using Proposition 6.7 we therefore get that

$$\begin{aligned}\hat{w}_{ab/}(v) &= \hat{w}_{\mathcal{B}_2}(v) + \hat{w}_{\mathcal{B}_3}(v) + \hat{w}_a abc/(v)\\ &\leq 4\nu\cdot\hat{w}(v) + \nu 2^{\delta+2}\cdot\hat{w}(v) + 2\nu\cdot\hat{w}(v)\\ &= 2\nu(2^{\delta+1}+3)\cdot\hat{w}(v)\ .\end{aligned}$$

Using Proposition 6.9, we get that

$$\frac{1}{4}(1-2\nu)\ \leq\ 2\nu(2^{\delta+1}+3)\ .$$

As $\nu = 2^{-70}$, we get that $\delta > \Delta = 65$. This completes the proof of Lemma 6.3. □

7. Concluding remarks. We described an improved method for obtaining lower bounds for the selection problem. Our method combines leaf counting and adversary arguments. Although the improvement obtained in the lower bound for median selection is tiny, we think that it represents an important step towards obtaining further improved lower bounds for this very interesting and challenging problem.

REFERENCES

[BFP+73] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan, *Time bounds for selection*, J. Comput. System Sci., 7 (1973), pp. 448–461.

[BJ85] S. W. Bent and J. W. John, *Finding the median requires 2n comparisons*, in Proceedings of the 17th Annual ACM Symposium on Theory of Computing, Providence, RI, 1985, pp. 213–216.

[CM89] W. Cunto and J. I. Munro, *Average case selection*, J. ACM, 36 (1989), pp. 270–279.

[DHUZ01] D. Dor, J. Håstad, S. Ulfberg, and U. Zwick, *On lower bounds for selecting the median*, SIAM J. Discrete Math., 14 (2001), pp. 299–311.

[Dor95] D. Dor, *Selection Algorithms*, Ph.D. thesis, Department of Computer Science, Tel Aviv University, Tel Aviv, Israel, 1995.

[DZ99] D. Dor and U. Zwick, *Selecting the median*, SIAM J. Comput., 28 (1999), pp. 1722–1758.

[DZ96] D. Dor and U. Zwick, *Median selection requires* $(2+\epsilon)n$ *comparisons*, in Proceedings of the 37th Annual Symposium on Foundations of Computer Science, Burlington, VT, 1996, pp. 125–134.

[FG78] F. Fussenegger and H. N. Gabow, *A counting approach to lower bounds for selection problems*, J. ACM, 26 (1978), pp. 227–238.

[FJ59] L. R. Ford and S. M. Johnson, *A tournament problem*, Amer. Math. Monthly, 66 (1959), pp. 387–389.

[FR75] R. W. Floyd and R. L. Rivest, *Expected time bounds for selection*, Comm. ACM, 18 (1975), pp. 165–173.

[HL71] F. K. Hwang and S. Lin, *Optimal merging of* 2 *elements with n elements*, Acta Inform., 1 (1971), pp. 145–158.

[Joh88] J. W. John, *A new lower bound for the set-partitioning problem*, SIAM J. Comput., 17 (1988), pp. 640–647.

[Kir74] D. G. Kirkpatrick, *Topics in the Complexity of Combinatorial Algorithms*, Technical report 74, University of Toronto, Toronto, Canada, 1974.

[Kir81] D. G. Kirkpatrick, *A unified lower bound for selection and set partitioning problems*, J. ACM, 28 (1981), pp. 150–165.

[Kis64] S. S. Kislitsyn, *On the selection of the kth element of an ordered set by pairwise comparisons*, Sibirsk. Math. Zh., 5 (1964), pp. 557–564.

[Knu98] D.E. Knuth, *Sorting and Searching, Vol.* 3 *of The Art of Computer Programming*, 2nd ed., Addison-Wesley, Reading, MA, 1998.

[MP82] I. Munro and P. V. Poblete, *A Lower Bound for Determining the Median*, Technical report CS-82-21, University of Waterloo, Waterloo, Canada, 1982.

[Pat96] M. S. Paterson, *Progress in selection*, in Proceedings of the Fifth Scandinavian Workshop on Algorithm Theory, Reykjavík, Iceland, 1996, pp. 368–379.

[Poh72] I. Pohl, *A sorting problem and its complexity*, Comm. ACM, 15 (1972), pp. 462–464.

[PY73] V. Pratt and F. Yao, *On lower bounds for computing the ith largest element*, in Proceedings of the 14th Annual IEEE Symposium on Switching and Automata theory, 1973, pp. 70–81.

[Sch32] J. Schreier, *On tournament elimination systems*, Math. Polska, 7 (1932), pp. 154–160 (in Polish).

[SPP76] A. Schönhage, M. Paterson, and N. Pippenger, *Finding the median*, J. Comput. System Sci., 13 (1976), pp. 184–199.

[SY80] P. K. Stockmeyer and F. F. Yao, *On the optimality of linear merge*, SIAM J. Comput., 9 (1980), pp. 85–90.

[Tro92] W. T. Trotter, *Combinatorics and Partially Ordered Sets: Dimention Theory*, The Johns Hopkins University Press, Baltimore, MD, 1992.

[Yap76] C. K. Yap, *New Lower Bounds for Medians and Related Problems*, Comput. Sci. Report 79, Yale University, New Haven, CT, 1976.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 326–355

THE MAXIMUM EDGE-DISJOINT PATHS PROBLEM IN BIDIRECTED TREES*

THOMAS ERLEBACH† AND KLAUS JANSEN‡

Abstract. A bidirected tree is the directed graph obtained from an undirected tree by replacing each undirected edge by two directed edges with opposite directions. Given a set of directed paths in a bidirected tree, the goal of the maximum edge-disjoint paths problem is to select a maximum-cardinality subset of the paths such that the selected paths are edge-disjoint. This problem can be solved optimally in polynomial time for bidirected trees of constant degree but is APX-hard for bidirected trees of arbitrary degree. For every fixed $\varepsilon > 0$, a polynomial-time $(5/3+\varepsilon)$-approximation algorithm is presented.

Key words. approximation algorithms, edge-disjoint paths, bidirected trees

AMS subject classifications. 68Q25, 68R10

PII. S0895480199361259

1. Introduction.

Research on disjoint paths problems in graphs has a long history [14]. In recent years, edge-disjoint paths problems have been brought into focus by advances in the field of communication networks. Many modern network architectures establish a virtual circuit between sender and receiver in order to achieve guaranteed quality of service. When a connection request is accepted, the network must allocate sufficient resources on all links along a path from the sender to the receiver. Edge-disjoint paths problems are at the heart of the arising resource allocation problems.

We study the maximum edge-disjoint paths problem (MEDP) for bidirected tree networks. A bidirected tree is the directed graph obtained from an undirected tree by replacing each undirected edge by two directed edges with opposite directions. Bidirected tree networks have been studied intensively because they are a good model for optical networks with pairs of unidirectional fiber links between adjacent nodes [28, 20, 27, 13, 12, 26, 16].

1.1. Preliminaries.

MEDP in bidirected trees is defined as follows. Given a bidirected tree $T = (V, E)$ and a set P of simple, directed paths in T, the goal is to find a subset $P' \subseteq P$ such that the paths in P' are edge-disjoint and the cardinality of P' is maximized. We say that an algorithm is a ρ-approximation algorithm for MEDP if it always outputs a subset $P' \subseteq P$ of edge-disjoint paths whose cardinality is at least a $(1/\rho)$-fraction of the cardinality of an optimal solution.

The *conflict graph* of a set of directed paths in a bidirected tree is an undirected graph with a vertex for each path and an edge between two vertices if the corresponding paths intersect (i.e., if they share an edge). One can view MEDP in bidirected trees as a maximum independent set problem in the conflict graph.

*Received by the editors September 6, 1999; accepted for publication (in revised form) May 21, 2001; published electronically August 3, 2001. A preliminary version of this article has appeared in *Proceedings of the Ninth Annual International Symposium on Algorithms and Computation (ISAAC'98)*, Lecture Notes in Comput. Sci. 1533, Springer-Verlag, Berlin, 1998, pp. 179–188.
http://www.siam.org/journals/sidma/14-3/36125.html

†Computer Engineering and Networks Laboratory, ETH Zürich, CH-8092 Zürich, Switzerland (erlebach@tik.ee.ethz.ch).

‡Institute of Computer Science and Applied Mathematics, Christian-Albrechts-University of Kiel, Olshausenstr. 40, D-24098 Kiel, Germany (kj@informatik.uni-kiel.de).

We assume that the given tree is rooted at an arbitrary node. For a node v, we let $p(v)$ denote the parent of v. The *level* of a node is then defined as its distance to the root node. The root has level zero. We say that a path *touches* a node if it begins at that node, passes through that node, or ends at that node. The level of a path is the minimum of the levels of all nodes it touches. The unique node on a path whose level is equal to the level of the path is the *least common ancestor* (*lca*) of the path. We denote a path that begins at node u and ends at node v by (u, v) and its *lca* by $lca(u, v)$.

We will occasionally make use of the following property of bipartite graphs: for $s = 1$ or $s = 2$, the fact that a maximum matching in a bipartite graph G has cardinality s implies that there are s vertices in G such that every edge is incident to at least one of these s vertices. (The property holds for arbitrary values of s and is known as the König theorem [25]; see, e.g., the book by Berge [6, pp. 132–133].)

1.2. Results. First, in section 2, we determine the complexity of MEDP in bidirected trees: MEDP can be solved optimally in polynomial time in bidirected trees of constant degree and in bidirected stars, but it is APX-hard in bidirected trees of arbitrary degree. The main result of this paper is summarized by the following theorem.

THEOREM 1.1. *For every fixed $\varepsilon > 0$, there is a polynomial-time approximation algorithm for the maximum edge-disjoint paths problem in bidirected trees with approximation ratio $5/3 + \varepsilon$.*

The description of the algorithm and a proof that the claimed approximation ratio is indeed achieved appear in section 3. In section 4, we discuss how some of our results can be generalized to the weighted version of the problem and to the maximum path coloring problem.

1.3. Related work.

Path coloring in bidirected trees. Previous work on bidirected trees has focused on the path coloring problem: Given a set of directed paths in a bidirected tree, assign colors to the paths such that paths receive different colors if they share an edge. The goal is to minimize the total number of colors used. This problem is $\mathcal{NP}$-hard even for binary trees [11, 26]. The best known approximation algorithms [12, 13] use at most $\lceil 5L/3 \rceil$ colors, where L is the maximum load (the load of an edge is the number of paths using that edge) and thus a lower bound on the optimal solution. Previous algorithms had used $15L/8$ colors [28] and $7L/4$ colors [20, 27] in the worst case. For binary trees, a randomized path coloring algorithm that uses at most $7L/5$ colors with high probability under certain conditions has been obtained [1]. For the special case of all-to-all path coloring in bidirected trees, it was shown that the optimal number of colors is equal to the maximum load [16].

Multicommodity flow in trees. Garg, Vazirani, and Yannakakis [15] studied the integral multicommodity flow problem in undirected trees, which is a generalization of MEDP in undirected trees. They showed that the problem with unit edge capacities (equivalent to MEDP in undirected trees) can be solved optimally in polynomial time. For undirected trees with edge capacities one or two, they proved the problem MAX SNP-hard. They also presented a 2-approximation algorithm for integral multicommodity flow in trees. It works by considering the demands in order of nonincreasing levels of their *lcas* and by satisfying them greedily. This approximation algorithm can be adapted to MEDP in bidirected trees, where it also gives a 2-approximation. The main idea that leads to our improved approximation algorithm for MEDP in

bidirected trees is to consider all paths with the same *lca* simultaneously instead of one by one.

On-line algorithms for MEDP in trees. MEDP has also been studied in the on-line scenario, where the paths are given to the algorithm one by one. The algorithm must accept or reject each path without knowledge about future requests. Preemption is not allowed. It is easy to see that no deterministic algorithm can have a competitive ratio better than the diameter of the tree in this case. Awerbuch et al. gave a randomized algorithm with competitive ratio $O(\log n)$ for undirected trees with n nodes [4]. Their algorithm also works for bidirected trees. An improved randomized algorithm with competitive ratio $O(\log d)$ for undirected trees with diameter d was given in [5].

MEDP for other topologies. If MEDP is studied for arbitrary graphs, the algorithm must solve both a routing problem and a selection problem. For arbitrary directed graphs with m edges, MEDP was recently shown to be $\mathcal{NP}$-hard to approximate within $m^{1/2-\varepsilon}$ [18]. Approximation algorithms with approximation ratio $O(\sqrt{m})$ are known [22]; see also [24, 30]. Better approximation ratios can be achieved for restricted classes of graphs. For a class of planar graphs containing two-dimensional mesh networks, an $O(1)$-approximation algorithm has been devised in [23]. We remark that the techniques we use in this paper are completely different from those used in [23].

2. Complexity results. First, we prove that MEDP in bidirected trees is APX-complete. APX is the class of all optimization problems in NPO that admit constant-factor approximation algorithms. (NPO contains all optimization problems whose instances and solutions can be recognized in polynomial time, which have solutions of polynomial size, and for which the objective value of a given solution can be computed in polynomial time.) APX-hardness implies that there is some $\delta > 1$ such that no δ-approximation algorithm can exist unless $\mathcal{P} = \mathcal{NP}$. In particular, there cannot be a polynomial-time approximation scheme for an APX-hard problem unless $\mathcal{P} = \mathcal{NP}$. See [2] for further background information concerning complexity classes for optimization problems, approximation preserving reductions, and a discussion of the relationship of APX-hardness to the earlier concept of MAX SNP-hardness.

We recall the definition of AP-reducibility from [2].

DEFINITION 2.1. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two optimization problems in* NPO. *For a solution y to an instance x of $\mathcal{P}_i$, let $ratio_{\mathcal{P}_i}(x, y)$ denote the ratio between the value of an optimal solution to x and the value of y (or the reciprocal of this ratio, whichever is larger than 1). $\mathcal{P}_1$ is called* AP-reducible *to $\mathcal{P}_2$ if two functions f and g and a positive constant $\alpha > 1$ with the following properties exist:*

(i) *For any instance x of $\mathcal{P}_1$ and for any rational $r > 1$, $f(x, r)$ is an instance of $\mathcal{P}_2$.*

(ii) *For any instance x of $\mathcal{P}_1$ and for any rational $r > 1$, if there is a solution to x, then there is also a solution to $f(x, r)$.*

(iii) *For any instance x of $\mathcal{P}_1$, for any rational $r > 1$, and for any solution y to $f(x, r)$, $g(x, y, r)$ is a solution to x.*

(iv) *f and g are computable in polynomial time for any fixed rational r.*

(v) *For any instance x of $\mathcal{P}_1$, for any rational $r > 1$, and for any solution y to $f(x, r)$, $ratio_{\mathcal{P}_2}(f(x, r), y) \leq r$ implies $ratio_{\mathcal{P}_1}(x, g(x, y, r)) \leq 1 + \alpha(r - 1)$.*

Intuitively, if $\mathcal{P}_1$ is AP-reducible to $\mathcal{P}_2$, then an r-approximation algorithm for $\mathcal{P}_2$ implies the existence of a $(1 + \alpha(r - 1))$-approximation algorithm for $\mathcal{P}_1$.

THEOREM 2.2. *MEDP in bidirected trees is* APX*-complete.*

Proof. We present an AP-reduction from the APX-complete maximum three-

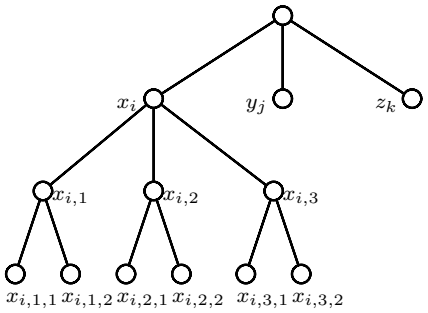


FIG. 2.1. *Construction of the tree for the three-dimensional matching problem.*

dimensional matching problem. The reduction is similar to the one used by Garg, Vazirani, and Yannakakis to prove MAX SNP-hardness of integral multicommodity flow in undirected trees with edge capacities one and two [15].

An instance of the maximum three-dimensional matching problem is given by three disjoint sets X, Y, Z with $|X| = |Y| = |Z| = n$ and a set of triples $S \subseteq X \times Y \times Z$. A *matching* of S is a subset $M \subseteq S$ such that no two elements of M agree in any coordinate. The goal is to compute a matching of maximum cardinality. This problem is known to be APX-complete even if each element of $X \cup Y \cup Z$ occurs in at most three triples in S [21]. In this bounded version of the problem, each triple can intersect at most six other triples. Then a greedy algorithm that chooses disjoint triples in arbitrary order will give a matching of cardinality at least $|S|/7$.

Let an instance I of the maximum three-dimensional matching problem with at most three occurrences of each element of $X \cup Y \cup Z$ be given as above. The function f of the AP-reduction is defined by the following construction of an instance I' of MEDP in bidirected trees—it does not depend on r. We create a bidirected tree T of depth three as follows. The root of T has $3n$ children: one for each $x_i \in X$, one for each $y_j \in Y$, and one for each $z_k \in Z$. Each node $x_i \in X$ has p_i children $x_{i,1},\ldots,x_{i,p_i}$, where $p_i \leq 3$ is the number of occurrences of x_i in S. Each $x_{i,j}$ has two children $x_{i,j,1}$ and $x_{i,j,2}$. See Figure 2.1 for a sketch of this construction; note that only a small subset of the nodes of the tree is actually shown there and that pairs of oppositely directed edges are depicted as undirected edges for the sake of simplicity.

Now, we create a set P of paths in T. Number the occurrences of x_i in the triples of S from 1 to p_i arbitrarily. For every triple in S we add three paths to P. Let triple (x_i, y_j, z_k) contain the lth occurrence of x_i in S. Then we add a path from $x_{i,l,1}$ to y_j, a path from z_k to $x_{i,l,2}$, and a path from $x_{i,l,1}$ to $x_{i,l,2}$.

We claim that S contains d disjoint triples if and only if P contains a subset P' of at least $|S| + d$ edge-disjoint paths and that d disjoint triples in S can be computed efficiently from a given subset $P' \subseteq P$ of at least $|S| + d$ edge-disjoint paths.

Assume that S contains d disjoint triples $(x_{i_1}, y_{j_1}, z_{k_1}),\ldots,(x_{i_d}, y_{j_d}, z_{k_d})$. Let triple $(x_{i_t}, y_{j_t}, z_{k_t})$ contain the l_tth occurrence of x_{i_t}. Then the following $|S| + d$ paths form a set P' of edge-disjoint paths: for each t, $1 \leq t \leq d$, choose the path from $x_{i_t,l_t,1}$ to y_{j_t}, the path from z_{k_t} to $x_{i_t,l_t,2}$, and $p_{i_t} - 1$ paths from $x_{i_t,l,1}$ to $x_{i_t,l,2}$ for $l \in \{1,\ldots,p_{i_t}\} \setminus \{l_t\}$; for each x_i that does not occur in the d disjoint triples, choose the p_i paths from $x_{i,l,1}$ to $x_{i,l,2}$ for $1 \leq l \leq p_i$.

Conversely, assume that there is a subset P' of P containing $|S| + d$ edge-disjoint paths. Note that P' can contain at most one path entering the subtree rooted at x_i from above and at most one path leaving the subtree rooted at x_i. The only

possibility for P' to contain more than p_i paths using edges of the subtree rooted at x_i is to contain one path from $x_{i,l_i,1}$ to some y_j, one path from some z_k to $x_{i,l_i,2}$, and $p_i - 1$ paths from $x_{i,l,1}$ to $x_{i,l,2}$ for $l \in \{1, 2, \ldots, p_i\} \setminus \{l_i\}$. In that case, P' contains $p_i + 1$ paths using edges of the subtree rooted at x_i. The only way for P' to contain at least $|S| + d$ paths is that P' contains exactly $p_i + 1$ paths using edges of the subtree rooted at x_i for at least d values of i. Then a set of d disjoint triples is obtained by taking, for each of the d values of i, the triple containing the l_ith occurrence of x_i.

The function g takes as arguments the instance I, a solution P' to I', and r. The instance I' itself is not an argument of g, but since I' can be computed from I using the function f, we can assume that I' is available during the computation of the function g. The computation of g does not depend on r and is done as follows. First, a greedy solution M_1 to I is computed. Second, a matching M_2 is obtained by taking the triple (x_i, y_j, z_k) for each pair of paths from $x_{i,l,1}$ to some y_j and from some z_k to $x_{i,l,2}$ that are contained in P'. Then we take the larger of M_1 and M_2 as the function value of $g(I, P', r)$. We have $|g(I, P', r)| = \max\{|M_1|, |M_2|\}$. Furthermore, we have $|M_1| \geq |S|/7$ by the remark about the greedy algorithm at the beginning of this proof and $|M_2| \geq |P'| - |S|$ by the arguments above.

Properties (i)–(iv) of an AP-reduction are clearly satisfied. It remains to show (v). Let d^* be the cardinality of an optimal matching for the original instance I of the maximum three-dimensional matching problem. Then an optimal solution to the constructed instance I' of MEDP consists of $|S| + d^*$ paths, as shown above. Note that $|S|/7 \leq d^* \leq |S|$. Assume that we have a solution P' to I' containing at least $(|S| + d^*)/r$ paths.

If $r \geq 13/12$, we can use $|g(I, P', r)| \geq |M_1| \geq |S|/7 \geq d^*/7$ to show that (v) holds with $\alpha = 72$, since $7 \leq 1 + 72(r - 1)$ in this case. Therefore assume that $r < 13/12$. If $|P'| \geq (|S| + d^*)/r$, we get

$$|P'| \geq \frac{|S| + (r-1)|S| + d^* - (r-1)|S|}{r} = |S| + \frac{d^* - (r-1)|S|}{r} \geq |S| + \frac{d^*}{\frac{r}{1-7(r-1)}} \ ,$$

where the last inequality used $|S| \leq 7d^*$. Thus we have $|g(I, P', r)| \geq |M_2| \geq |P'| - |S| \geq (1 - 7(r-1))d^*/r$. For $1 < r < 13/12$ we get

$$\frac{r}{1-7(r-1)} = \frac{r}{8-7r} = 1 + \frac{8}{8-7r}(r-1) \leq 1 + 19.2(r-1) < 1 + 72(r-1).$$

Again, (v) holds with $\alpha = 72$. Therefore, the reduction is indeed an AP-reduction, and we have proved that MEDP in bidirected trees is APX-hard. Since MEDP in bidirected trees is contained in APX by our main result (Theorem 1.1), we get that the problem is APX-complete. □

Nevertheless, MEDP can be solved optimally in polynomial time if the input is restricted in certain ways. First, consider the case that the maximum degree of the given tree is bounded by a constant. The optimal solution can be computed by dynamic programming in this case. We process the nodes of the tree in order of nonincreasing levels. At every node v, we record for each possible subset S of edge-disjoint paths touching v and its parent (note that $|S| \leq 2$) the maximum number of paths contained in the subtree rooted at v that can be accepted in addition to the paths in S. Node v is processed only when these values are known for all its children. We can then enumerate all possible edge-disjoint subsets of paths touching v. For each such subset, we can look up the corresponding values stored at children of v and update the values stored at v accordingly. Note that there are only polynomially

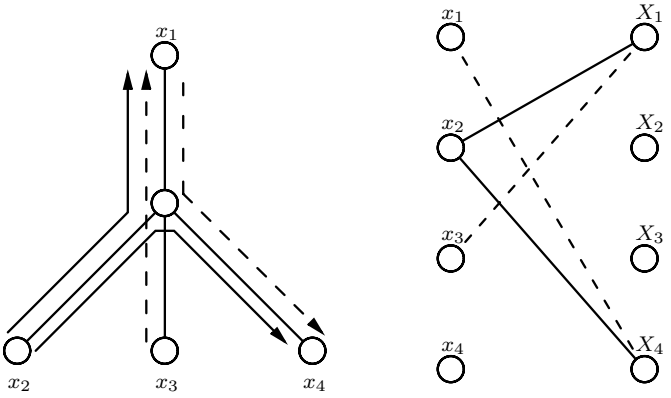


FIG. 2.2. *Reducing MEDP in a star to bipartite matching.*

many subsets to consider at each node. When the root node has been processed, the optimal solution can easily be constructed.

Another special case that can be solved optimally in polynomial time is the case that the given bidirected tree T is a star, i.e., it contains only one node with degree greater than one. MEDP in bidirected stars can be reduced to the maximum matching problem in a bipartite graph as follows. (See Figure 2.2 for an example, in which a subset of edge-disjoint paths and the corresponding matching in the bipartite graph are drawn with dashed lines.) First, we can assume without loss of generality that every given path uses exactly two edges of the star; if a path uses only one edge, we can add a new node to the star and extend the path by one edge without changing the set of solutions. Now, observe that every path uses exactly one edge directed towards the center and one edge directed away from the center of the star. Construct a bipartite graph G by including a vertex for every edge of the star and by adding an edge between two vertices u and v in G for every path in T that uses the edges corresponding to u and v. Two paths in T are edge-disjoint if and only if the corresponding edges in G do not share an endpoint. Sets of edge-disjoint paths in T correspond to matchings in G. A maximum matching in G can be computed in polynomial time [19].

The latter result can actually be generalized from stars to *spiders*. A spider is a bidirected tree in which at most one node (the center) has degree greater than two. MEDP in a bidirected spider can be solved in polynomial time using an algorithm for the maximum-weight bipartite matching problem as a subroutine. The bipartite graph G is constructed as above from the paths touching the center of the spider, and the weight of an edge e in G specifies how many fewer paths not touching the center of the spider can be accepted if the path corresponding to e is accepted. The details are left to the reader.

3. Approximating the optimal solution. Fix any $\varepsilon > 0$. We assume that $\varepsilon \leq 1/3$ because otherwise $5/3 + \varepsilon > 2$ and the simple bottom-up greedy algorithm derived from [15] achieves approximation ratio 2 already. Let an instance of the maximum edge-disjoint paths problem be given by a bidirected tree T and a set P of directed paths in T. Denote by P^* an arbitrary optimal solution for the given instance.

The algorithm proceeds in two passes. In the first pass, it processes the nodes of T in order of nonincreasing levels (i.e., bottom-up). Assume that the algorithm

is about to process node v. Let P_v denote the subset of all paths $(u, w) \in P$ with $lca(u, w) = v$ that do not intersect any of the paths that have been accepted by the algorithm at a previous node and that do not use any edges that have been *reserved* or *fixed* by the algorithm (see below). For the sake of simplicity, we can assume without loss of generality that we have $u \neq v \neq w$ for all paths $(u, w) \in P_v$; otherwise, we could add an additional child to v for each path in P_v starting or ending at v and make the path start or end at this new child instead. Every path $p \in P_v$ uses exactly two edges incident to v, and we refer to these two edges as the *top edges* of p. We say that two paths (u_1, w_1) and (u_2, w_2) with *lca* v are *equivalent* if they use the same two edges incident to v, i.e., if their top edges are the same. For a set Q of paths with the same *lca*, this defines a partition of Q into different *equivalence classes of paths* in the natural way.

While the algorithm processes node v, it tries to determine for the paths in P_v whether they should be included in the solution (these paths are called *accepted*) or should not (these paths are called *rejected*). Sometimes, however, the algorithm cannot make this decision right away. In these cases the algorithm will leave some paths in an intermediate state and resolve them later on. The possibilities for paths in such intermediate states are

(i) undetermined paths,
(ii) groups of deferred paths,
(iii) groups of exclusive paths, and
(iv) groups of 2-exclusive paths.

We refer to undetermined paths and to paths in groups of exclusive paths and in groups of 2-exclusive paths as *unresolved paths* and to paths in groups of deferred paths as *deferred paths*. The status of unresolved paths is resolved at later nodes during the first pass. The second pass of the algorithm proceeds top-down and accepts one path from each group of deferred paths.

3.1. Paths in intermediate states. In the following we give explanations regarding the possible groups of paths in intermediate states. First, the algorithm will sometimes leave a single path p of P_v in an undetermined state. If P_v has only one equivalence class of paths, accepting a path $p \in P_v$ might cause the algorithm to miss the chance of accepting two paths of smaller level (than v) later on. Hence, the algorithm could at best achieve a 2-approximation. Therefore, instead of accepting or rejecting the paths in P_v right away, the algorithm picks one of them and makes it an *undetermined path*. All other paths in P_v, if any, are rejected, and the undetermined path will be accepted or rejected at a later node.

A second situation in which the algorithm does not accept or reject all paths in P_v right away is sketched in Figure 3.1. (Here and in the following, pairs of oppositely directed edges are drawn as undirected edges in all figures.) In this situation, the algorithm decides to accept one of several intersecting paths from P_v, but it defers the decision of which one of them to accept. The intersecting paths are called a *group of deferred paths*. All paths in a group of deferred paths use the same edge incident to v and to a child c of v. In the figure, this is the edge (c, v). (The case that the deferred paths share the edge (v, c) is symmetrical.) Furthermore, each deferred path also uses an edge (v, c') connecting v and a child $c' \neq c$, and not all of the deferred paths use the same such edge. If the algorithm decides to create a new group of deferred paths, it marks the edge (c, v) as *reserved* (ensuring that no path accepted at a node processed after v can use the edge) but leaves all edges (v, c') for children $c' \neq c$ available. The reserved edge is indicated by a dashed arrow in Figure 3.1. The

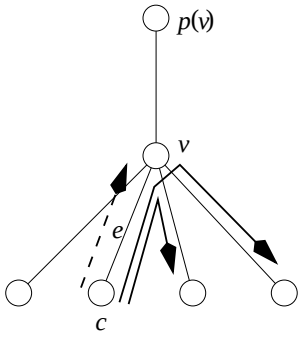


FIG. 3.1. *A group of deferred paths.*

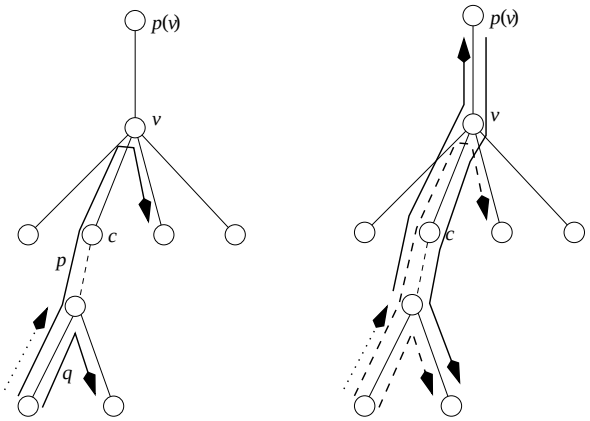


FIG. 3.2. *Possible configuration of a group of exclusive paths (left-hand side), and a situation in which both exclusive paths are blocked (right-hand side).*

motivation for introducing groups of deferred paths is as follows: first, the reserved edge blocks at most one path of smaller level that could be accepted in an optimal solution; second, no matter which path using the edge $(p(v), v)$ is accepted at a node processed after v, that path uses at most one of the edges (v, c'), and as there is still at least one deferred path that does not use that particular edge (v, c'), the algorithm can pick such a deferred path in the second pass. When processing later nodes during the first pass, the algorithm actually treats the group of deferred paths like a single accepted path that uses only the reserved edge of the deferred paths.

A *group of exclusive paths* is sketched in Figure 3.2 (left-hand side). Such a group consists of one path q (called the *lower path*) contained in the subtree rooted at a child c of v and one path p (called the *higher path*) with *lca* v that intersects q. At most one of the two paths can be accepted, but if the algorithm picks the wrong one this choice can cause the algorithm to accept only one path, while the optimal solution would accept the other path and one or two additional paths. Hence, the algorithm defers the decision of which path to accept until a later node. For now, it marks only the top edge of path q that is intersected by p as fixed. (Fixed edges are indicated by dotted arrows in our figures.) Obviously, a group of exclusive paths has the following property.

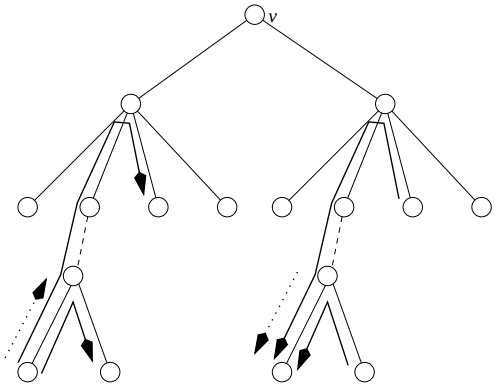


FIG. 3.3. *Group of* 2*-exclusive paths consisting of a pair of independent groups of exclusive paths.*

PROPERTY (E). *If at most one path touching v but not using the fixed edge is accepted at a later node, either p or q can still be accepted. Only when two paths touching v are accepted at a later node can they block p and q from being accepted.*

The right-hand side of Figure 3.2 shows how two paths accepted at a later node can block both exclusive paths. While processing later nodes, the algorithm will try to avoid this whenever possible.

The last types of unresolved paths are sketched in Figures 3.3 and 3.4. These *groups of* 2*-exclusive paths* consist of a set of four paths at most two of which can be accepted. More precisely, the first possibility for a group of 2-exclusive paths is to consist of two independent groups of exclusive paths (Figure 3.3), i.e., of two groups of exclusive paths such that the fixed edge of one group is directed towards the root and the fixed edge of the other group is directed towards the leaves. Furthermore, the two groups must either be contained in disjoint subtrees (as shown in Figure 3.3) or have only their lower paths contained in disjoint subtrees and their higher paths not intersecting each other. A pair of independent groups of exclusive paths has two fixed edges: the fixed edges of both groups.

The second possibility for a group of 2-exclusive paths is to consist of a group of exclusive paths contained in a subtree rooted at a child of v and two paths p_1 and p_2 with *lca* v that intersect the exclusive paths (but not their fixed edge) in a way such that accepting p_1 and p_2 would block both of the exclusive paths from being accepted (Figure 3.4). Two edges are marked fixed, namely, the top edge of the higher exclusive path intersected by a path with *lca* v and the top edge of the lower exclusive path intersected by a path with *lca* v. It is not difficult to show by case analysis that a group of 2-exclusive paths has the following property.

PROPERTY (2E). *If at most one path touching v but not using a fixed edge is accepted at a later node, two paths from the group of* 2*-exclusive paths can still be accepted. If two paths touching v but not using a fixed edge are accepted at a later node, at least one path from the group of* 2*-exclusive paths can still be accepted.*

While processing later nodes, the algorithm will try to avoid accepting two paths touching v such that only one path from the group of 2-exclusive paths can be accepted.

3.2. Invariants. In section 3.4 we will present the details of how the algorithm proceeds during the first pass. At the same time, we will show that the approximation ratio achieved by the algorithm is $5/3 + \varepsilon$. In order to establish this, we will prove by

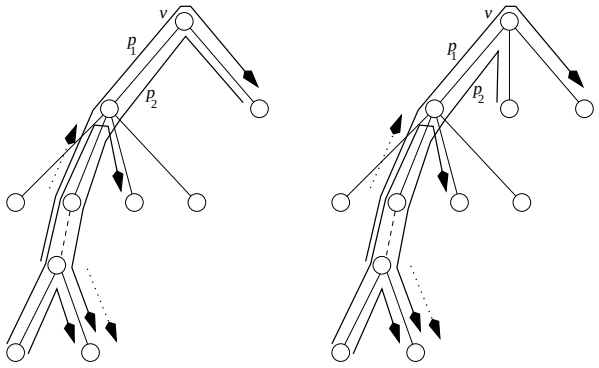


FIG. 3.4. *Further configurations of groups of* 2*-exclusive paths.*

induction that the following invariants can be maintained throughout the first pass. These invariants hold before the first node of T is processed, and they hold again each time an additional node of T has been processed. A node v is called a *root of a processed subtree* if the node v has already been processed but its parent has not.

INVARIANT A. *For every root v of a processed subtree, all paths in that subtree are accepted, rejected, or deferred except if one of the following cases occurs:*

(i) *The subtree contains one undetermined path. All other paths contained in the subtree are accepted, rejected, or deferred. No edge in the subtree is marked fixed.*

(ii) *The subtree contains one group of exclusive paths. All other paths contained in the subtree are accepted, rejected, or deferred. The only edge marked fixed in the subtree is the one from the group of exclusive paths.*

(iii) *The subtree contains one group of* 2*-exclusive paths. All other paths contained in the subtree are accepted, rejected, or deferred. The only edges marked fixed in the subtree are the two from the group of* 2*-exclusive paths.*

All accepted paths are edge-disjoint and do not contain any reserved edges. Every unresolved path is edge-disjoint from all accepted paths and does not contain any reserved edges. Every deferred path contains exactly one reserved edge: the reserved edge of the group of deferred paths to which the path belongs. If a deferred path p intersects an accepted or unresolved path q, then the level of q is smaller than that of p.

INVARIANT B. *Let A be the set of all paths that have already been accepted by the algorithm. Let F be the set of all paths in P whose lca has not yet been processed and which are not blocked by any of the accepted paths, by reserved edges, or by fixed edges. Let d be the number of groups of deferred paths that are contained in processed subtrees. Let U be the set of all undetermined paths. Let X be the union of all groups of exclusive paths and groups of* 2*-exclusive paths. Then there is a subset $O \subseteq F \cup U \cup X$ of edge-disjoint paths satisfying the following conditions:*

(a) $|P^*| \leq (5/3 + \varepsilon)(|A| + d) + |O|$.

(b) *For every group of exclusive paths, O contains one path from that group; for every group of* 2*-exclusive paths, O contains two paths from that group.*

Intuitively, the set O represents a subset of P containing edge-disjoint paths that could still be accepted by the algorithm and that has the following property: if the algorithm accepts at least a $1/(5/3+\varepsilon)$-fraction of the paths in O (in addition to the paths it has already accepted), its output is a $(5/3+\varepsilon)$-approximation of the optimal solution.

Observe that the invariants are satisfied initially with $A = \emptyset$, $d = 0$, $F = P$,

$U = \emptyset$, $X = \emptyset$, and $O = P^*$. While it will be easy to see from the description of the algorithm that Invariant A is indeed maintained throughout the first pass, special care must be taken to prove that Invariant B is maintained as well.

3.3. The second pass. If the invariants are satisfied after the root node is processed, we have $F = \emptyset$, $O \subseteq U \cup X$, and $|P^*| \leq (5/3 + \varepsilon)(|A| + d) + |O|$. At this time, there can still be one undetermined path (which can, but need not be contained in O: therefore, $|O| \in \{0, 1\}$ in this case), one group of exclusive paths (from which O contains exactly one path, $|O| = 1$), or one group of 2-exclusive paths (from which O contains two edge-disjoint paths, $|O| = 2$). If there is an undetermined path, the algorithm accepts it. If there is a group of exclusive paths, the algorithm accepts one of them arbitrarily. If there is a group of 2-exclusive paths, the algorithm accepts two edge-disjoint paths of them arbitrarily. The algorithm accepts at least $|O|$ additional paths in this way, and the resulting set A' of accepted paths satisfies $|A'| \geq |A| + |O|$ and, therefore, $|P^*| \leq (5/3 + \varepsilon)(|A'| + d)$.

In the second pass, the algorithm processes the nodes of the tree in reverse order, i.e., according to nondecreasing levels (top-down). At each node v that is the *lca* of at least one group of deferred paths, it accepts one path from each of the groups of deferred paths such that these paths are edge-disjoint from all previously accepted paths and from each other. This can always be done due to the definition of groups of deferred paths. Hence, the number of paths accepted by the algorithm increases by d in the second pass, and the set A'' of paths that are accepted by the algorithm in the end satisfies $|A''| = |A'| + d$ and, therefore, $|P^*| \leq (5/3 + \varepsilon)|A''|$. This establishes Theorem 1.1.

3.4. Details of the first pass. Assume that the algorithm is about to process node v. Recall that $P_v \subseteq P$ is the set of all paths with *lca* v that do not intersect any previously accepted path nor any fixed or reserved edge. Let U_v be the set of undetermined paths contained in subtrees rooted at children of v. Let X_v be the set of all paths in groups of exclusive paths and groups of 2-exclusive paths contained in subtrees rooted at children of v. In the following, we explain how the algorithm processes node v and determines which of the paths in $P_v \cup U_v \cup X_v$ should be accepted, rejected, deferred, or left (or put) in an unresolved state.

Observe that for a given set of paths with *lca* v the problem of determining a maximum-cardinality subset of edge-disjoint paths is equivalent to solving MEDP in a star and can thus be done in polynomial time by computing a maximum matching in a bipartite graph (cf. section 2). Whenever we use an expression like *compute a maximum number of edge-disjoint paths in* $S \subseteq P_v$ in the following, we imply that the computation should be carried out by employing this reduction to maximum matching.

Observe that each child of the current node v is the root of a processed subtree, which can, by Invariant A, contain at most one of the following: one undetermined path, or one group of exclusive paths, or one group of 2-exclusive paths. Let k be the number of children of v that have an undetermined path in their subtree, let ℓ be the number of children of v that have a group of exclusive paths, and let m be the number of children of v that have a group of 2-exclusive paths. We use the expression *subtrees with exclusive paths* to refer to all subtrees rooted at children of v with either a group of exclusive paths or with a group of 2-exclusive paths.

Note that one main difficulty lies in determining which of the paths in $U_v \cup X_v$ should be accepted and which should be rejected. If $k + \ell + 2m$ is bounded by a constant (this will be Case 1), all possible combinations of accepting and rejecting

paths in $U_v \cup X_v$ can be tried out in polynomial time, but if $k + \ell + 2m$ is large (this will be Case 2), the algorithm must proceed in a different way in order to make sufficiently good decisions. The exact threshold for determining when $k + \ell + 2m$ is considered large and, consequently, the running-time of the algorithm, depend on the constant ε.

Let F, U, X, A, and d denote the quantities defined in section 3.2 at the instant just before the algorithm processes node v. Let F', U', X', A', and d' denote the respective quantities right after node v is processed. Furthermore, denote by a_v the number of paths that are newly accepted while processing v and by d_v the number of groups of deferred paths that are newly created while processing v.

We can assume that there is a set $O \subseteq F \cup U \cup X$ of edge-disjoint paths satisfying conditions (a) and (b) of Invariant B before v is processed. In every single case of the following case analysis, we show how to construct a set O' that satisfies Invariant B after v is processed. O' is obtained from O by replacing paths, removing paths, or inserting paths as required. In particular, O' must be a set of edge-disjoint paths satisfying $O' \subseteq F' \cup U' \cup X'$. Therefore, all paths intersecting a newly accepted path or the reserved edge of a newly created group of deferred paths must be removed from O. Note that at most two such paths can have a smaller level than v because all such paths of smaller level must use the edge $(v, p(v))$ or $(p(v), v)$. Paths that are rejected by the algorithm must be removed or replaced in O. If a new group of exclusive paths or group of 2-exclusive paths is created, O' must contain one or two paths, respectively, from that group so that condition (b) of Invariant B is maintained. Furthermore, we must ensure that $|O'|$ is smaller than $|O|$ by at most $(\frac{5}{3}+\varepsilon)(a_v+d_v)$. As the value $|A|+d$ increases by $a_v + d_v$ while v is processed (i.e., we have $|A'| + d' = |A| + d + a_v + d_v$), this implies that condition (a) of Invariant B also holds after v is processed, i.e., $|P^*| \leq (5/3 + \varepsilon)(|A'| + d') + |O'|$.

3.4.1. Case 1: Few subtrees with unresolved paths. This case applies if $k + \ell + 2m \leq \frac{2}{\varepsilon}$. Before we go into the details, we give a brief outline of the various subcases that we consider and how we handle them. First, we can afford to compute the cardinality s of a largest set of edge-disjoint paths in $P_v \cup U_v \cup X_v$ in this case. Note that $s \geq k + \ell + 2m$. If $s = 0$, there is nothing to do at this node. If $s = 1$ or $s = 2$, we are in Case 1.1 or Case 1.2 and distinguish the subcases shown in Table 3.1. In Case 1.1, with $s = 1$, we can never accept a path because this might block two paths considered later. Therefore, we can create only undetermined paths, groups of exclusive paths, or groups of deferred paths. In Case 1.2, with $s = 2$, there must be two edges e_1 and e_2 incident to v such that all paths in P_v intersect e_1 or e_2. Furthermore, there is only a small number of different possibilities of how paths in P_v can intersect unresolved paths in subtrees under v. Therefore, we can consider all possible configurations in each of the subcases. For each configuration, we accept two paths (or create groups of deferred paths) if we can ensure that this blocks at most three paths from O. If four paths could be blocked (which is the maximum possible if $s = 2$), we create a group of 2-exclusive paths instead and argue that the invariants are still satisfied. If $s \geq 3$, we can accept all s paths since this blocks at most two paths from O of smaller level, and thus it suffices to remove at most $s + 2 \leq (5/3)s$ paths from O.

Now we give the details of the analysis of Case 1.

Case 1. $k+\ell+2m \leq \frac{2}{\varepsilon}$. The algorithm can try out all combinations of accepting or rejecting unresolved paths in the subtrees rooted at children of v: for undetermined paths there are two possibilities (accepting or rejecting the path), for groups of ex-

TABLE 3.1
Outline of subcases of Cases 1.1 *and* 1.2.

Case	k	ℓ	m	Action
1.1.1	0	0	0	Create undetermined path or group of deferred paths.
1.1.2	1	0	0	Do nothing (if $P_v = \emptyset$), or create group of exclusive paths.
1.1.3	0	1	0	Do nothing (must have $P_v = \emptyset$).
1.2.1	0	0	1	Do nothing (must have $P_v = \emptyset$).
1.2.2	0	2	0	Accept lower paths of both groups of exclusive paths, or create group of 2-exclusive paths.
1.2.3	1	1	0	Accept undetermined path and lower path of group of exclusive paths, or create group of 2-exclusive paths.
1.2.4	0	1	0	Accept two paths, or accept one path and create a group of deferred paths, or create a group of 2-exclusive paths.
1.2.5	0	0	0	Accept two paths, or accept one path and create one group of deferred paths, or create two groups of deferred paths.
1.2.6	1	0	0	Accept undetermined path and accept a disjoint path or create a group of deferred paths, or accept two paths (or create groups of deferred paths) in P_v.
1.2.7	2	0	0	Accept both undetermined paths, or create group of 2-exclusive paths.

clusive paths there are two possibilities (accepting the lower path or accepting the higher path), and for groups of 2-exclusive paths there are either four possibilities (in the case of a pair of independent groups of exclusive paths as shown in Figure 3.3: accepting the lower or higher path in one group and the lower or higher path in the other group) or two relevant possibilities (in the cases shown in Figure 3.4: accepting the lower or higher path of the group of exclusive paths contained in the group of 2-exclusive paths and the edge-disjoint path among the remaining two paths; note that accepting no path of the group of exclusive paths and only the remaining two paths blocks more paths from F than any of the other two possibilities; hence we do not need to consider this third possibility) of accepting two edge-disjoint paths of the group. Hence, the number of possible combinations is bounded from above by $2^{k+\ell}4^m = 2^{k+\ell+2m} \leq 4^{1/\varepsilon} = O(1)$. For each combination γ, the algorithm computes a maximum number s_γ of edge-disjoint paths in P_v not intersecting any of the u_γ paths from $U_v \cup X_v$ that are (tentatively) accepted for this combination. Let s be the maximum of $u_\gamma + s_\gamma$, taken over all combinations γ. Note that s is the cardinality of a maximum-cardinality subset of edge-disjoint paths in $P_v \cup U_v \cup X_v$. If $s = 0$, we have $k = \ell = m = 0$ and $P_v = \emptyset$, and the algorithm does nothing and proceeds with the next node. Otherwise, we distinguish the following cases.

Case 1.1. $s = 1$. As $s \geq k + \ell + 2m$, $s = 1$ implies $m = 0$ and $k + \ell \leq 1$.

Case 1.1.1. $k = \ell = m = 0$. If P_v has only one equivalence class of paths, pick one path, say, p, arbitrarily and make it an undetermined path. (Hence, $X' = X$ and $U' = U \cup \{p\}$.) Reject all other paths in P_v. If O contains a path $p' \neq p$ from P_v, replace p' by p in O to obtain O' (in order to ensure $O' \subseteq F' \cup U' \cup X'$); otherwise let $O' = O$. We have $a_v = d_v = 0$ and $|O| = |O'|$. Obviously, the invariants are satisfied.

If P_v has more than one equivalence class of paths, there must be an edge e incident to v that is shared by all paths in P_v (as a consequence of the König theorem). Make P_v a group of deferred paths with reserved edge e. We have $a_v = 0$ and $d_v = 1$. O can contain at most one path intersecting edge e: either a path from P_v or a path of smaller level. It suffices to remove this path from O in order to obtain a valid set O', and we get $|O| - |O'| \in \{0, 1\}$. Thus, $|O| - |O'| \leq (5/3 + \varepsilon)(a_v + d_v)$ holds, and the invariants are satisfied.

Case 1.1.2. $k = 1$, $\ell = m = 0$. There is one child c of v that has an undetermined path p with *lca* w in its subtree, possibly $w = c$. If $P_v = \emptyset$, the algorithm does nothing and leaves p in its undetermined state. If $P_v \neq \emptyset$, all paths in P_v must intersect p in the same edge, say, in the edge (u, w) with $w = p(u)$. (The case that they intersect p in an edge (w, u) is symmetrical.) The algorithm picks an arbitrary path q from P_v and makes $\{p, q\}$ a group of exclusive paths with fixed edge (u, w). (Hence, $X' = X \cup \{p, q\}$ and $U' = U \setminus \{p\}$.) All other paths in P_v are rejected, and we have $a_v = d_v = 0$. We must ensure that O' contains p or q in order to satisfy condition (b) of Invariant B. If O does not contain any path from $P_v \cup U_v$, by Property (E) either p or q can be inserted into O after removing at most one path of smaller level. If O contains a path p' from $P_v \cup U_v$ already, this path can be replaced by p or q if $p' \neq p, q$. We have $|O| \leq |O'| \leq |O| + 1$, and the invariants are satisfied.

Case 1.1.3. $k = m = 0$, $\ell = 1$. There is one child of v that has a group of exclusive paths in its subtree. As any path from P_v could be combined with a path from the group of exclusive paths to obtain two edge-disjoint paths and because we have assumed $s = 1$, we must have $P_v = \emptyset$. Hence, the algorithm does nothing at node v and leaves the group of exclusive paths in its intermediate state.

Case 1.2. $s = 2$. Observe that $k + \ell + 2m \leq s = 2$. In many of the subcases of Case 1.2, the algorithm will yield $a_v + d_v = 2$. If O contains at most one path from $P_v \cup U_v \cup X_v$, removing that path and at most two paths of smaller level is clearly sufficient to obtain a valid set O' in such subcases. Therefore, we do not repeat this argument in every relevant subcase; instead, we discuss only the case that O contains two paths from $P_v \cup U_v \cup X_v$.

Case 1.2.1. $m = 1$, $k = \ell = 0$. There is a subtree rooted at a child of v that contains a group of 2-exclusive paths. We must have $P_v = \emptyset$ because any path in P_v could be combined with two paths from X_v to form a set of three edge-disjoint paths. Hence, the algorithm does nothing at node v and leaves the group of 2-exclusive paths in its unresolved state.

Case 1.2.2. $\ell = 2$, $k = m = 0$. There are two children of v whose subtrees contain a group of exclusive paths. Note that $s = 2$ implies $P_v = \emptyset$ in this case, as any path from P_v could be combined with one exclusive path from each subtree to obtain a set of three edge-disjoint paths.

If the fixed edges of both groups of exclusive paths point in the same direction (i.e., are both directed to the root or to the leaves), the algorithm accepts the lower paths of both groups of exclusive paths. The higher paths are rejected, and no edge is marked fixed anymore. We have $a_v = 2$ and $d_v = 0$, and at most three paths must be removed from O to obtain a valid set O': the two paths from the groups of exclusive paths that are contained in O and at most one path of smaller level using the edge between v and $p(v)$ whose direction is opposite the direction of the formerly fixed edges.

If the fixed edges of the groups of exclusive paths point in different directions (i.e., one is directed towards the root and one towards the leaves), the groups represent a pair of independent groups of exclusive paths, and the algorithm can create a new group of 2-exclusive paths. Note that O contains two paths from the new group of 2-exclusive paths already because it contained one path from each of the two groups of exclusive paths in X_v due to condition (b) of Invariant B. Therefore, we can set $O' = O$, and the invariants are satisfied.

Case 1.2.3. $k = \ell = 1$, $m = 0$. There is one child of v that has a group of exclusive paths in its subtree and one child of v that has an undetermined path in its subtree.

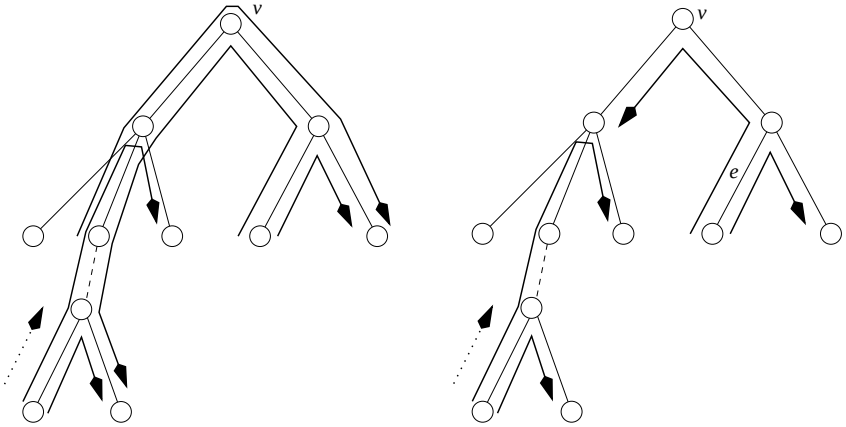


FIG. 3.5. *Case* 1.2.3.1: *P_v contains two edge-disjoint paths (left-hand side); Case* 1.2.3.2(a)*: The fixed edge and e have the same direction (right-hand side).*

All paths in P_v must intersect the undetermined path because otherwise a path from P_v could be combined with the undetermined path and an exclusive path to obtain a set of three edge-disjoint paths.

Case 1.2.3.1. There are two edge-disjoint paths in P_v. In this case, the situation must be as shown on the left-hand side of Figure 3.5: the two edge-disjoint paths from P_v must intersect the group of exclusive paths in a way that blocks all exclusive paths from being accepted, and there cannot be any other kinds of paths in P_v.

The algorithm accepts the lower path from the group of exclusive paths and the undetermined path, and it rejects all other paths in $P_v \cup X_v$. No edge is marked fixed anymore. We have $a_v = 2$ and $d_v = 0$. Note that any combination of two edge-disjoint paths from $P_v \cup U_v \cup X_v$ blocks at least three of the four top edges of the paths accepted by the algorithm. Hence, if O contains two paths from $P_v \cup U_v \cup X_v$, it can contain at most one path of smaller level intersecting the paths accepted by the algorithm, and it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.3.2. $P_v \neq \emptyset$ and all paths in P_v intersect the same edge e of the undetermined path.

Case 1.2.3.2(a). The direction of e is the same as that of the fixed edge of the group of exclusive paths (see the right-hand side of Figure 3.5). The algorithm accepts the undetermined path and the lower path from the group of exclusive paths. All other paths in $P_v \cup X_v$ are rejected, and no edge is marked fixed anymore. We have $a_v = 2$ and $d_v = 0$. If O contains two paths from $P_v \cup U_v \cup X_v$, these paths also use the fixed edge and the edge e, and at most one further path from O can be blocked by the paths accepted by the algorithm (because such a path must use the edge between v and $p(v)$ in the direction opposite the direction of e). Thus, it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.3.2(b). The direction of e is different from that of the fixed edge, and there is a path $p \in P_v$ that does not intersect the higher exclusive path (see the left-hand side of Figure 3.6). The algorithm uses X_v, p and the undetermined path together to create a new group of 2-exclusive paths consisting of a pair of independent groups of exclusive paths. All other paths in P_v are rejected by the algorithm. In addition to the fixed edge of the old group of exclusive paths, the edge e is marked

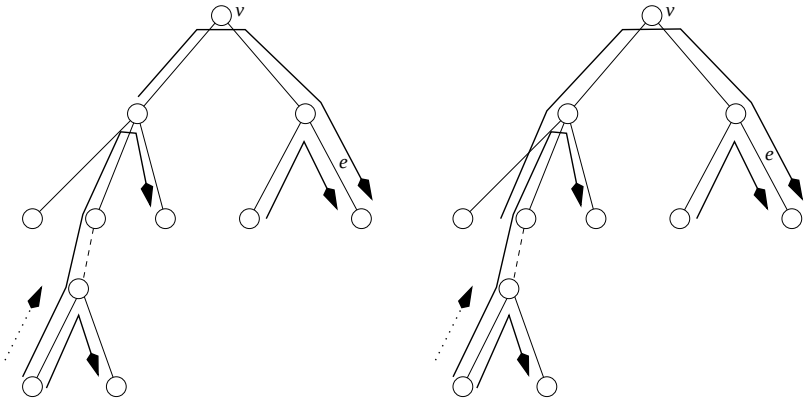


FIG. 3.6. *Cases* 1.2.3.2(b) *and* 1.2.3.2(c): *The fixed edge and e have different directions.*

fixed. Note that O contains one path from X_v due to condition (b) of Invariant B. If O contains the undetermined path or the path p, let $O' = O$. If O contains a path other than p from P_v, replace this path by p or by the undetermined path. (One of these must be possible.) If O does not contain a path from $P_v \cup U_v$ but contains a path p' using the edge between v and $p(v)$ in the direction given by edge e, replace p' either by p or by the undetermined path (one of the two must be possible). If O does not contain a path from $P_v \cup U_v$ and no path using the edge between v and $p(v)$ in the direction given by edge e, add either p or the undetermined path to O. In any case, the invariants are satisfied. In particular, $|O'| \geq |O|$.

Case 1.2.3.2(c). The direction of e is different from that of the fixed edge, and all paths in P_v intersect the higher exclusive path. (See the right-hand side of Figure 3.6.) The algorithm accepts the undetermined path and the lower path from the group of exclusive paths, and it rejects all other paths from $P_v \cup X_v$. No edge is marked fixed anymore. We have $a_v = 2$ and $d_v = 0$. If O contains two paths from $P_v \cup U_v \cup X_v$, it must contain at least one of the two paths accepted by the algorithm, and the other path in O uses a top edge of the other path accepted by the algorithm. O contains at most one path of smaller level intersecting the paths accepted by the algorithm, and it suffices to remove at most three paths from O in order to obtain a valid set O'.

Case 1.2.3.3. $P_v = \emptyset$. The algorithm accepts the undetermined path and the lower path from the group of exclusive paths, and it rejects all other paths in $P_v \cup X_v$. No edge is marked fixed anymore. We have $a_v = 2$ and $d_v = 0$. Any combination of two edge-disjoint paths from $P_v \cup U_v \cup X_v$ blocks at least three of the four top edges of the paths accepted by the algorithm. Hence, if O contains two paths from $P_v \cup U_v \cup X_v$, it can contain at most one path of smaller level intersecting the paths accepted by the algorithm, and it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.4. $\ell = 1$, $k = m = 0$. There is one child c of v that has a group of exclusive paths in its subtree. Denote the higher and the lower path in the group of exclusive paths by p and q, respectively. Assume without loss of generality that the fixed edge e' of the group of exclusive paths is directed towards the root of the tree (as shown in Figure 3.7). Note that $s = 2$ implies $P_v \neq \emptyset$. We distinguish further cases regarding the maximum number of edge-disjoint paths in P_v.

Case 1.2.4.1. There are two edge-disjoint paths p_1 and p_2 in P_v. As $s = 2$, p_1 and p_2 must intersect the exclusive paths in a way that blocks all of them from being

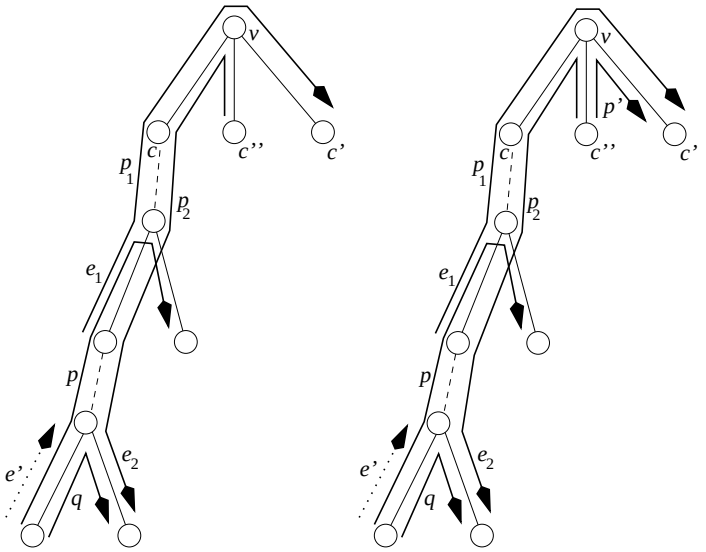


FIG. 3.7. *Case* 1.2.4.1: P_v *contains two edge-disjoint paths that block the exclusive paths.*

accepted. See Figure 3.7. Let p_1 intersect p, and let p_2 intersect q. Let $c' \neq c$ be the child of v such that p_1 uses the edges (c, v) and (v, c'), and let $c'' \neq c$ be the child of v such that p_2 uses the edges (c'', v) and (v, c). Note that $c' = c''$ is possible. Let the top edge of p intersected by p_1 be e_1, and let the top edge of q intersected by p_2 be e_2. As $P_v \cup X_v$ contains only two edge-disjoint paths, every path $p' \in P_v$ must intersect either edge e_1, or intersect edge e_2, or intersect both p_1 and p_2. (The latter case is possible only if $c' \neq c''$ and if all paths in P_v that intersect e_1 use the edges (c, v) and (v, c') and all paths in P_v that intersect e_2 use the edges (c'', v) and (v, c); in that case, p' must use (c'', v) and (v, c'), as shown on the right-hand side of Figure 3.7.)

Case 1.2.4.1(a). All paths in P_v that intersect e_1 use the edges (c, v) and (v, c'), and all paths in P_v that intersect e_2 use the edges (c'', v) and (v, c).

First, assume that all paths in P_v intersect either e_1 or e_2. Note that there are exactly two equivalence classes of paths in P_v in this case. See Figure 3.7 (left-hand side). The algorithm uses the group of exclusive paths and one representative from each of the two equivalence classes of paths in P_v to create a group of 2-exclusive paths. All other paths in P_v are rejected. The fixed edge e' of the group of exclusive paths is no longer marked fixed; instead, the edges e_1 and e_2 are marked fixed. If O contains two paths from $P_v \cup X_v$, one of them must be from X_v due to condition (b) of Invariant B, and the other can be replaced by a path in the new group of 2-exclusive paths. Otherwise, it is possible to remove the path from X_v and at most one additional path from O such that the resulting set contains no path from $P_v \cup X_v$, at most one path of smaller level touching v, and no path of smaller level intersecting a fixed edge of the new group of 2-exclusive paths. By Property (2E), two paths from the new group of 2-exclusive paths can then be inserted into that set to obtain O'. We have $|O'| \geq |O|$, and the invariants are satisfied.

Now, assume that there is a path $p' \in P_v$ that intersects neither e_1 nor e_2. As noted above, we must have $c' \neq c''$ in this case, and p' must use the edges (c'', v) and (v, c'). See Figure 3.7 (right-hand side). The algorithm accepts the lower path from the group of exclusive paths and the path p', and it rejects all other paths in

$P_v \cup X_v$. No edge is marked fixed anymore. We have $a_v = 2$ and $d_v = 0$. Note that any combination of two edge-disjoint paths from $P_v \cup X_v$ blocks at least three of the four top edges of the paths accepted by the algorithm. Hence, if O contains two paths from $P_v \cup X_v$, it can contain at most one path of smaller level intersecting the paths accepted by the algorithm, and it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.4.1(b). There are at least two equivalence classes of paths in P_v intersecting the higher path of the group of exclusive paths. The algorithm accepts the lower path of the group of exclusive paths and makes the paths in P_v intersecting the higher path a group of deferred paths. All other paths in $P_v \cup X_v$ are rejected, and no edge is marked fixed anymore. The reserved edge of the group of deferred paths is the top edge shared by all these paths. If O contains two paths from $P_v \cup X_v$, note that one of the two paths must be from X_v (due to condition (b) of Invariant B) and that these two paths also block the top edges of the lower path of the group of exclusive paths. Hence, O cannot contain any path of smaller level intersecting the lower path, and it can contain at most one path of smaller level intersecting the reserved edge of the newly deferred paths. It suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.4.1(c). There is only one equivalence class of paths in P_v intersecting the higher path of the group of exclusive paths, and there are at least two equivalence classes of paths in P_v intersecting the lower path of the group of exclusive paths. The algorithm accepts the higher path of the group of exclusive paths and makes the paths in P_v intersecting the lower path a group of deferred paths. All other paths in $P_v \cup X_v$ are rejected, and no edge is marked fixed anymore. The reserved edge of the group of deferred paths is the top edge shared by all these paths. If O contains two paths from $P_v \cup X_v$, note that one of the two paths must be from X_v (due to condition (b) of Invariant B) and that these two paths also block edge e_1. Hence, O cannot contain any path of smaller level intersecting e_1, and it can contain at most one path of smaller level intersecting the reserved edge of the newly deferred paths or the top edge of the higher path that is directed towards the leaves because all such paths must use the edge $(p(v), v)$. It suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.4.2. P_v does not contain two edge-disjoint paths. Let e be an edge incident to v such that all paths in P_v use edge e.

Case 1.2.4.2(a). $e = (v, c')$ for some $c' \neq c$, and P_v has at least two different equivalence classes of paths. The algorithm makes all paths in P_v a new group of deferred paths with reserved edge e and accepts q, the lower path of the group of exclusive paths. Path p is rejected, and no edge in this subtree is marked fixed anymore. We have $a_v = d_v = 1$. If O contains two paths from $P_v \cup X_v$, these paths block two of the three top edges blocked by the algorithm: the fixed edge e' of the group of exclusive paths and edge e. O can contain at most one path of smaller level that intersects the path accepted by the algorithm or the reserved edge of the new group of deferred paths, and it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.4.2(b). $e \neq (v, c')$ for all children $c' \neq c$ of v, or P_v has only one equivalence class of paths. If there is a path $p' \in P_v$ that does not intersect q, the algorithm accepts p' and q. If all paths in P_v intersect q, the algorithm accepts p and an arbitrary path from P_v. In both cases, all other paths in $P_v \cup X_v$ are rejected, and no edge in this subtree is marked fixed anymore. We have $d_v = 0$ and $a_v = 2$. Assume

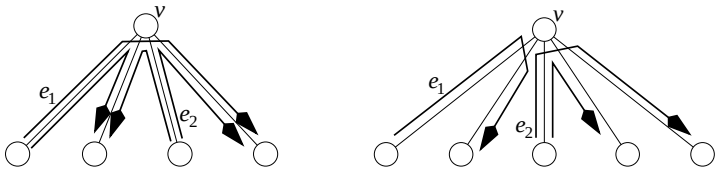


FIG. 3.8. *Case* 1.2.5(a): *All sets of two edge-disjoint paths use the same four top edges (left-hand side); Case* 1.2.5(b): *there is only one equivalence class of paths using edge* e_1 *but more than one class using edge* e_2 *(right-hand side).*

that O contains two paths from $P_v \cup X_v$. We will show that it suffices to remove at most three paths from O to obtain a valid set O'.

If the algorithm has accepted p, O must also contain p and a path from P_v, thus blocking at least three of the four top edges of the paths accepted by the algorithm. At most one further path in O can be blocked by the paths accepted by the algorithm.

Now assume that the algorithm has accepted q. Observe that the two paths from $P_v \cup X_v$ that are in O must also use the edges e' and e, thus blocking two of the four top edges of paths accepted by the algorithm. If e and e' have the same direction, O can contain at most one path of smaller level intersecting the paths accepted by the algorithm because such a path must use the edge $(p(v), v)$. If P_v has only one equivalence class of paths, the paths from $P_v \cup X_v$ that are in O block three of the four top edges of paths accepted by the algorithm, and again it suffices to remove at most one path of smaller level from O. Finally, consider the case that P_v has more than one equivalence class of paths and that $e = (v, c)$. Since edge e blocks more paths of smaller level than the top edge of q that is directed towards the leaves, the two paths from $P_v \cup X_v$ that are in O do in fact block at least as many paths of smaller level as three of the four top edges of the paths accepted by the algorithm.

Case 1.2.5. $k = \ell = m = 0$. As $s = 2$, there must be two edges incident to v such that all paths in P_v use at least one of these two edges (by the König theorem). Let e_1 and e_2 be two such edges.

Case 1.2.5(a). All possible sets of two edge-disjoint paths from P_v use the same four edges incident to v. See the left-hand side of Figure 3.8 for an example. The algorithm picks two arbitrary edge-disjoint paths from P_v, accepts them, and rejects all other paths from P_v. We have $a_v = 2$ and $d_v = 0$. If O contains two paths from P_v, removing these two paths is sufficient to obtain a valid set O' because they use the same top edges as the paths accepted by the algorithm, and O cannot contain any further path intersecting the paths accepted by the algorithm.

In the following, let D be the set of paths in P_v that intersect all other paths from P_v. In other words, a path $p \in P_v$ is in D if P_v does not contain a path q that is edge-disjoint from p. Note that if Case 1.2.5(a) does not apply, it follows that either the paths in $P_v \setminus D$ using edge e_1 or those using edge e_2 must have more than one equivalence class of paths.

Case 1.2.5(b). There is only one equivalence class C of paths in $P_v \setminus D$ using edge e_1 but more than one equivalence class of paths in $P_v \setminus D$ using edge e_2 and not intersecting a path from C. See the right-hand side of Figure 3.8. (The case with e_1 and e_2 exchanged is symmetrical. Furthermore, note that the case that there is only one equivalence class C of paths in $P_v \setminus D$ using edge e_1 and only one equivalence class of paths in $P_v \setminus D$ using edge e_2 and not intersecting a path from C satisfies the condition of Case 1.2.5(a).) The algorithm picks a path p from C arbitrarily, accepts p,

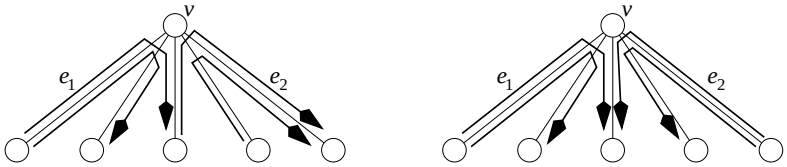


FIG. 3.9. *Case* 1.2.5(c): *Configurations in which two groups of deferred paths can be created.*

and makes the paths using edge e_2 and not intersecting p a group of deferred paths with reserved edge e_2. All other paths in P_v are rejected. We have $a_v = 1$ and $d_v = 1$. If O contains two paths from P_v, these paths must also use both top edges of p and the newly reserved edge, and thus removing these two paths from O is sufficient to obtain a valid set O'.

Case 1.2.5(c). There is more than one equivalence class of paths in $P_v \setminus D$ using edge e_1, there is more than one equivalence class of paths in $P_v \setminus D$ using edge e_2, and Case 1.2.5(a) does not apply. See Figure 3.9. The algorithm makes the paths in $P_v \setminus D$ using e_1 a group of deferred paths with reserved edge e_1 and the paths in $P_v \setminus D$ using e_2 a group of deferred paths with reserved edge e_2. All other paths in P_v are rejected. Note that no matter which paths of smaller level are accepted by the algorithm later on, there are still two paths, one in each of the two groups of newly deferred paths, that are edge-disjoint from these paths of smaller level and from each other. (Otherwise, Case 1.2.5(a) would apply.) We have $a_v = 0$ and $d_v = 2$. If O contains two paths from P_v, these paths use e_1 and e_2 as well, and removing these two paths from O is sufficient to obtain a valid set O' because O cannot contain any further path intersecting a reserved edge of the newly deferred paths.

Case 1.2.6. $k = 1$, $\ell = m = 0$. There is one child of v that has an undetermined path p in its subtree. Let $P'_v \subseteq P_v$ denote the set of paths in P_v that do not intersect p. We begin by making some simple observations. First, P'_v must not contain two edge-disjoint paths. Hence, there must be an edge e incident to v that is shared by all paths in P'_v. Second, $s = 2$ implies that the maximum number of edge-disjoint paths in P_v is at most two.

Let the *lca* of the undetermined path be v', and let c be the child of v whose subtree contains the undetermined path (possibly $c = v'$). Let v_1 and v_2 be children of v' such that the undetermined path uses the edges (v_1, v') and (v', v_2). We consider a number of subcases regarding the number of equivalence classes in P'_v.

Case 1.2.6(a). P'_v is empty. Let P_1 and P_2 denote the sets of paths in P_v that intersect p in the edge (v_1, v') and in the edge (v', v_2), respectively. Note that $P_v = P_1 \cup P_2$ and that $P_1 \neq \emptyset \neq P_2$. For $i = 1, 2$, the algorithm accepts an arbitrary path from P_i if P_i has only one equivalence class of paths and creates a new group of deferred paths from P_i otherwise. The undetermined path p is rejected. We have $a_v + d_v = 2$. If O contains two paths from $P_v \cup U_v$, removing these two paths is sufficient because they block at least as many paths of smaller level as the newly accepted paths or newly reserved edges.

Case 1.2.6(b). P'_v has one equivalence class of paths. The algorithm accepts an arbitrary path from P'_v and the undetermined path p. All other paths in P_v are rejected. We have $a_v = 2$ and $d_v = 0$. Assume that O contains two paths from $P_v \cup U_v$. If O contains p, O must also contain a path from P'_v, and it suffices to remove these two paths from O to obtain a valid set O'. If O does not contain p but contains a path from P'_v, O must also contain a path from P_v that intersects p; these two paths

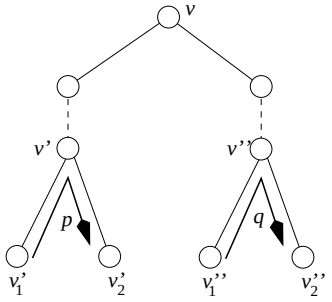


FIG. 3.10. *Case* 1.2.7: *v has two children with undetermined paths in their subtrees.*

block at least three of the four top edges blocked by the algorithm, and it suffices to remove these two paths and at most one path of smaller level. Finally, if O contains neither p nor a path from P'_v, O must contain two paths from P_v that intersect p in different top edges and at least one of which intersects also a top edge of the paths in P'_v; again, it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.6(c). P'_v has more than one equivalence class of paths. Let e be the edge incident to v that is shared by all paths in P'_v. The algorithm accepts the undetermined path p and creates a new group of deferred paths from the paths in P'_v. All other paths in P_v are rejected. We have $a_v = d_v = 1$. Assume that O contains two paths from $P_v \cup U_v$. If O contains p, O must also contain a path from P'_v, and it suffices to remove these two paths from O to obtain a valid set O'. If O does not contain p but contains a path from P'_v, O must contain a path from P_v that intersects p; these two paths block at least two of the three top edges blocked by the algorithm, and it suffices to remove these two paths and at most one path of smaller level. Finally, if O contains neither p nor a path from P'_v, O must contain two paths from P_v that intersect p in different top edges; again, these two paths block at least two of the three top edges blocked by the algorithm, and it suffices to remove at most three paths from O to obtain a valid set O'.

Case 1.2.7. $k = 2$, $\ell = m = 0$. Two children of v have undetermined paths in their subtrees. Denote the undetermined paths by p and q. See Figure 3.10. As $s = 2$, every path in P_v must intersect at least one undetermined path. In addition, if there are two paths in P_v that intersect one undetermined path in different top edges, at least one of them must also intersect the other undetermined path. Let P_1 and P_2 denote the sets of paths in P_v that intersect p and q, respectively. Note that $P_1 \cap P_2 \neq \emptyset$ is possible.

Case 1.2.7(a). There are edge-disjoint paths p_1 and p_2 in P_v such that p_1 intersects p in a top edge e_1 but does not intersect q, and p_2 intersects q in a top edge e_2 but does not intersect p, and such that e_1 and e_2 have different directions (i.e., one is directed towards the root, and the other is directed towards the leaves). The algorithm makes p, q, p_1, and p_2 a group of 2-exclusive paths consisting of a pair of independent groups of exclusive paths and rejects all other paths from P_v. The edges e_1 and e_2 are marked fixed. If O contains two paths from the new group of 2-exclusive paths already, let $O' = O$. Otherwise, it is possible to replace paths in O by paths from the new group of 2-exclusive paths to obtain O'. In any case, $|O'| \geq |O|$.

Case 1.2.7(b). If the condition for Case 1.2.7(a) does not hold, the algorithm accepts p and q and rejects all paths from P_v. We have $a_v = 2$ and $d_v = 0$. Assume that O contains two paths from $P_v \cup U_v$. If O contains p and q, it suffices to remove

these two paths. If O contains only one of p and q, say, p, it must contain a path from P_v that intersects q, and these two paths block three of the four top edges blocked by the algorithm. If O contains neither p nor q, it must contain two paths from P_v. If at least one of these two paths in O intersects both p and q, these two paths again block at least three of the four top edges blocked by the algorithm. If both paths in O intersect only one of p and q, it must be the case that one of them intersects p in an edge e_1 and one of them intersects q in an edge e_2. If e_1 and e_2 have the same direction, O can contain at most one path of smaller level intersecting a path accepted by the algorithm. If e_1 and e_2 have different directions, the condition of Case 1.2.7(a) applies.

Case 1.3. $s \geq 3$. The algorithm accepts the s paths and rejects all other paths from $P_v \cup U_v \cup X_v$. No edge in this subtree is marked fixed anymore. As s is the maximum number of edge-disjoint paths in $P_v \cup U_v \cup X_v$, O can contain at most s paths from $P_v \cup U_v \cup X_v$. Furthermore, O can contain at most two paths from F using the edges $(v, p(v))$ or $(p(v), v)$, and these are the only two further paths in O that could possibly be blocked by the s paths accepted by the algorithm. Hence, a valid set O' can be obtained from O by deleting at most $s + 2$ paths. As $s + 2 \leq (5/3)s$, the invariants are maintained.

3.4.2. Case 2: Many subtrees with unresolved paths. Case 2 applies if $k + \ell + 2m > \frac{2}{\varepsilon}$. In this case, the algorithm cannot try out all possibilities of accepting or rejecting unresolved paths in polynomial time. Instead, it computes four candidate sets of edge-disjoint paths in $P_v \cup U_v \cup X_v$ and accepts the paths in the largest of these four sets. Unlike for Case 1, the subcases of Case 2 do not correspond to different actions taken by the algorithm. Instead, the subcases apply to different ranges of the ratio between $s_3 - s_1$ and k, where s_1 is the maximum number of edge-disjoint paths in P_v not intersecting a path in U_v and s_3 is the maximum number of edge-disjoint paths in P_v without any restriction. In each of the subcases, a lower bound on the number of paths accepted by the algorithm and an upper bound on the optimal number of edge-disjoint paths in $P_v \cup U_v \cup X_v$ are derived to show that the invariant is satisfied. The detailed analysis of Case 2 is as follows.

Case 2. $k + \ell + 2m > \frac{2}{\varepsilon}$. The algorithm calculates four candidate sets of edge-disjoint paths from $P_v \cup U_v \cup X_v$ and chooses the largest of them. For obtaining two of the four sets, we employ a method of removing paths from an arbitrary set S of edge-disjoint paths in P_v such that $\ell + 2m$ exclusive paths from X_v can be accepted in addition to the paths remaining in S. The resulting set of edge-disjoint paths in $S \cup X_v$ has cardinality $|S| + \ell + 2m - r$, where r is the number of paths that were removed from S. The details of the method and a proof that $r \leq (|S| + \ell + m)/3$ will be presented later in Lemma 3.1. With this tool we are ready to describe the candidate sets S_1, S_2, S_3, and S_4. Let $P'_v \subseteq P_v$ be the subset of paths in P_v that do not intersect any undetermined path in U_v.

1. Compute a maximum number s_1 of edge-disjoint paths in P'_v. S_1 is obtained by taking these paths, all k undetermined paths, and as many additional edge-disjoint paths from X_v as possible. We have $|S_1| \geq k + s_1 + m$ because S_1 contains k undetermined paths and at least m paths from groups of 2-exclusive paths in X_v due to Property (2E).

2. S_2 is obtained from S_1 by removing r of the s_1 paths in $S_1 \cap P_v$ from S_1 such that $\ell + 2m$ exclusive paths can be accepted. S_2 contains $\ell + 2m$ exclusive paths, and according to Lemma 3.1 only $r \leq (s_1 + \ell + m)/3$ of the s_1 paths in $S_1 \cap P_v$ were removed to obtain S_2. As S_2 still contains the k undetermined paths, we have

$|S_2| \geq k+m+(2/3)(s_1+\ell+m)$. In addition, we have $|S_2| \geq k+\ell+2m \geq \frac{2}{\varepsilon}$ because S_2 contains all k undetermined paths from U_v and $\ell+2m$ exclusive paths.

3. S_3 is obtained by first computing a maximum number s_3 of edge-disjoint paths in P_v and then adding as many edge-disjoint paths from $X_v \cup U_v$ as possible. We have $|S_3| \geq s_3+m$ because S_3 contains at least m paths from groups of 2-exclusive paths in X_v due to Property (2E).

4. S_4 is obtained from S_3 by removing r of the s_3 paths in $S_3 \cap P_v$ from S_3 such that $\ell+2m$ exclusive paths can be accepted, in the same way as S_2 is obtained from S_1. Since $r \leq (s_3+\ell+m)/3$ according to Lemma 3.1, we have $|S_4| \geq m+(2/3)(s_3+\ell+m)$.

The algorithm accepts the paths in that set S_i with maximum cardinality and rejects all other paths from $P_v \cup U_v \cup X_v$. We have $a_v = \max\{|S_1|, |S_2|, |S_3|, |S_4|\}$ and $d_v = 0$. Note that $a_v \geq |S_2| \geq \frac{2}{\varepsilon}$ and that this implies $2 \leq \varepsilon a_v$.

Let $O_v = O \cap (P_v \cup U_v \cup X_v)$, and let b' be the number of paths from P_v that are contained in O_v and that intersect at least one of the k undetermined paths. Observe that O_v can contain at most $k - b'/2$ undetermined paths from U_v. Note that the maximum number of edge-disjoint paths in P_v is s_3 and that the maximum number of edge-disjoint paths in P'_v is s_1. Using $|O_v| \leq s_3 + k - b'/2 + \ell + 2m$ if $b' \geq s_3 - s_1$ and using $|O_v| \leq s_1 + b' + k - b'/2 + \ell + 2m$ if $b' < s_3 - s_1$, we get

$$|O_v| \leq s_1 + (s_3 - s_1)/2 + k + \ell + 2m. \tag{3.1}$$

With this upper bound on $|O_v|$ and the lower bounds on the cardinalities of the four sets S_i, we can now prove that at least one of the sets S_i satisfies $|O_v|+2 \leq (5/3+\varepsilon)|S_i|$. Since it suffices to remove at most $|O_v|+2$ paths from O in order to obtain a valid set O', this implies that the invariants are maintained.

If $k = 0$, we have $|O_v| \leq s_3 + \ell + 2m$ and $a_v \geq |S_4| \geq (2/3)(s_3 + \ell + 2m)$. This implies $|O_v| + 2 \leq (3/2)a_v + 2 \leq (3/2 + \varepsilon)a_v$. If $k > 0$, let $\alpha = (s_3 - s_1)/k$ and distinguish the following cases.

Case 2.1. $\alpha > 3/2$. If $\ell + 2m \leq (\alpha/2)k$, (3.1) gives $|O_v| \leq s_1 + (1+\alpha)k$, and we have $a_v \geq |S_3| \geq s_3 + m \geq s_1 + \alpha k$. We obtain $|O_v| + 2 \leq a_v(1+\alpha)/\alpha + 2 \leq (5/3)a_v + 2 \leq (5/3+\varepsilon)a_v$. If $\ell + 2m \geq (\alpha/2)k$, we use (3.1) and $a_v \geq |S_4|$ to bound the ratio between $|O_v| + 2$ and a_v as follows:

$$\begin{aligned}
\frac{|O_v|+2}{a_v} &\leq \frac{s_1 + (1+\alpha/2)k + \ell + 2m}{(2/3)s_1 + (2\alpha/3)k + (2/3)\ell + (5/3)m} + \frac{2}{a_v} \\
&\leq \frac{3}{2} + \frac{(1-\alpha/2)k}{(2/3)s_1 + (2\alpha/3)k + (2/3)\ell + (5/3)m} + \varepsilon \\
&\leq \frac{3}{2} + \frac{(1-\alpha/2)k}{(2\alpha/3)k + (\alpha/3)k} + \varepsilon \leq 1 + \frac{1}{\alpha} + \varepsilon \leq \frac{5}{3} + \varepsilon.
\end{aligned}$$

Case 2.2. $4/3 < \alpha \leq 3/2$. If $\ell + 2m \leq (3/2)(\alpha - 1)k$, (3.1) gives $|O_v| \leq s_1 + (2\alpha - 1/2)k$, and we have $a_v \geq |S_3| \geq s_3 + m \geq s_1 + \alpha k$. We obtain $|O_v| + 2 \leq a_v(2\alpha - 1/2)/\alpha + 2 \leq (5/3)a_v + 2 \leq (5/3 + \varepsilon)a_v$. If $\ell + 2m \geq (3/2)(\alpha - 1)k$, we use (3.1) and $a_v \geq |S_2|$ to bound the ratio between $|O_v| + 2$ and a_v as follows:

$$\begin{aligned}
\frac{|O_v|+2}{a_v} &\leq \frac{s_1 + (1+\alpha/2)k + \ell + 2m}{(2/3)s_1 + k + (2/3)\ell + (5/3)m} + \frac{2}{a_v} \\
&\leq \frac{3}{2} + \frac{(\alpha-1)k/2}{(2/3)s_1 + k + (2/3)\ell + (5/3)m} + \varepsilon \\
&\leq \frac{3}{2} + \frac{(\alpha-1)k/2}{k + (\alpha-1)k} + \varepsilon \leq 2 - \frac{1}{2\alpha} + \varepsilon \leq \frac{5}{3} + \varepsilon.
\end{aligned}$$

Case 2.3. $\alpha \leq 4/3$. From (3.1) we get $|O_v| \leq s_1 + (5/3)k + \ell + 2m$, and we have $a_v \geq |S_2| \geq (2/3)s_1 + k + (2/3)\ell + (5/3)m$. We obtain $|O_v| + 2 \leq (5/3)a_v + 2 \leq (5/3 + \varepsilon)a_v$.

We have shown that $|O_v| + 2 \leq (5/3 + \varepsilon)a_v$ holds in all subcases of Case 2. To complete the description of Case 2, we still have to explain the method for removing paths from S_1 and S_3 in order to obtain S_2 and S_4, respectively. The method takes an arbitrary set S of edge-disjoint paths in P_v and removes paths from S to obtain a set S' such that every subtree with exclusive paths is touched by at most one path in S'. The motivation for this is that S can cause all paths from a group of exclusive paths to be blocked only if two paths from S intersect the corresponding subtree (Property (E)). Similarly, if only one path from a group of 2-exclusive paths can be accepted, S must contain two paths from P_v that intersect the corresponding subtree (Property (2E)).

The method proceeds as follows. Consider a graph G with the paths in S as its vertices and an edge between two paths if they touch the same child of v. G has maximum degree two and consists of a collection of chains and cycles. Note that every edge of G corresponds to a child of v that is touched by two paths in S. We are interested in the maximal parts of chains and cycles that consist entirely of edges corresponding to children of v that are the roots of subtrees with exclusive paths. There are the following possibilities for such parts:

(i) a cycle such that all paths on the cycle have both endpoints in a subtree with exclusive paths;

(ii) a chain such that the paths at both ends have only one endpoint in a subtree with exclusive paths, while the internal paths have both endpoints in subtrees with exclusive paths;

(iii) a chain such that the path at one end has only one endpoint in a subtree with exclusive paths, while all other paths have both endpoints in a subtree with exclusive paths;

(iv) a chain such that all its paths have both endpoints in a subtree with exclusive paths.

Note that every such maximal part of a cycle or chain has length (number of paths) at least two because it contains at least one edge. The method for removing paths proceeds as follows. Cycles of even length and chains are handled by removing every other path from S, starting with the second path for chains. Cycles of odd length are handled by removing two consecutive paths in one place and every other path from the rest of the cycle.

Consider the example depicted in Figure 3.11. The node v has eight children, named a to h, and six of them (c to h) are roots of subtrees with exclusive paths (indicated by an exclamation mark). A set S of edge-disjoint paths in P_v is sketched. The graph G obtained from this set is shown in Figure 3.12, and the label of a vertex in G is u-w if the corresponding path begins in the subtree rooted at u and ends in the subtree rooted at w. With respect to (i)–(iv) above, G contains a cycle of type (i) with length three (containing the paths f-g, g-h, and h-f) and a chain of type (ii) with length three (containing the paths a-d, d-c, and c-b). According to the rules given above, three paths would be removed from S: two paths, say, f-g and g-h, from the cycle, and the path d-c from the chain of length three.

It is easy to see that this process always ensures that in the end S contains, for each subtree with exclusive paths, at most one path with an endpoint in that subtree. Hence, due to Properties (E) and (2E), S can be filled up with edge-disjoint exclusive

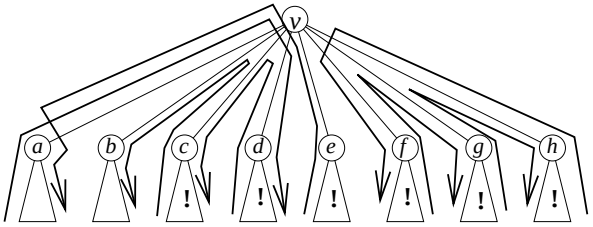


FIG. 3.11. *Set of edge-disjoint paths in P_v.*

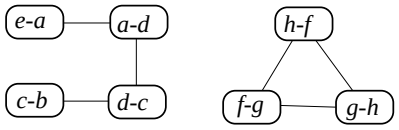


FIG. 3.12. *Graph G representing the structure of the paths.*

paths until it contains all $\ell + 2m$ exclusive paths.

LEMMA 3.1. *Let v be a node with $\ell+m$ children with exclusive paths. Let $S \subseteq P_v$ be a set of edge-disjoint paths. Let $S' \subseteq S$ be the set of paths obtained from S by removing paths according to the method described above. Let $|S| = s$ and $|S \setminus S'| = r$. Then $r \leq (s + \ell + m)/3$.*

Proof. Let a be the number of cycles of type (i), and let a_i, $1 \leq i \leq a$, be the length of the ith cycle. Denote the number of chains of type (ii) by b and their lengths by b_i, $1 \leq i \leq b$. Denote the number of chains of type (iii) by c and their lengths by c_i, $1 \leq i \leq c$. Denote the number of chains of type (iv) by d and their lengths by d_i, $1 \leq i \leq d$. Note that $a_i, b_i, c_i, d_i \geq 2$ for all i. As the number of paths contained in the union of all these chains and cycles is at most s, we have $\sum_{i=1}^{a} a_i + \sum_{i=1}^{b} b_i + \sum_{i=1}^{c} c_i + \sum_{i=1}^{d} d_i \leq s$. Furthermore, considering the number of children with exclusive paths covered by each chain or cycle, we obtain $\sum_{i=1}^{a} a_i + \sum_{i=1}^{b}(b_i-1)+\sum_{i=1}^{c} c_i+\sum_{i=1}^{d}(d_i+1) \leq \ell+m$. Bounding $b_i-1 \geq b_i/2$, $c_i \geq c_i/2$, and $d_i + 1 \geq d_i/2$ in the latter inequality and adding up the two inequalities, we obtain $\sum_{i=1}^{a} 2a_i+\sum_{i=1}^{b}(3/2)b_i+\sum_{i=1}^{c}(3/2)c_i+\sum_{i=1}^{d}(3/2)d_i \leq s+\ell+m$. Taking into account that $r = \sum_{i=1}^{a} \lceil \frac{a_i}{2} \rceil + \sum_{i=1}^{b} \lfloor \frac{b_i}{2} \rfloor + \sum_{i=1}^{c} \lfloor \frac{c_i}{2} \rfloor + \sum_{i=1}^{d} \lfloor \frac{d_i}{2} \rfloor$ and that $\lceil a_i/2 \rceil \leq (2/3)a_i$ for $a_i \geq 2$, the lemma follows. □

In the example displayed in Figure 3.11, we had $s = 7$ and $\ell + m = 6$, and it was sufficient to remove $r = 3$ paths. Indeed, $r \leq (s + \ell + m)/3 = 4\frac{1}{3}$.

3.5. Running-time of the algorithm. The running-time of our algorithm is polynomial in the size of the input for fixed $\varepsilon > 0$ but exponential in $1/\varepsilon$. Let a bidirected tree $T = (V, E)$ with n nodes and a set P containing h directed paths in T (each path specified by its endpoints) be given. For arbitrary $\varepsilon > 0$, we show below that our approximation algorithm can be implemented to run in time

$$O\left(n + h + 4^{1/\varepsilon} \min\left\{\sqrt{n} \cdot h, h^{1.5}\right\}\right).$$

Note that we can choose $\varepsilon = 1/\log n$ and still achieve running-time polynomial in

the size of the input. The resulting algorithm has approximation ratio at most $5/3 + 1/\log n$ and, therefore, asymptotic approximation ratio 5/3: if the optimal solution contains many paths, n must also be large, and the approximation ratio gets arbitrarily close to 5/3.

We have implemented the algorithm and evaluated it experimentally [10]. On instances where the requests were generated by choosing source and destination uniformly at random, the algorithm, with ε set to 1/10, usually produced solutions that were very close to the optimal solution (within less than 1 percent). For a detailed discussion of the implementation and the experimental results, we refer to [10].

We sketch how the claimed running-time is achieved. After an $O(n)$ preprocessing step similar to the *lca* algorithm of [29], we can compute for every path $p \in P$ its *lca* and its top edges in time $O(h)$ in total. For every node v of T, we compute a list L_v of all paths p with $lca(p) = v$. This takes time $O(h)$.

When node v is processed in the first pass, P_v is computed from L_v by checking for every path in L_v whether it is blocked by an accepted path or by a fixed or reserved edge and discarding it if this is the case. In order to perform each of these checks in constant time, a special edge-marking scheme is employed: When a path p is accepted, we mark the top edges of p, all unmarked upward edges in the subtree below the upward top edge of p, and all unmarked downward edges in the subtree below the downward top edge of p. When an edge becomes fixed or reserved, the edges in the subtree below the edge are marked similarly. Since each edge is marked only once, the total time spent in marking is $O(n)$. With this marking scheme, a path in L_v is blocked by a previously accepted path or a fixed or reserved edge if and only if its first or last edge is marked.

The remaining processing of v is dominated by the time for computing maximum matchings, which has to be done $4^{1/\varepsilon}$ times if Case 1 applies and four times (to compute S_1, S_2, S_3, and S_4) if Case 2 applies. A maximum matching in a bipartite graph with n_1 vertices and m_1 edges can be computed in time $O(\sqrt{n_1}m_1)$ [19]. Therefore, the time for computing a maximum matching in the bipartite graph constructed from (a subset) of P_v can be bounded by $O(\sqrt{\min\{\deg_v, |P_v|\}} \cdot |P_v|)$, where $\deg_v$ is the degree of v in T. To see this, note that the bipartite graph has at most $|P_v|$ edges and that the number of nonisolated vertices is bounded by $2|P_v|$ and by $2\deg_v$. Therefore, the processing of vertex v in the first pass (excluding the time for edge marking, which is accounted separately) takes time at most $O(\deg_v + |L_v| + 4^{1/\varepsilon}\sqrt{\min\{\deg_v, |L_v|\}} \cdot |L_v|)$. Summing this over all vertices v, we obtain the bound $O(n + h + 4^{1/\varepsilon} \min\{\sqrt{n} \cdot h, h^{1.5}\})$. Finally, the second pass can easily be implemented to run in time $O(n + h)$.

4. Generalizations. There are several generalizations of MEDP. First, it is meaningful to consider the *weighted* version of the problem, where each path has a certain weight and the goal is to maximize the total weight of the accepted paths. The weighted version of MEDP can still be solved optimally in polynomial time in bidirected stars and spiders (by reduction to maximum-weight matching in a bipartite graph) and in bidirected trees of bounded degree (by a minor modification of the dynamic programming procedure given in section 2).

We do not know how to generalize the $(5/3 + \varepsilon)$-approximation algorithm for MEDP from section 3 to the weighted case. However, we have obtained a $(5/3 + \varepsilon)$-approximation algorithm for the weighted version of MEDP in bidirected trees using a completely different approach in [9]. This algorithm is based on linear programming and converts the fractional optimum solution of a linear programming relaxation into

an approximate integral solution using the path coloring algorithm from [12, 13] as a subroutine.

Another generalization of MEDP is the *maximum path coloring* (MaxPC) problem. For a given bidirected tree $T = (V, E)$, set P of directed paths in T, and number W of colors, the MaxPC problem is to compute a subset $P' \subseteq P$ and a W-coloring of P'. The goal is to maximize the cardinality of P'. The MaxPC problem is equivalent to finding a maximum (induced) W-colorable subgraph in the conflict graph of the given paths. Studying MaxPC is motivated by the admission control problem in all-optical wavelength-division multiplexing (WDM) networks without wavelength converters: every wavelength (color) can be used to establish a set of connections provided that the paths corresponding to the connections are edge-disjoint, and the number of available wavelengths is limited [7]. The weighted variant of MaxPC is interesting as well.

Both MaxPC and weighted MaxPC can be solved optimally in polynomial time for bidirected stars by using an algorithm for (the weighted version of) the capacitated b-matching problem. Given an undirected graph $G = (V, E)$ and a function $b : V \to \mathbb{N}_0$, a b-matching is a function $x : E \to \mathbb{N}_0$ such that $\sum_{e \in \Gamma(v)} x(e) \leq b(v)$ for every vertex $v \in V$, where $\Gamma(v)$ is the set of edges incident to v. There exists a polynomial-time algorithm for computing a b-matching of maximum weight, i.e., a b-matching for which $\sum_{e \in E} w(e)x(e)$ is maximum, where $w : E \to \mathbb{Q}$ is an arbitrary weight function. If the multiplicities with which edges can be included into the b-matching are bounded by edge capacities $c(e)$ (i.e., if we require $0 \leq x(e) \leq c(e)$ for all $e \in E$), the resulting problem is called the capacitated b-matching problem and can also be solved in polynomial time [17, pp. 257–259]. In order to solve the weighted MaxPC problem with W colors for a given set of weighted paths in a star, transform the paths into a bipartite multigraph as in the case of MEDP (cf. Figure 2.2) and then compute a capacitated b-matching of maximum weight, where we set $b(v) = W$ for all vertices v and $c(e) = 1$ for all edges e of the bipartite multigraph. This b-matching corresponds to a maximum-weight subset of the given paths such that at most W paths in the subset use the same edge. Since any set of paths with maximum load W in a bidirected star can be colored efficiently with W colors (by edge-coloring the corresponding bipartite multigraph), this subset is an optimal solution to the given MaxPC problem.

If both the number W of colors and the maximum degree of the bidirected tree are bounded by constants, MaxPC and weighted MaxPC can be solved optimally in polynomial time by dynamic programming (similar to the procedure in section 2). MaxPC is $\mathcal{NP}$-hard for arbitrary W in bidirected binary trees (because path coloring is $\mathcal{NP}$-hard) and for $W = 1$ in bidirected trees of arbitrary degree (because it is equivalent to MEDP in this case).

In order to obtain approximation algorithms for MaxPC with arbitrary number W of colors, a technique due to Awerbuch et al. [3] can be employed. It allows reducing the problem with W colors to MEDP with only a small increase in the approximation ratio. The technique works for MaxPC in arbitrary graphs G; we discuss it here only for trees. Let an instance of MaxPC be given by a bidirected tree $T = (V, E)$, a set P of paths in T, and a number W of colors. An approximation algorithm A for arbitrary number W of colors is obtained from an approximation algorithm A_1 for one color (i.e., for the maximum edge-disjoint paths problem) by running W copies of A_1, giving as input to the ith copy the bidirected tree T and the set of paths that have not been accepted by the first $i - 1$ copies of A_1 (see Figure 4.1). The output

```
Algorithm A
Input: bidirected tree T, set P of paths, number W of colors
Output: disjoint subsets P_1,...,P_W of P (each P_i is edge-disjoint)
begin
for i = 1 to W do
    begin
    P_i ← A_1(T, P);
    P ← P \ P_i;
    end
end
```

FIG. 4.1. *Reduction from many colors to one color.*

of A is the union of the W sets of paths output by the copies of A_1, and the paths in the ith set are assigned color i.

In [3] it is shown that Algorithm A obtained using this technique has approximation ratio at most $\rho + 1$ if A_1 has approximation ratio ρ, even if different colors are associated with different network topologies. For identical networks, which we have in our application, the approximation ratio achieved by A can even be bounded by $1/(1-(1-\frac{1}{\rho W})^W)$, which is smaller than $1/(1-\mathrm{e}^{-1/\rho})$ for all W. It is remarked in [3] that this bound can be viewed as an adaptation of a similar result in [8]. A formal proof is given in [31]. This reduction works also in the weighted case.

Since we have an optimal algorithm for MEDP in bidirected trees of bounded degree and $(5/3+\varepsilon)$-approximation algorithms for MEDP in arbitrary bidirected trees, we can employ the above technique and obtain approximation algorithms with ratio $1/(1-1/e) \approx 1.58$ for MaxPC in bidirected trees of bounded degree and with ratio approximately 2.22 for MaxPC in arbitrary bidirected trees.

Acknowledgments. The authors are grateful to Stefano Leonardi for pointing out the reduction from MaxPC with an arbitrary number of colors to MaxPC with one color, and to Adi Rosén for informing them about the improved analysis for the ratio obtained by this reduction in the case of identical networks for all colors and for supplying a preliminary version of [3].

REFERENCES

[1] V. Auletta, I. Caragiannis, C. Kaklamanis, and P. Persiano, *Randomized path coloring on binary trees*, in Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2000), Lecture Notes in Comput. Sci. 1913, Springer-Verlag, Berlin, 2000, pp. 60–71.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and Approximation. Combinatorial Optimization Problems and Their Approximability Properties*, Springer-Verlag, Berlin, 1999.

[3] B. Awerbuch, Y. Azar, A. Fiat, S. Leonardi, and A. Rosén, *On-line competitive algorithms for call admission in optical networks*, Algorithmica, 31 (2001), pp. 29–43.

[4] B. Awerbuch, Y. Bartal, A. Fiat, and A. Rosén, *Competitive non-preemptive call control*, in Proceedings of the 5th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'94), Arlington, VA, 1994, pp. 312–320.

[5] B. Awerbuch, R. Gawlick, T. Leighton, and Y. Rabani, *On-line admission control and circuit routing for high performance computing and communication*, in Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS'94), Santa Fe, NM, 1994, pp. 412–423.

[6] C. Berge, *Graphs and Hypergraphs*, 2nd ed., North-Holland, Amsterdam, 1976.

[7] N. K. Cheung, K. Nosu, and G. Winzer, eds., *Dense Wavelength Division Multiplexing Techniques for High Capacity and Multiple Access Communication Systems*, IEEE Journal

on Selected Areas in Communications, vol. 8, no. 6, IEEE Communications Society, New York, 1990, special issue.

[8] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser, *Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms*, Management Sci., 23 (1977), pp. 789–810.

[9] T. Erlebach and K. Jansen, *Conversion of coloring algorithms into maximum weight independent set algorithms*, in ICALP Workshops 2000, Proceedings in Informatics 8, Carleton Scientific, Ontario, Canada, 2000, pp. 135–145.

[10] T. Erlebach and K. Jansen, *Implementation of approximation algorithms for weighted and unweighted edge-disjoint paths in bidirected trees*, in Proceedings of the 4th Workshop on Algorithm Engineering (WAE 2000), Saarbrücken, Germany, 2000.

[11] T. Erlebach and K. Jansen, *The complexity of path coloring and call scheduling*, Theoret. Comput. Sci., 255 (2001), pp. 33–50.

[12] T. Erlebach, K. Jansen, C. Kaklamanis, M. Mihail, and P. Persiano, *Optimal wavelength routing on directed fiber trees*, Theoret. Comput. Sci., 221 (1999), pp. 119–137.

[13] T. Erlebach, K. Jansen, C. Kaklamanis, and P. Persiano, *An optimal greedy algorithm for wavelength allocation in directed tree networks*, in Proceedings of the DIMACS Workshop on Network Design: Connectivity and Facilities Location, DIMACS Ser. Discrete Math. Theoret. Comput. Sci. 40, AMS, Providence, RI, 1998, pp. 117–129.

[14] A. Frank, *Packing paths, circuits, and cuts—a survey*, in Paths, Flows, and VLSI-Layout, B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, eds., Springer-Verlag, Berlin, 1990, pp. 47–100.

[15] N. Garg, V. V. Vazirani, and M. Yannakakis, *Primal-dual approximation algorithms for integral flow and multicut in trees*, Algorithmica, 18 (1997), pp. 3–20.

[16] L. Gargano, P. Hell, and S. Perennes, *Colouring paths in directed symmetric trees with applications to WDM routing*, in Proceedings of the 24th International Colloquium on Automata, Languages and Programming (ICALP'97), Lecture Notes in Comput. Sci. 1256, Springer-Verlag, Berlin, 1997, pp. 505–515.

[17] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin, 1988.

[18] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis, *Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems*, in Proceedings of the 31st Annual ACM Symposium on Theory of Computing (STOC'99), Atlanta, GA, 1999, pp. 19–28.

[19] J. E. Hopcroft and R. M. Karp, *An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs*, SIAM J. Comput., 2 (1973), pp. 225–231.

[20] C. Kaklamanis and P. Persiano, *Efficient wavelength routing on directed fiber trees*, in Proceedings of the 4th Annual European Symposium on Algorithms (ESA'96), Lecture Notes in Comput. Sci. 1136, Springer-Verlag, Berlin, 1996, pp. 460–470.

[21] V. Kann, *Maximum bounded 3-dimensional matching is MAX SNP-complete*, Inform. Process. Lett., 37 (1991), pp. 27–35.

[22] J. Kleinberg, *Approximation Algorithms for Disjoint Paths Problems*, Ph.D. thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, 1996.

[23] J. Kleinberg and É. Tardos, *Disjoint paths in densely embedded graphs*, in Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS'95), Milwaukee, WI, 1995, pp. 52–61.

[24] S. G. Kolliopoulos and C. Stein, *Approximating disjoint-path problems using greedy algorithms and packing integer programs*, in Proceedings of the 6th Integer Programming and Combinatorial Optimization Conference (IPCO VI), Lecture Notes in Comput. Sci. 1412, Springer-Verlag, Berlin, 1998, pp. 153–168.

[25] D. König, *Graphen und Matrizen*, Mat. Fiz. Lapok, 38 (1931), pp. 116–119.

[26] S. R. Kumar, R. Panigrahy, A. Russel, and R. Sundaram, *A note on optical routing on trees*, Inform. Process. Lett., 62 (1997), pp. 295–300.

[27] V. Kumar and E. J. Schwabe, *Improved access to optical bandwidth in trees*, in Proceedings of the 8th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'97), New Orleans, LA, 1997, pp. 437–444.

[28] M. Mihail, C. Kaklamanis, and S. Rao, *Efficient access to optical bandwidth*, in Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS'95), Milwaukee, WI, 1995, pp. 548–557.

[29] B. Schieber and U. Vishkin, *On finding lowest common ancestors: Simplification and parallelization*, SIAM J. Comput., 17 (1988), pp. 1253–1262.

[30] A. Srinivasan, *Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems*, in Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS'97), Miami Beach, FL, 1997, pp. 416–425.

[31] P.-J. Wan and L. Liu, *Maximal throughput in wavelength-routed optical networks*, in Multichannel Optical Networks: Theory and Practice, DIMACS Ser. Discrete Math. Theoret. Comput. Sci. 46, AMS, Providence, RI, 1998, pp. 15–26.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 356–369

PARTITIONS AND (m AND n) SUMS OF PRODUCTS—TWO CELL PARTITION*

GREGORY L. SMITH†

Abstract. We let $SP_m(\langle X_t\rangle_{t=1}^{\infty})$ denote the sums of m increasing products from a sequence $\langle X_t\rangle_{t=1}^{\infty}$. Given $m \neq n$, we construct a two cell partition of $\mathbb{N}$ so that neither cell contains $SP_n\langle X\rangle_{t=1}^{\infty}) \cup SP_m(\langle y\rangle >_{t=1}^{\infty})$ for any sequences $\langle X_t\rangle_{t=1}^{\infty}$ and $\langle y\rangle_{t=1}^{\infty}$.

Key words. partition, sequence, sums of products, finite products

AMS subject classification. 05

PII. S0895480198349014

Introduction. In 1976 Erdos asked whether it is always possible, given a two cell partition of $\mathbb{N}$, to find one cell and an infinite subset all of whose "multilinear expressions... (where each variable occurs only once)" are in that cell. This question was answered in the negative in 1980 [1] when a counterexample was produced for a weaker assertion. However, in 1995 [2] it was shown that Erdos's assertion does hold for a special kind of multilinear expression, namely, sums of a fixed number of products with indices in increasing order. That is, given $n \in \mathbb{N}$ and a sequence $\langle y_t\rangle_{t=1}^{\infty}$ in $\mathbb{N}$, we let $SP_n(\langle y_t\rangle_{t=1}^{\infty}) = \{\sum_{k=1}^{n}\prod_{t\in F_k} y_t : F_1, F_2, \ldots, F_n$ are finite nonempty subsets of $\mathbb{N}$ and for $k \in \{1, 2, \ldots, n-1\}, \max F_k \leq \min F_{k+1}\}$. In addition it was shown that given $m \neq n$ there exists a two cell partition of $\mathbb{N}/S$ such that one cell does not contain $SP_n(\langle Y_t\rangle_{t=1}^{\infty})$ for any sequence $\langle Y_t\rangle_{t=1}^{\infty}$ and the other does not contain $SP_m(\langle X_t\rangle_{t=1}^{\infty})$ for any sequence $\langle X_t\rangle_{t=1}^{\infty}$.

In this paper we provide a construction for such a partition. We make use throughout this work of results obtained in [2].

1. A two cell partition which distinguishes $SP_m(\langle x_t\rangle_{t=1}^{\infty})$ from $SP_n(\langle y_t\rangle_{t=1}^{\infty})$. We will now identify a two cell partition which will distinguish $SP_m(\langle x_t\rangle_{t=1}^{\infty})$ and $SP_n(\langle y_t\rangle_{t=1}^{\infty})$ as guaranteed by [2, Theorem 3.17]. Again as in [2], we fix m and n in $\mathbb{N}$ with $m > n$, and we pick a prime p and an integer k such that $p^k | m, p^k \not| n$ and $p^{k+1} \not| m$.

DEFINITION 1.1.

$$\begin{aligned}
A_0 &= \{p^j : j < \omega\}\\
A_1 &= \{x \in \mathbb{N}\backslash A_0 : p^{2j} < x < p^{2j+1/2} \text{ for some } j \in \omega\},\\
A_2 &= \{x \in \mathbb{N}\backslash A_0 : p^{2j+1/2} < x < p^{2j+1} \text{ for some } j \in \omega\},\\
A_3 &= \{x \in \mathbb{N}\backslash A_0 : p^{2j+1} < x < p^{2j+3/2} \text{ for some } j \in \omega\},\\
A_4 &= \{x \in \mathbb{N}\backslash A_0 : p^{2j+3/2} < x < p^{2j+2} \text{ for some } j \in \omega\}.
\end{aligned}$$

DEFINITION 1.2. *Given $x \in \mathbb{N}$ pick $a(x) \in \mathbb{N}$ with $p^{a(x)} \leq x < p^{a(x)+1}$ and choose $\langle\delta(x,t)\rangle_{t=0}^{a(x)}$ in $\{0, 1, \ldots, p-1\}$ such that $x = \sum_{t=0}^{a(x)}\delta(x,t)p^t$. Let $b(x) = \max\{t < a(x) : \delta(x,t) \neq 0\}$ (where $b(x) = -1$ if $x = \delta(x, a(x))p^{a(x)}$), $c(x) = \max\{t < a(x) : \delta(x,t) \neq p-1\}$ (where $c(x) = -1$ if each $\delta(x,t) = p-1$ for $t < a(x)$), and $d(x) = \min\{t \leq a(x) : \delta(x,t) \neq 0\} = \max\{t : p^t | x\}$. Let $l(x) = \delta(x, a(x))$. Let $\lambda(x) = |\{t : \delta(x,t) \neq 0\}|$.*

*Received by the editors December 7, 1998; accepted for publication (in revised form) March 6, 2001; published electronically August 3, 2001.
http://www.siam.org/journals/sidma/14-3/34901.utml
†Department of Mathematics, Norfolk State University, Norfolk, VA 23504 (glsmith@nsu.edu).

One should note that $a(x), b(x), c(x)$, and $d(x)$ are, respectively, the positions of the leftmost nonzero digit, the next to leftmost nonzero digit, the leftmost digit below $p-1$ to the right of $a(x)$, and the rightmost nonzero digit of x when x is written in base p without leading zeros. $l(x)$ is the leading digit of x in base p, and $\lambda(x)$ is the number of nonzero digits in x. Therefore, if $x = 304130$ and $p = 5$, then $a(x) = 5, b(x) = 3, c(x) = 4, d(x) = 1$, and $l(x) = 3$. The cells which we will be considering are based on $\{A_0, A_1, A_2, A_3, A_4\}$ which is a partition on $\mathbb{N}$. Then A_0 and A_2 are divided into the congruence classes of $\mathbb{Z}p^{k+1}$. A_3 is divided into $2p^{k+1}$ parts: the classes of $\mathbb{Z}p^{k+1}$, which are divided into D and $\mathbb{N}\backslash D$, where $D = \{\sum_{t=1}^{m} z_t$: each $z_t \in A_0$ and $z_1 < z_2 < \cdots < z_m\}$. A_4 is divided into $2p^{k+1}$ parts: the classes of $\mathbb{Z}p^{k+1}$, which are further divided into L_1 and $\mathbb{N}\backslash L_1$, where $L_1 = \{\sum_{t=1}^{m} z_t$: for each $t \in \{1, 2, \ldots, m\}, d(z_t) > a(z_t) - c(z_t)$ and if $t > 1, a(z_t) - c(z_t) > a(z_{t-1})\}$. A_1 is divided into $4p^{k+1}$ parts, namely, the classes of $\mathbb{Z}p^{k+1}$ which are divided by D and $\mathbb{N}\backslash D$, which are further divided by L_2 and $\mathbb{N}\backslash L_2$, where $L_2 = \{\sum_{t=1}^{m} z_t$: for each $t \in \{1, 2, \ldots, m\}, d(z_t) > a(z_t) - b(z_t)$ and if $t > 1, a(z_t) - b(z_t) > a(z_{t-1})\}$.

Given a partition $\mathbb{P}$ we write $x \approx y \pmod{\mathbb{P}}$ to mean that x and y are in the same cell of $\mathbb{P}$. We now define more precisely a partition $\mathbb{P}$ of $\mathbb{N}$.

DEFINITION 1.3. *Define* $\mathbb{P}$ *by agreeing that* $x \approx y \pmod{\mathbb{P}}$ *if and only if*

(1) $x \approx y \pmod{\{A_0, A_1, A_2, A_3, A_4\}}$;
(2) $x \equiv y \bmod p^{k+1}$;
(3) if $x, y \in A_1 \cup A_3$, then $x \approx y \pmod{\{D, \mathbb{N}\backslash D\}}$;
(4) if $x, y \in A_1$, then $x \approx y \pmod{\{L_2, \mathbb{N}\backslash L_2\}}$;
(5) if $x, y \in A_4$, then $x \approx y \pmod{\{L_1, \mathbb{N}\backslash L_1\}}$.

THEOREM 1.4. *There do not exist sequences* $\langle x_t\rangle_{t=1}^{\infty}$ *and* $\langle y\rangle_{t=1}^{\infty}$ *such that* $SP_n(\langle x_{t=1}^{\infty}\rangle) \cup SP_m(\langle y\rangle_{t=1}^{\infty})$ *is monochrome with respect to the partition* $\mathbb{P}$.

Proof. This was proved in [2, Theorem 3.16], except that there the partition $\mathbb{P}$ included an additional division of A_1 requiring that $x \approx y \pmod{\{E, \mathbb{N}\backslash E\}}$, where E was the set of all numbers whose base p expansion has exactly m nonzero digits, and thus $D \subset E$. An examination of the proof shows that the only place the division based on E is used is in the proof of case (4), and that the division based on D works just as well in that case. □

We will first identify cells M' and N', respectively, and we will put the elements of a cell from partition $\mathbb{P}$ into M' if there exists a sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_m(\langle x_t\rangle_{t=1}^{\infty})$ is contained in that cell (similarly for N'). In addition, given a set $S \subset \mathbb{N}$, if we can show that there does not exist an increasing sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset S$, then we will put the elements of S into N' (similarly for M').

DEFINITION 1.5. $N' = (\mathbb{N}\backslash\mathbb{N}p^k) \cup (A_0 \cap \mathbb{N}p^{k+1}) \cup (A_2 \cap \mathbb{N}p^{k+1}) \cup (A_3 \cap \mathbb{N}p^{k+1}) \cup (A_4 \cap \mathbb{N}\backslash L_1 \cap \mathbb{N}p^{k+1}) \cup (A_1 \cap L_2 \cap D \cap \mathbb{N}p^{k+1}) \cup (A_1 \cap \mathbb{N}\backslash L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1})$ *and* $M' = (\mathbb{N}p^k\backslash\mathbb{N}p^{k+1}) \cup (A_4 \cap L_1 \cap \mathbb{N}p^{k+1}) \cup (A_1 \cap L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}) \cup (A_1 \cap \mathbb{N}\backslash L_2 \cap D \cap \mathbb{N}p^{k+1})$.

We now show that $\{N', M'\}$ is a partition of $\mathbb{N}$. We show this by first noting that $\{\mathbb{N}\backslash\mathbb{N}p^k, \mathbb{N}p^k\backslash\mathbb{N}p^{k+1}, \mathbb{N}p^{k+1}\}$ is clearly a partition of $\mathbb{N}$, and $\mathbb{N}\backslash\mathbb{N}p^k \subset N'$ while $\mathbb{N}p^k\backslash\mathbb{N}p^{k+1} \subset M'$. We now need only to account for $\mathbb{N}p^{k+1}$. Recall that $\{A_0, A_1, A_2, A_3, A_4\}$ is a partition of $\mathbb{N}$. Hence $\{A_0 \cap \mathbb{N}p^{k+1}, A_1 \cap \mathbb{N}p^{k+1}, A_2 \cap \mathbb{N}p^{k+1}, A_3 \cap \mathbb{N}p^{k+1}, A_4 \cap \mathbb{N}p^{k+1}\}$ is a partition of $\mathbb{N}p^{k+1}$, and $[(A_0 \cap \mathbb{N}p^{k+1}) \cup (A_2 \cap \mathbb{N}p^{k+1}) \cup (A_3 \cap \mathbb{N}p^{k+1})] \subset N'$. Observe that $\{A_4 \cap \mathbb{N}\backslash L_1 \cap \mathbb{N}p^{k+1}, A_4 \cap L_1 \cap \mathbb{N}p^{k+1}\}$ is a partition of $A_4 \cap \mathbb{N}p^{k+1}$, and $[(A_4 \cap \mathbb{N}\backslash L_1 \cap \mathbb{N}p^{k+1})] \subset N'$ while $(A_4 \cap L_1 \cap \mathbb{N}p^{k+1}) \subset M'$. Note that $\{A_1 \cap L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}, A_1 \cap \mathbb{N}\backslash L_2 \cap D \cap \mathbb{N}p^{k+1}, A_1 \cap \mathbb{N}\backslash L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}\}$ is a partition of $A_1 \cap \mathbb{N}p^{k+1}$, and $[(A_1 \cap \mathbb{N}\backslash L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}) \cup (A_1 \cap L_2 \cap D \cap \mathbb{N}p^{k+1})] \subset N'$, while $[(A_1 \cap L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}) \cup (A_1 \cap \mathbb{N}\backslash L_2 \cap D \cap \mathbb{N}p^{k+1})] \subset M'$.

The following is easily verified.

LEMMA 1.6. *There does not exist an increasing sequence* $\langle x_t\rangle_{t=1}^{\infty}$ *such that* $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_0$.

DEFINITION 1.7. $A_5 = \{x\epsilon\mathbb{N}\backslash A_0 : p^{2j+1/4} < x < p^{2j+1/2}$ *for some* $j\epsilon\,\omega\}$.

We state the following without proof.

LEMMA 1.8.

(a) *If* $x, y\epsilon A_5$ *then* $xy\epsilon A_2$.

(b) *If* $x\epsilon A_2$ *and* $y\epsilon A_3$, *then* $xy\epsilon A_1 \cup A_4$.

LEMMA 1.9. *There does not exist an increasing sequence* $\langle x_t\rangle_{t=1}^{\infty}$ *such that* $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subseteq A_2$.

Proof. Suppose we have an increasing sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_m(\langle x_t\rangle)_{t=1}^{\infty} \subseteq A_2$. Pick $\ell\epsilon\mathbb{N}$ such that $1+p^{-} < p^{1/2}$. (If $p > 2, k = 1$ will suffice. To see this consider that in base $p, 121 < 1000$. That is, $p^2 + 2p + 1 < p^3$ so that $(1 + \frac{1}{p})^2 < p$.) Let $y = \sum_{t=1}^{m-1} x_t$ and pick v such that $x_v > p^{a(y)+\ell}$ (Definition 1.2).

We first show that

$$(*) \text{ if } F \text{ is a finite nonempty subset of } \mathbb{N} \text{ and } \min F \geq v, \text{then} \prod_{t\epsilon F} x_t \notin A_3 \cup A_4.$$

Indeed, let such F be given and let $z = \prod_{t\epsilon F} x_t$. Observe that $a(z) + 1 - \ell \geq a(y)+\ell+1-\ell = a(y)+1$. Thus $p^{a(z)} < z+y < p^{a(z)+1}+p^{a(y)+1} \leq p^{a(z)+1}+p^{a(z)+1-\ell} = p^{a(z)+1}(1+p^{-\ell}) < p^{a(z)+3/2}$. Since $z+y\epsilon A_2$ we have $a(z)$ is not odd; that is, $z \notin A_3\cup A_4$ as claimed.

Next observe that for at most one $t \geq v$ we can have $x_t\epsilon A_2$. (For if $s > t \geq v$ and x_s and x_t are in A_2, then by [2, Lemma 3.12(a)], $x_s{\cdot}x_t\epsilon A_3\cup A_4$, contradicting (*).) Also observe that for all $t > v, x_t \notin A_0$. Indeed, if we had such $x_t = p^{a(x_t)}$ we would have $p^{a(x_t)} < x_t+y < p^{a(x_t)}+p^{a(y)+1} \leq p^{a(x_t)}+p^{a(x_t)-\ell} = p^{a(x_t)}(1+p^{-\ell}) < p^{a(x_t)+1/2}$, while $x_t + y\epsilon A_2$, a contradiction. (We have $p^{a(x_t)-\ell} \geq p^{a(y)+1}$ because $a(x_t) \geq a(y)+\ell+1$.)

Now pick $w \geq v$ such that $p^{a(x_w)} > y/(p^{1/4} - 1)$ and for all $t \geq w, x_t\epsilon A_2$. Then for all $t \geq w, p^{a(x_t)+1/2} - p^{a(x_t)+1/4} > y$. Now we claim that for all $t \geq w, x_t\epsilon A_5$. Therefore let $t \geq w$ be given. Since $x_t \notin A_0 \cup A_2$ and by (*) $x_t \notin A_3 \cup A_4$ we have $x_t\epsilon A_1$. Then $a(x_t)$ is even and $p^{a(x_t)} < x_t < p^{a(x_t)+1/2}$ and $x_t + y > p^{a(x_t)+1/2}$ since $x_t + y\epsilon A_2$. Thus $x_t > p^{a(x_t)+1/2} - y > p^{a(x_t)+1/4}$, so $x_t\epsilon A_5$ as claimed.

Now pick i, j, s, t, with $i > j > s > t \geq w$. Then $x_i \cdot x_j\epsilon A_2$ and $x_s \cdot x_t\epsilon A_2$ by Lemma 1.8(a) so $x_i \cdot x_j \cdot x_s \cdot x_t\epsilon A_3 \cup A_4$ by [2, Lemma 3.12(a)]. This contradicts (*) and completes the proof. □

LEMMA 1.10. *There does not exist an increasing sequence* $\langle x_t\rangle_{t=1}^{\infty}$ *such that* $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_3$.

Proof. Suppose such a sequence $\langle x_t\rangle_{t=1}^{\infty}$ exists. By [2, Theorem 2.4 and Lemmas 2.2(b), 3.12(a) and 3.12(b)] we can assume that $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_j$ for $j = 0, 1,$ or 4. First assume that $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_0$, in which case each $x_t\epsilon A_0$. Now consider $y = x_{m-1} + \cdots + x_2 + x_1$. Note for infinitely many i's we have $x_i > y$ and $a(x_i)$ is even or odd. If infinitely many of the $a(x_i)$'s are even, then pick one such i. Then $a(x_i + y) = a(x_i)$ so $x_i + y\epsilon A_1 \cup A_2$, a contradiction. If infinitely many of the $a(x_i)$'s are odd, then pick $v > w$ such that $a(x_v)$ and $a(x_w)$ are odd and x_v and x_w are larger than y. Hence $x_v x_w + y\epsilon A_1 \cup A_2$, a contradiction.

Now assume that $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4$. Again, let $y = x_{m-1} + \cdots + x_2 + x_1$. Pick ℓ such that $a(x_\ell) > a(y) + 2$. Now $x_\ell + y\epsilon SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subseteq A_3$ and $p^{a(x_\ell)+1} < x_\ell + y < p^{a(x_\ell)+1} + p^{a(y)+1} < p^{a(x_\ell)+2}$ and $a(x_\ell)$ is odd, so in fact $p^{a(x_\ell)} < x_\ell + y < p^{a(x_\ell)+1/2}$. However, then $p^{a(x_\ell)} < x_\ell < p^{a(x_\ell)+1/2}$, so $x_\ell\epsilon A_3$, a contradiction.

Finally assume that $FP(\langle x_t\rangle_{t=1}^{\infty}) \subseteq A_1$. Pick ℓ such that $p^{a(x_\ell)+1/2} > y/(p^{1/2}-1)$, so $y < p^{a(x_\ell)+1} - p^{a(x_\ell)+1/2}$. Then $p^{a(x_\ell)} < x_\ell < p^{a(x_\ell)+1/2}$ so $p^{a(x_\ell)} < x_\ell + y < p^{a(x\ell)+1}$ and $a(x_\ell)$ is even, while $x_\ell + y \epsilon A_3$, a contradiction. □

LEMMA 1.11. *There does not exist an increasing sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash\mathbb{N}p^k$.*

Proof. Given a sequence $\langle x_t\rangle_{t=1}^{\infty}$ we can assume that the elements of $FP(\langle x_t\rangle_{t=1}^{\infty})$ are congruent mod p^{k+1} by [2, Theorem 2.4 and Lemma 2.2(b)]. Pick i such that $x_t \equiv i \mod p^k$ for each t. Therefore, $x_1 + x_2 + \cdots + x_m \equiv m \cdot i \mod p^k$. However, $p^k | m \cdot i$. Therefore, $SP_m(\langle x_t\rangle_{t=1}^{\infty})$ can be contained only in the class 0 in $\mathbb{Z}p^k$. □

We state the following without proof.

LEMMA 1.12. *Given a prime p and $a, z \epsilon \mathbb{N}$. If $a \equiv a^2 \mod p^z$, then $a \equiv 0$ or $a \equiv 1 \mod p^z$.*

LEMMA 1.13. *There does not exist an increasing sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}p^k\backslash\mathbb{N}p^{k+1}$.*

Proof. Suppose such a sequence $\langle x_t\rangle_{t=1}^{\infty}$ and ℓ exist. By [2, Theorem 2.4 and Lemma 2.2(b)], we can assume that the elements of $FP(\langle x_t\rangle_{t=1}^{\infty})$ are congruent mod p^{k+1}. Hence for $x \epsilon FP(\langle x_t\rangle_{t=1}^{\infty})$ we have that $x \equiv 0 \mod p^{k+1}$ or $x \equiv 1 \mod p^{k+1}$ since $\overline{0}$ and $\overline{1}$ are the only idempotents in $\mathbb{Z}p^{k+1}$ by Lemma 1.12. If $x \equiv 0 \mod p^{k+1}$ for $x \epsilon FP(\langle x_t\rangle_{t=1}^{\infty})$ we have $SP_n\langle x_t\rangle_{t=1}^{\infty} \subseteq \mathbb{N}p^{k+1}$. Therefore $x \equiv 1 \mod p^{k+1}$ for $\epsilon FP(\langle x_t\rangle_{t=1}^{\infty})$. Hence $n \cdot 1 \equiv \ell p^k \mod p^{k+1}$ for some $\ell \in \{1, 2, \ldots, p-1\}$. Therefore for some c, $cp^{k+1} = n - \ell p^k$. Therefore, $n = cp^{k+1} - \ell p^k$ which means that $p^k | n$, a contradiction. □

LEMMA 1.14. *Given the sequence where $x_t = p^{3\cdot 2^{2t-1}\cdot 4k} - p^{2\cdot 2^{2t-1}\cdot 4k}$ for $t \epsilon \mathbb{N}$, let $F \epsilon [\mathbb{N}]_{>0}^{<\omega}$ with $r = \min F$ and let $\ell(F) = \sum_{t\epsilon F} 3 \cdot 2^{2t-1} \cdot 4k$.*

(a) *Then $p^{\ell(F)} > \prod_{t\epsilon F} x \geq p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$.*

(b) *Then $a(\prod_{t\epsilon F} x_t) - c(\prod_{t\epsilon F} x_t) \geq a(x_r) - c(x_r) - 1$.*

Proof. (a) Given $F \epsilon [\mathbb{N}]_{>0}^{<\omega}$, let $\ell(F) = \sum_{t\epsilon F} 3 \cdot 2^{2t-1} \cdot 4k$ and let $r = \min F$. We claim that $p^{\ell(F)} > \prod_{t\epsilon F} x_t \geq p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$ which we will show by induction of $|F|$. Suppose $|F| = 1$. Then $F = \{r\}$ and $p^{\ell(F)} = p^{3\cdot 2^{2r-1}\cdot 4k} > x_r = p^{3\cdot 2^{2r-1}\cdot 4k} - p^{2\cdot 2^{2r-1}\cdot 4k} = p^{\ell(F)} - p^{\ell(F)-2^{2r-1}\cdot 4k}$. Now assume $|F| > 1$. Let $G = F\backslash\{r\}$. We have $p^{\ell(F)-3\cdot 2^{2r-1}\cdot 4k} = p^{\ell(G)} > \prod_{t\epsilon G} x_t \geq p^{\ell(G)} - \sum_{t\epsilon G} p^{\ell(G)-2^{2t-1}\cdot 4k} = p^{\ell(F)-3\cdot 2^{2r-1}\cdot 4k} - \sum_{t\epsilon G} p^{\ell(F)-3\cdot 2^{2r-1}\cdot 4k-2^{2r-1}\cdot 4k}$. Also $p^{3\cdot 2^{2r-1}\cdot 4k} > x_r = p^{3\cdot 2^{2r-1}\cdot 4k} - p^{2\cdot 2^{2r-1}\cdot 4k}$. Hence $p^{\ell(F)} > \prod_{t\epsilon G} x_t \cdot x_r \geq p^{\ell(F)} - \sum_{t\epsilon G} p^{\ell(F)-2^{2t-1}\cdot 4k} - p^{\ell(F)-3\cdot 2^{2r-1}\cdot 4k+2\cdot 2^{2r-1}\cdot 4k} + \sum_{t\epsilon G} p^{\ell(F)-3\cdot 2^{2r-1}\cdot 4k-2^{2t-1}\cdot 4k+2\cdot 2^{2r-1}\cdot 4k} > p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$. Therefore $p^{\ell(F)} > \prod_{t\epsilon F} x_t \geq p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$.

(b) Suppose $|F| = 1$; then $F = \{r\}$ and we have that $x_r = p^{\ell(F)} - p^{\ell(F)-2^{2r-1}\cdot 4k}$. Hence the digits are $p-1$ from place position $\ell(F)-1$ to $\ell(F) - 2^{2r-1} \cdot 4k$ in base p. Therefore,

$$\begin{aligned} a\left(\prod_{t\epsilon F} x_t\right) - c\left(\prod_{t\epsilon F} x_t\right) &= \ell(F) - 1 - [\ell(F) - 2^{2r-1} \cdot 4k - 1] \\ &= 2^{2r-1} \cdot 4k. \end{aligned}$$

Now assume $|F| > 1$. Then we have that $p^{\ell(F)} > \prod_{t\epsilon F} x_t \geq p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$ by (a). If $\prod_{t\epsilon F} x_t = p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$, then the digits are $p-1$ from place

position $\ell(F)-1$ to $\ell(F)-2^{2r-1}\cdot 4k+1$. Therefore,

$$\begin{aligned} a\left(\prod_{t\epsilon F} x_t\right) - c\left(\prod_{t\epsilon F} x_t\right) &= \ell(F)-1-[\ell(F)-2^{2r-1}\cdot 4k] \\ &= 2^{2r-1}\cdot 4k-1. \end{aligned}$$

And if $\prod_{t\epsilon F} x_t > p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$, then the number of $p-1$'s starting from place position $\ell(F)-1$ may increase. Hence $c(\prod_{t\epsilon F} x_t) \le \ell(F)-2^{2r-1}\cdot 4k$. Therefore, in this case we have that

$$\begin{aligned} a\left(\prod_{t\epsilon F} x_t\right) - c\left(\prod_{t\epsilon F} x_t\right) &\ge \ell(F)-1-[\ell(F)-2^{2r-1}\cdot 4k] \\ &= 2^{2r-1}\cdot 4k-1. \end{aligned}$$

Therefore, in either case,

$$a\left(\prod_{t\epsilon F} x_t\right) - c\left(\prod_{t\epsilon F} x_t\right) \ge 2^{2r-1}\cdot 4k-1 = a(x_r)-c(x_r)-1. \qquad \square$$

LEMMA 1.15. *The sequence where* $x_t = p^{3\cdot 2^{2t-1}\cdot 4k} - p^{2\cdot 2^{2t-1}\cdot 4k}$ *for* $t\in\mathbb{N}$ *has the following properties:*

(a) $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap \mathbb{N}p^{k+1}$,
(b) $a(x_t)-c(x_t) < d(x_t)$ *for* $t\epsilon\mathbb{N}$,
(c) $a(x_{t+1})-c(x_{t+1}) > a(\prod_{\ell=1}^{t} x_\ell)+1$ *for* $t\epsilon\mathbb{N}$,
(d) $a(\prod_{t\epsilon F_i} x_t) - c(\prod_{t\epsilon F_i} x_t) > a(\prod_{t\epsilon F_j} x_t)$ *for* $F_i, F_j \epsilon [\mathbb{N}]_{>0}^{<\omega}$ *and* $\max F_j < \min F_i$,
(e) $a(\prod_{t\epsilon F} x_t) - c(\prod_{t\epsilon F} x_t) < d(\prod_{t\epsilon F} x_t)$ *for* $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$,
(f) $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4$ *for* $u\epsilon\mathbb{N}\backslash\{1\}$,
(g) $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash L_1$.

Note, in the case $n=1$, *that we are considering* $FP(\langle x_t\rangle_{t=1}^{\infty})$ *and stated results hold.*

Proof. (a) Given $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$, let $\ell(F) = \sum_{t\epsilon F} 3\cdot 2^{2t-1}\cdot 4k$ and let $r=\min F$. By Lemma 1.14(a) we have that

$$p^{\ell(F)} > \prod_{t\epsilon F} x_t \ge p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}.$$

Now note that $2p^{\ell(F)-2^{2r-1}\cdot 4k} > \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$. Let $x,y\epsilon F$ and suppose $x<y$. Then $p^{\ell(F)-2^{2x-1}\cdot 4k} > p^{\ell(F)-2^{2y-1}\cdot 4k}$. And since the elements of F are all distinct we have that $2p^{\ell(F)-2^{2r-1}\cdot 4k} > \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k}$ since $r=\min F$.

Hence

$$\begin{aligned} \prod_{t\epsilon F} x_t &\ge p^{\ell(F)} - \sum_{t\epsilon F} p^{\ell(F)-2^{2t-1}\cdot 4k} \\ &> p^{\ell(F)} - 2p^{\ell(F)-2^{2r-}\cdot 4k} \\ &\ge p^{\ell(F)} - p^{\ell(F)-7} \text{ since } p\ge 2 \text{ and } r,k\ge 1 \\ &= p^{\ell(F)}(1-p^{-7}) \\ &> p^{\ell(F)-1/2}. \end{aligned}$$

Therefore $\prod_{t\epsilon F} x_t \epsilon A_4 \cap \mathbb{N}p^{k+1}$.

(b) Consider $x_t = p^{3\cdot 2^{2t-1}\cdot 4k} - p^{2\cdot 2^{2t-1}\cdot 4k}$. Therefore

$$\begin{aligned} a(x_t) - c(x_t) &= 3\cdot 2^{2t-1}\cdot 4k - 1\left[2\cdot 2^{2t-1}\cdot 4k - 1\right] \\ &= 2^{2t-1}\cdot 4k \\ &< 2\cdot 2^{2t-1}\cdot 4k \\ &= d(x_t). \end{aligned}$$

(c)

$$\begin{aligned} a\left(\prod_{t=1}^{\ell} x_t\right) &= \ell - 1 - \sum_{t=1}^{\ell} a(x_t) \text{by (a) and [2, Lemma 3.8(a)]} \\ &= \ell - 1 + \sum_{t=1}^{\ell}(3\cdot 2^{2t-1}\cdot 4k - 1) \\ &= \sum_{t=1}^{\ell} 3\cdot 2^{2t-1}\cdot 4k - 1 \\ &< \frac{3}{2}\cdot 4k\cdot \sum\nolimits_{t=1}^{\ell} 4^t \\ &= \frac{3}{2}\cdot 4k\cdot \frac{4^{\ell+1}-4}{3} \\ &= 4k\cdot \frac{4^{\ell+1}-4}{2} \\ &= 4k(2\cdot 4^{\ell} - 2) \\ &= 4k(2^{2\ell+1} - 2) \\ &< 2^{2\ell+1}\cdot 4k - 1 \\ &= a(x_{\ell+1}) - c(x_{\ell+1}) - 1. \end{aligned}$$

Hence $a(x_{\ell+1}) - c(x_{\ell+1}) > a(\prod_{t=1}^{\ell} x_t) + 1$.

(d) Pick $F_i, F_j \epsilon [\mathbb{N}]^{<\omega}_{>0}$, where $\max F_j < \min F_i$. Let $r = \min F_i$.

Therefore

$$\begin{aligned} a\left(\prod_{t\epsilon F_i} x_t\right) - c\left(\prod_{t\epsilon F_i} x_t\right) &\geq a(x_r) - c(x_r) - 1 \text{ by Lemma 1.14(b)} \\ &> a\left(\prod_{t=1}^{r-1} x_t\right) \text{ by (c)} \\ &\geq a\left(\prod_{t\epsilon F_j} x_t\right). \end{aligned}$$

Hence $a(\prod_{t\epsilon F_i} x_t) - c(\prod_{t\epsilon F_i} x_t) > a(\prod_{t\epsilon F_j} x_t)$.

(e) Given $F\epsilon[\mathbb{N}]^{<\omega}_{>0}$, let $r = \min F$. Note

$$\begin{aligned} a\left(\prod_{t\epsilon F} x_t\right) - c\left(\prod_{t\epsilon F} x_t\right) &\leq a(x_r) - c(x_r) \text{ by (a) and [2, Lemma 3.10(a)]} \\ &< d(x_r) \text{ by (b)} \\ &\leq d\left(\prod_{t\epsilon F} x_t\right). \end{aligned}$$

(f) For $u\epsilon\mathbb{N}$ pick $x\epsilon SP_u(\langle x_t\rangle_{t=1}^{\infty})$. Hence $x = \prod_{t\epsilon F_1} x_t + \prod_{t\epsilon F_2} x_t + \cdots + \prod_{t\epsilon F_u} x_t$, where $\max F_i < \min F_{i+1}$ for $i\epsilon\{1,\ldots,u-1\}$. Note $\prod_{t\epsilon F_u} x_t\epsilon A_4$. And since we have no carrying when adding by (d) and (e), we have that $a(x) = a(\prod_{t\epsilon F_u} x_t)$. Hence since $\prod_{t\epsilon F} x_t > p^{a(x)+1/2}$ we have $x > p^{a(x)+1/2}$. Therefore $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4$.

(g) Pick $x\epsilon SP_n(\langle x_t\rangle_{t=1}^{\infty})$. Hence $x = \prod_{t\epsilon F_1} x_t + \prod_{t\epsilon F_2} x_t + \cdots + \prod_{t\epsilon F_n} x_t$, where $\max F_i < \min F_{i+1}$ for $i\epsilon\{1,\ldots,n-1\}$. Suppose $x\epsilon L_1$. Then there exist $z_1, z_2, \ldots, z_m$ such that $\prod_{t\epsilon F_n} x_t + \cdots + \prod_{t\epsilon F_1} x_t = z_m + \cdots + z_1$, where $d(z_m) > a(z_m) - c(z_m) > a(z_{m-1}) > d(z_{m-1}) > a(z_{m-1}) - c(z_{m-1}) > \cdots > d(z_1) > a(z_1) - c(z_1)$.

Now pick the first α and β starting from the left, where $a(\prod_{t\epsilon F_\alpha} x_t) = a(z_\beta)$ and $a(\prod_{t\epsilon F_\alpha} x_t) \neq d(z_\beta)$. This must occur since $m > n$. Hence we have case (A), where $d(\prod_{t\epsilon F_\alpha} x_t) < d(z_\beta)$, and case (B), where $d(\prod_{t\epsilon F_\alpha} x_t) > d(z_\beta)$. The following case illustration will be helpful; * means a nonzero digit in base p.

$$
\begin{array}{lccccc}
 & a(z_\beta) & & d(z_\beta) & a(z_{\beta-1}) & d(z_{\beta-1}) \\
 & * & & * & * & * \\
\text{(A)} & * & & & *\rightarrow & \\
 & a(\prod_{t\epsilon F_\alpha} x_t) & & & d(\prod_{t\epsilon F_\alpha} x_t) & \\
 & * & * & \leftarrow * & & \\
\text{(B)} & a(\prod_{t\epsilon F_\alpha} x_t) & d(\prod_{t\epsilon F_\alpha} x_t) & a(\prod_{t\epsilon F_\alpha - 1} x_t) & &
\end{array}
$$

Now consider case (A). If $c(z_\beta)$ lies between $a(z_\beta)$ and $d(z_\beta)$, then clearly $a(z_\beta) - c(z_\beta) = a(\prod_{t\epsilon F_\alpha} x_t) - c(\prod_{t\epsilon F_\alpha} x_t)$. However, if $c(z_\beta)$ lies just to the right of $d(z_\beta)$, which will occur if the digits are $p-1$ from $a(z_\beta)$ to $d(z_\beta)$, and $a(z_{\beta-1})$ is also to the immediate right of $d(z_\beta)$, then $c(\prod_{t\epsilon F_\alpha} x_t)$ can lie to the right of $a(z_{\beta-1})$ (since $a(z_{\beta-1})$ might be the position for $p-1$) in which case $a(z_\beta) - c(z_\beta) < a(\prod_{t\epsilon F_\alpha} x_t) - c(\prod_{t\epsilon F_\alpha} x_t)$. Therefore, we have $d(\prod_{t\epsilon F_\alpha} x_t) \leq a(z_{\beta-1}) < a(z_\beta) - c(z_\beta) \leq a(\prod_{t\epsilon F_\alpha} x_t) - c(\prod_{t\epsilon F_\alpha} x_t) < d(\prod_{t\epsilon F_\alpha} x_t)$, a contradiction. As for case (B), if $c(\prod_{t\epsilon F_\alpha} x_t)$ lies between $a(\prod_{t\epsilon F_\alpha} x_t)$ and $d(\prod_{t\epsilon F_\alpha} x_t)$, then $a(\prod_{t\epsilon F_\alpha} x_t) - c(\prod_{t\epsilon F_\alpha} x_t) = a(z_\beta) - c(z_\beta)$. And if $c(\prod_{t\epsilon F_\alpha} x_t)$ is to the immediate right of $d(\prod_{t\epsilon F_\alpha} x_t)$, which will occur if the digits are $p-1$ from $a(\prod_{t\epsilon F_\alpha} x_t)$ to $d(\prod_{t\epsilon F_\alpha} x_t)$, and $a(\prod_{t\epsilon F_{\alpha-1}} x_t)$ is just to the right of $d(\prod_{t\epsilon F_\alpha} x_t)$, then $c(z_\beta)$ can lie to the right of $a(\prod_{t\epsilon F_{\alpha-1}} x_t)$ (since $p-1$ may be in the $a(\prod_{t\epsilon F_{\alpha-1}} x_t)$ position) in which case $a(\prod_{t\epsilon F_\alpha} x_t) - c(\prod_{t\epsilon F_\alpha} x_t) < a(z_\beta) - c(z_\beta)$. Therefore, we have $d(z_\beta) \leq a(\prod_{t\epsilon F_{\alpha-1}} x_t) < a(\prod_{t\epsilon F_\alpha} x_t) - c(\prod_{t\epsilon F_\alpha} x_t) \leq a(z_\beta) - c(z_\beta) < d(z_\beta)$, a contradiction. Hence $x\epsilon\mathbb{N}\backslash L_1$. Therefore $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash L_1$. □

LEMMA 1.16. *There is a sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap L_1 \cap \mathbb{N}p^{k+1}$.*

Proof. Pick the sequence $\langle x_t\rangle_{t=1}^{\infty}$ of Lemma 1.15. Then by Lemma 1.15(a) we have that $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap \mathbb{N}p^{k+1}$. And by Lemma 1.15(f) we have $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4$. By Lemmas 1.15(d) and (e) we have for $x\epsilon SP_m(\langle x_t\rangle_{t=1}^{\infty})$ that $x\epsilon L_1$. Hence $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset L_1$. And since $\overline{0}\cdot m = \overline{0}$ in $\mathbb{Z}p^{k+1}$ we have that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap L_1 \cap \mathbb{N}p^{k+1}$. □

LEMMA 1.17. *There is a sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap \mathbb{N}\backslash L_1 \cap \mathbb{N}p^{k+1}$.*

Proof. Again pick the sequence $\langle x_t\rangle_{t=1}^{\infty}$ of Lemma 1.15. Then $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap \mathbb{N}p^{k+1}$ by Lemma 1.15(a). And by Lemma 1.15(f) we have $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4$. By Lemma 1.15(g) we have that $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash L_1$. And since $\overline{0}\cdot m = \overline{0}$ in $\mathbb{Z}p^{k+1}$ we have that $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset A_4 \cap \mathbb{N}\backslash L_1 \cap \mathbb{N}p^{k+1}$. □

LEMMA 1.18. *Given the sequence where $x_t = \sum_{i=1}^{m+1} p^{(1+i)(m+1)^{2t-1}\cdot 6k}$ for $t\epsilon\mathbb{N}$, let $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$ with $r = \min F$ and let $\ell(F) = \sum_{t\epsilon F}(m+2)(m+1)^{2t-1}\cdot 6k$.*

(a) *Then*

$$p^{\ell(F)} < \prod_{t\epsilon F} x_t < p^{\ell(F)} + 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}.$$

(b) *Then*

$$a\left(\prod_{t\epsilon F} x_t\right) - b\left(\prod_{t\epsilon F} x_t\right) \geq a(x_r) - b(x_r) - 2.$$

Proof. (a) Given $F\epsilon[\mathbb{N}]^{\leq\omega}_{>0}$, let $\ell(F) = \sum_{t\epsilon F}(m+2)(m+1)^{2t-1}\cdot 6k$ and let $r = \min F$. We claim that $p^{\ell(F)} < \prod_{t\epsilon F} x_t < p^{\ell(F)} + 2\sum_{t\epsilon F} p^{\ell(F)-(m-1)^{2t-1}\cdot 6k+1}$ which we will show by induction on $|F|$. Suppose $|F| = 1$. Then $F = \{r\}$ and $p^{\ell(F)} = p^{(m+2)(m+1)^{2r-1}\cdot 6k} < x_r = \sum_{i=1}^{m+1} p^{(1+i)(m+1)^{2r-1}\cdot 6k} < p^{(m+2)(m+1)^{2r-1}\cdot 6k} + p^{(m+1)(m+1)^{2r-1}\cdot 6k+1} < p^{(m+2)(m+1)^{2r-1}\cdot 6k} + 2p^{(m+1)(m+1)^{2r-1}\cdot 6k+1} = p^{\ell(F)} + 2p^{\ell(F)-(m+2)(m+1)^{2t-1}\cdot 6k+1}$. Now assume $|F| > 1$. Let $G = F/\{r\}$. We have $p^{\ell(F)-(m+2)(m+1)^{2r-1}\cdot 6k} = p^{\ell(G)} < \prod_{t\epsilon G} x_t < p^{\ell(G)} + 2\sum_{t\epsilon G} p^{\ell(G)-(m+1)^{2t-1}\cdot 6k+1} = p^{\ell(F)-(m+2)(m+1)^{2r-1}\cdot 6k} + 2\sum_{t\epsilon G} p^{\ell(F)-(m+2)(m+1)^{2r-1}\cdot 6k-(m+1)^{2t-1}\cdot 6k+1}$. Hence since $x_r < p^{(m+2)(m+1)^{2r-1}\cdot 6k} + p^{(m+1)(m+1)^{2r-1}\cdot 6k+1}$, we have

$$\begin{aligned}
p^{\ell(F)} &< \prod_{t\epsilon G} x_t \cdot x_r < p^{\ell(F)} + 2\sum_{t\epsilon G} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1} + p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} \\
&\quad + 2\sum_{t\epsilon G} p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1-(m+1)^{2t-1}\cdot 6k+1} \\
&< p^{\ell(F)} + 2\sum_{t\epsilon G} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1} + p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} + p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} \\
&= p^{\ell(F)} + 2\sum_{t\epsilon G} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1} + 2p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} \\
&= p^{\ell(F)} + 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}.
\end{aligned}$$

(b) By (a) we have that $p^{\ell(F)} < \prod_{t\epsilon F} x_t < p^{\ell(F)} + 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}$. Observe that $p^{\ell(F)} + 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}$ has its leftmost nonzero digit in place position $\ell(F)$ in base p, and its next to leftmost nonzero digit comes from $2p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1}$ and so occurs at or to the right of position $\ell(F) - (m+1)^{2r-1}\cdot 6k+1$. Hence $b(\prod_{t\epsilon F} x_t \leq \ell(F) - (m+1)^{2r-1}\cdot 6k+2$, and $a(\prod_{t\epsilon F} x_t) = \ell(F)$.

Therefore

$$\begin{aligned}
a\left(\prod_{t\epsilon F} x_t\right) - b\left(\prod_{t\epsilon F} x_t\right) &\leq \ell(F) - [\ell(F) - (m+1)^{2r-1}\cdot 6k+2] \\
&= (m+1)^{2r-1}\cdot 6k+1 \\
&= a(x_r) - b(x_r) - 2. \qquad \square
\end{aligned}$$

LEMMA 1.19. *The sequence where* $x_t = \sum_{i=1}^{m+1} p^{(1+i)(m+1)^{2t-1}\cdot 6k}$ *for* $t\epsilon\mathbb{N}$ *has the following properties:*

(a) $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1 \cap \mathbb{N}p^{k+1}$,
(b) $a(x_t) - b(x_t) < d(x_t)$ *for* $t\epsilon\mathbb{N}$,
(c) $a(x_{t+1}) - b(x_{t+1}) > a(\prod_{\ell=1}^{t} x_\ell) + 2$ *for* $t\epsilon\mathbb{N}$,
(d) $a(\prod_{t\epsilon F_i} x_t) - b(\prod_{t\epsilon F_i} x_t) > a(\prod_{t\epsilon F_j} x_t)$ *for* $F_i, F_j\epsilon[\mathbb{N}]_{>0}^{<\omega}$ *and* $\max F_j < \min F_i$,
(e) $a(\prod_{t\epsilon F} x_t) - b(\prod_{t\epsilon F} x_t) < d(\prod_{t\epsilon F} x_t)$ *for* $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$,
(f) $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1$ *for* $u\epsilon[\mathbb{N}]\backslash\{1\}$,
(g) $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash D$ *for* $u\epsilon\mathbb{N}\backslash\{1\}$,
(h) $SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash L_2$.

Again, in the case $n = 1$, *we are considering* $FP(\langle x_t\rangle_{t=1}^{\infty})$ *and stated results hold.*

Proof. (a) Given $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$, let $\ell(F) = \sum_{t\epsilon F}(m+2)(m+1)^{2t-1}\cdot 6k$ and let $r = \min F$. By Lemma 1.18(a) we have that

$$p^{\ell(F)} < \prod_{t\epsilon F} x_t < p^{\ell(F)} + 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}.$$

Now note that

$$4p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} > 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}.$$

Let $x, y\epsilon F$ and suppose $x < y$. Then $p^{\ell(F)-(m+1)^{2x-1}\cdot 6k+1} > p^{\ell(F)-(m+1)^{2y-1}\cdot 6k+1}$. And since the elements of F are distinct we have that $a(p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1}) = a(\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1})$. Therefore $4p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} > 2\sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1}$ since $r = \min F$. Hence

$$\begin{aligned}
\prod_{t\epsilon F} x_t &< p^{\ell(F)} + \sum_{t\epsilon F} p^{\ell(F)-(m+1)^{2t-1}\cdot 6k+1} \\
&< p^{\ell(F)} + 4p^{\ell(F)-(m+1)^{2r-1}\cdot 6k+1} \\
&\le p^{\ell(F)} + p^{\ell(F)-16} \text{ since } m \ge 2, r, k \ge 1 \text{ and } p^2 \ge 4 \\
&= p^{\ell(F)}(1 + p^{-6}) \\
&< p^{\ell(F)+1/2}.
\end{aligned}$$

Therefore $\prod_{t\epsilon F} x_t\epsilon A_1 \cap \mathbb{N}p^{k+1}$.

(b) Consider $x_t = \sum_{i=1}^{m+1} p^{(1+i)(m+1)^{2t-1}\cdot 6k}$. Therefore

$$\begin{aligned}
a(x_t) - b(x_t) &= (m+2)(m+1)^{2t-1}\cdot 6k - (m+1)(m+1)^{2t-1}\cdot 6k \\
&= (m+1)^{2t-1}\cdot 6k \\
&< 2(m+1)^{2t-1}\cdot 6k \\
&= d(x_t).
\end{aligned}$$

(c)

$$\begin{aligned}
a\left(\prod_{t=1}^{\ell} x_t\right) &= \sum_{t=1}^{\ell} a(x_t) \text{ by (a) and [2, Lemma 3.8(b)]} \\
&= \sum_{t=1}^{\ell} (m+2)(m+1)^{2t-1} \cdot 6k \\
&= \frac{m+2}{m+1} \cdot 6k \cdot \sum_{t=1}^{\ell} (m+1)^{2t} \\
&= \frac{m+2}{m+1} \cdot 6k \cdot \frac{(m+1)^{2\ell+2} - (m+1)^2}{(m+1)^2 - 1} \\
&= (m+2)6k \cdot \frac{(m+1)^{2\ell+1} - (m+1)}{m^2 + 2m} \\
&= 6k \cdot \frac{(m+1)^{2\ell+1} - (m+1)}{m} \\
&< (m+1)^{2\ell+1} \cdot 6k - 2 \\
&= a(x_{\ell+1}) - b(x_{\ell+1}) - 2.
\end{aligned}$$

(d) Pick $F_i, F_j \epsilon [\mathbb{N}]^{<\omega}_{>0}$, where $\max F_j < \max F_i$. Let $r = \min F_i$. Therefore

$$\begin{aligned}
a\left(\prod_{t\epsilon F_i} x_t\right) - b\left(\prod_{t\epsilon Fi} x_t\right) &\geq a(x_r) - b(x_r) - 2 \text{ by Lemma 1.18(b)} \\
&\geq a\left(\prod_{t=1}^{r-1} x_t\right) \text{ by (c)} \\
&\geq a\left(\prod_{t=F} x_t\right).
\end{aligned}$$

Hence $a(\prod_{t\epsilon F_i} x_t) - b(\prod_{t\epsilon F_i} x_t) > a(\prod_{t\epsilon F_j} x_t)$.

(e) Given $F\epsilon[\mathbb{N}]^{<\omega}_{>0}$, let $r = \min F$. Note

$$\begin{aligned}
a\left(\prod_{t\epsilon F} x_t\right) - b\left(\prod_{t\epsilon F} x_t\right) &\leq a(x_r) - b(x_r) \text{ by (a) and [2, Lemma 3.10(b)]} \\
&< d(x_r) \text{ by (b)} \\
&\leq d\left(\prod_{t\epsilon F} x_t\right).
\end{aligned}$$

Therefore $a(\prod_{t\epsilon F} x_t) - b(\prod_{t\epsilon F} x_t) < d(\prod_{t\epsilon F} x_t)$.

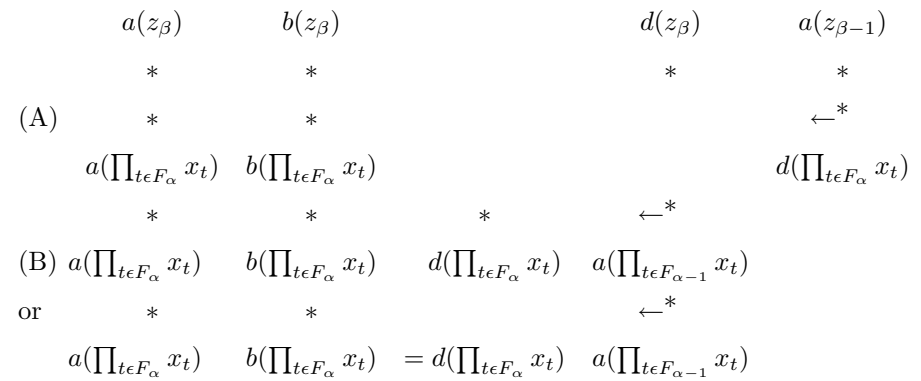
(f) For $u\epsilon[\mathbb{N}]$ pick $x\epsilon SP_u(\langle x_t\rangle_{t=1}^{\infty})$. Hence $x = \prod_{t\epsilon F_1} x_t + \prod_{t\epsilon F_2} x_t + \cdots + \prod_{t\epsilon F_u} x_t$, where $\max F_i < \min F_{i+1}$, for $i\epsilon\{1,\ldots,u-1\}$. Note that $\prod_{t\epsilon F_u} x_t \epsilon A_1$ by (a), and we have $p^{\ell(F_u)} > \prod_{t\epsilon F_u} x_t < p^{\ell(F_u)} + 2\sum_{t\epsilon F_u} p^{\ell(F_u)-(m+1)^{2t-1}\cdot 6k+1}$ by Lemma 1.15(a). Let $r = \min F_u$. Then $2p^{\ell(F_u)-(m+1)^{2r-1}\cdot 6k+1} \leq p^{\ell(F_u)-18}$ since $p \geq 2, m \geq 2$, and $r, k \geq 1$. Hence $b(\prod_{t\epsilon F_u} x_t) \leq \ell(F_u) - 18$. And since we have no carrying when adding by (d) and (e), we have $p^{\ell(F_u)} < \prod_{t\epsilon F_1} x_t + \cdots + \prod_{t\epsilon F_u} x_t < p^{\ell(F_u)+1/2}$ since $b(\prod_{t\epsilon F_u} x_t) \leq \ell(F_u) - 18$. Therefore $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1$.

(g) Given $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$, let $r = \max F$. We will show by induction on $|F|$ that $\prod_{t\epsilon F} x_t$ has at least $m+1$ nonzero digits in its base p expansion. Suppose $|F| = 1$. Then $F = \{r\}$ and $x_r = \sum_{i=1}^{m+1} p^{(l+i)(m+1)^{2r-1}\cdot 6k}$ has at least $m+1$ nonzero digits in its base p expansion. Now assume $|F| > 1$. Let $G = F\backslash\{r\}$. We have that $\prod_{t\epsilon G} x_t$ has at least $m+1$ nonzero digits in its base p expansion. Hence since $a(x_r) - b(x_r) > a(\prod_{t\epsilon G} x_t) + 2$ by (c) we have that $\prod_{t\epsilon G} x_t \cdot x_r = \prod_{t\epsilon F} x_t$ written in base p has a copy of $\prod_{t\epsilon F} x_t$ from position $a(\prod_{t\epsilon F} x_t)$ to $a(x_r)$. Therefore $\prod_{t\epsilon F} x_t$ has at least $m+1$ nonzero digits in its base p expansion.

Now pick $x\epsilon SP_u(\langle x_t\rangle_{t=1}^{\infty})$ for $u\epsilon\mathbb{N}$. Hence $x = \prod_{t\epsilon F_1} x_t + \prod_{t\epsilon F_2} x_t + \cdots + \prod_{t\epsilon F_u} x_t$, where $\max F_i < \min F_{i+1}$ for $i\epsilon\{1,\ldots,u-1\}$. Since we have no carrying when adding by (d) and (e), we have that $\prod_{t\epsilon F_1} x_t + \cdots + \prod_{t\epsilon F_u} x_t$ has at least $m+1$ nonzero digits in its base p expansion. So $x\epsilon\mathbb{N}\backslash D$. Therefore $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash D$.

(h) Pick $x\epsilon SP_n(\langle x_t\rangle_{t=1}^{\infty})$. Hence $x = \prod_{t\epsilon F_1} x_t + \cdots + \prod_{t\epsilon F_n} x_t$, where $\max F_i < \min F_{i+1}$ for $i\epsilon\{1,\ldots,n-1\}$. Suppose $x\epsilon L_2$. Then there exist $z_1, z_2, \ldots, z_m$ such that $\prod_{t\epsilon F_n} x_t + \cdots + \prod_{t\epsilon F_1} x_t = z_m + \cdots + z_1$, where $d(z_m) > a(z_m) - b(z_m) > a(z_{m-1}) > d(z_{m-1}) > a(z_{m-1}) - b(z_{m-1}) > \cdots > d(z_1) > a(z_1) - b(z_1)$.

Now pick the first α and β starting from the left where $a(\prod_{t\epsilon F} x_t) = a(z_\beta)$ and $d(\prod_{t\epsilon F_\alpha}) \neq d(z_\beta)$. This will occur since $m > n$. Hence we have case (A) where $d(\prod_{t\epsilon F_\alpha} x_t) < d(z_\beta)$ and case (B) where $d(\prod_{t\epsilon F_\alpha} x_t) > d(z_\beta)$. The following illustration of cases will be helpful, where * means a nonzero digit in base p.



Note that for $F\epsilon[\mathbb{N}]_{>0}^{<\omega}$ $\prod_{t\epsilon F} x_t$ is not a single nonzero digit in base p. Also z_β is not a single nonzero digit in base p since then one would have $d(z_\beta) = a(z_\beta) < a(z_\beta) + 1 = a(z_\beta) - b(z_\beta)$. Then in either case (A) or (B) we have $a(z_\beta) - b(z_\beta) = a(\prod_{t\epsilon F_\alpha} x_t) - b(\prod_{t\epsilon F_\alpha} x_t)$.

In case (A) we have $d(z_\beta) \leq a(z_{\beta-1}) < a(z_\beta) - b(z_\beta) = a(\prod_{t\epsilon F_\alpha} x_t) - b(\prod_{t\epsilon F_\alpha} x_t) < d(\prod_{t\epsilon F_\alpha} x_t)$, a contradiction. In case (B), we have $d(z_\beta) \leq a(\prod_{t\epsilon F_{\alpha-1}} x_t) < a(\prod_{t\epsilon F_\alpha} x_t)$

$-b(\prod_{t\epsilon F_\alpha} x_t) = a(z_\beta) - b(z_\beta) < d(z_\beta)$, a contradiction. Hence $x\epsilon\mathbb{N}\backslash L_2$. Therefore $SP_n(\langle x_t\rangle_{t=1}^\infty) \subset \mathbb{N}\backslash L_2$. □

LEMMA 1.20. *There is a sequence* $\langle x_t\rangle_{t=1}^\infty$ *such that* $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset A_1 \cap L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}$.

Proof. Pick the sequence $\langle x_t\rangle_{t=1}^\infty$ of Lemma 1.19. Then by Lemma 1.19(a) we have that $FP(\langle x_t\rangle_{t=1}^\infty) \subset A_1 \cap \mathbb{N}p^{k+1}$. By Lemma 1.19(f) we have $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset A_1$. By Lemmas 1.19(d) and (e) we have for $x\epsilon SP_m(\langle x_t\rangle_{t=1}^\infty)$ that $x\epsilon L_2$. Therefore $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset L_2$. And $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset \mathbb{N}\backslash D$ by Lemma 1.19(g). Also, since $\overline{0}\cdot m = \overline{0}$ in $\mathbb{Z}p^{k+1}$, we have that $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset A_1 \cap L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}$. □

LEMMA 1.21. *There is a sequence* $\langle x_t\rangle_{t=1}^\infty$ *such that* $SP_n(\langle x_t\rangle_{t=1}^\infty) \subset A_1 \cap \mathbb{N}\backslash L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}$.

Proof. Again pick the sequence $\langle x_t\rangle_{t=1}^\infty$ of Lemma 1.19. For $n = 1$ we have $FP(\langle x_t\rangle_{t=1}^\infty) \subset A_1 \cap \mathbb{N}p^{k+1}$ by Lemma 1.19(a). And by Lemma 1.19(f) we have $SP_n(\langle x_t\rangle_{t=1}^\infty) \subset A_1$. By Lemma 1.19(h) we have that $SP_n(\langle x_t\rangle_{t=1}^\infty) \subset \mathbb{N}\backslash L_2$. And $SP_n(\langle x_t\rangle_{t=1}^\infty) \subset \mathbb{N}\backslash D$ by Lemma 1.19(g). Also since $\overline{0}\cdot n = \overline{0}$ in $\mathbb{Z}p^{k+1}$ we have that $SP_n(\langle x_t\rangle_{t=1}^\infty) \subset A_1 \cap \mathbb{N}\backslash L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}$. □

LEMMA 1.22. *The sequence where* $x_t = p^{3\cdot 2^{2t-1}\cdot 4k}$ *for* $t\epsilon\mathbb{N}$ *has the following properties:*

(a) $FP(\langle x_t\rangle_{t=1}^\infty) \subset A_0 \cap \mathbb{N}p^{k+1}$,

(b) $a(x_{t+1}) > a(\prod_{\ell=1}^t x_\ell) + 8$ *for* $t\epsilon\mathbb{N}$,

(c) $d(\prod_{t\epsilon F_i} x_t) > a(\prod_{t\epsilon F_j} x_t) + 8$ *for* $F_i, F_j[\mathbb{N}]^{<\omega}_{>0}$ *and* $\max F_j < \min F_i$,

(d) $SP_u(\langle x_t\rangle_{t=1}^\infty) \subset A_1$ *for* $u\epsilon[\mathbb{N}]\backslash\{1\}$,

(e) $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset D$,

(f) $SP_m(\langle x_t\rangle_{t=1}^\infty) \subset \mathbb{N}\backslash L_2$.

Proof. (a) Given $F\epsilon[\mathbb{N}]^{<\omega}_{>0}$, clearly $\prod_{t\epsilon F} x_t\epsilon A_0 \cap \mathbb{N}p^{k+1}$ since $x_t\epsilon A_0 \cap \mathbb{N}p^{k+1}$ for all $t\epsilon\mathbb{N}$. Hence $FP(\langle x_t\rangle_{t=1}^\infty) \subset A_0 \cap \mathbb{N}p^{k+1}$.

(b)

$$\begin{aligned}
a\left(\prod_{t=1}^{\ell} x_t\right) &= \sum_{t=1}^{\ell} a(x_t) \text{ due to (a)} \\
&= \sum_{t=1}^{\ell} 3\cdot 2^{2t-1}\cdot 4k \\
&= \frac{3}{2}\cdot 4k\cdot \sum_{t=1}^{\ell} 4^t \\
&= \frac{3}{2}\cdot 4k\frac{4^{\ell+1}-4}{3} \\
&= 4k\cdot\frac{4^{\ell+1}-4}{2} \\
&= 4k(2\cdot 4^\ell - 2) \\
&= 4k(2^{2\ell+1}-2) \\
&< 3\cdot 2^{2\ell+1}\cdot 4k - 8 \\
&= a(x_{\ell+1}) - 8.
\end{aligned}$$

Hence $a(x_{\ell+1}) > a(\prod_{t=1}^\ell x_t) + 8$.

(c) Pick $F_i, F_j\epsilon[\mathbb{N}]^{<\omega}_{>0}$ where $\max F_j < \min F_i$. Let $r = \min F_i$.

Therefore

$$\begin{aligned} d\left(\prod_{t\epsilon F_i} x_t\right) &= a\left(\prod_{t\epsilon F_i} x_t\right) && \text{due to (a)} \\ &\geq a(x_r) \\ &> a\left(\prod_{t=1}^{r-1} x_t\right) + 8 \\ &\geq a\left(\prod_{t\epsilon F_j} x_t\right) = 8 \end{aligned}$$

(d) For $u\epsilon\mathbb{N}\backslash\{1\}$ pick $x\epsilon SP_u(\langle x_t\rangle_{t=1}^{\infty})$. Hence $x = \prod_{t\epsilon F_1} x_t + \prod_{t\epsilon F_2} x_t + \cdots + \prod_{t\epsilon F_u} x_t$, where $\max F_i < \min F_{i+1}$ for $i\epsilon\{1,\ldots,u-1\}$. Note $\prod_{t\epsilon F_i} x_t A_0$ for $i\epsilon\{1,\ldots,u\}$ by (a) and let $\ell = a(\prod_{t\epsilon F_u} x_t)$. Since we have no carrying when adding in base p by (c), we have that $p^{\ell} < \prod_{t\epsilon F_1} x_t + \cdots + \prod_{t\epsilon F_u} x_t < p^{\ell+1/2}$ since $a(\prod_{t\epsilon F_{u-1}} x_t) < \ell - 8$ by (c). Therefore $SP_u(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1$.

(e) Pick $x\epsilon SP_m(\langle x_t\rangle_{t=1}^{\infty})$. Hence $x = \prod_{t\epsilon F_1} x_t + \cdots + \prod_{t\epsilon F_m} x_t$ where $\max F_i < \min F_{i+1}$ for $i\epsilon\{1,\ldots,m-1\}$. Note again that $\prod_{t\epsilon F_i} x_t\epsilon A_0$ for $i\epsilon\{1,\ldots,m\}$ by (a). Hence $\prod_{t\epsilon F_1} x_t + \cdots + \prod_{t\epsilon F_m} x_t$ has m ones in its base p expansion since there is no carrying when adding by (c). Therefore $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset D$.

(f) Pick $x\epsilon SP_m(\langle x_t\rangle_{t=1}^{\infty})$. By (e) x has exactly m nonzero digits in its base p expansion. Recall that $L_2 = \sum_{t-1}^{m} z_t$: for each $t\epsilon\{1,2,\ldots,m\}, d(z_t) > a(z_t) - b(z_t)$ and if $t > 1, a(z_t) - b(z_t) > a(z_{t-1})\}$. And note that if $\sum_{t=1}^{m} z_t$ is as in the definition of L_2, then z_t does not have an expansion consisting of a single nonzero digit, since then one would have $d(z_t) = a(z_t) < a(z_t) + 1 = a(z_t) - b(z_t)$. Therefore the elements of L_2 have at least $2m$ nonzero digits. Hence $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash L_2$. □

LEMMA 1.23. *There is a sequence $\langle x_t\rangle_{t=1}^{\infty}$ such that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1 \cap \mathbb{N}\backslash L_2 \cap D \cap \mathbb{N}p^{k+1}$.*

Proof. Pick the sequence $\langle x_t\rangle_{t=1}^{\infty}$ of Lemma 1.19. Then by Lemma 1.22(a) we have that $FP(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}p^{k+1}$. And by Lemma 1.22(d) we have $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1$. We have that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset D$ by Lemma 1.22(e). Also $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}\backslash L_2$ by Lemma 1.22(f). And since $\overline{0}\cdot\overline{0}$ in $\mathbb{Z}p^{k+1}$ we have that $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset A_1 \cap \mathbb{N}\backslash L_2 \cap D \cap \mathbb{N}p^{k+1}$. □

LEMMA 1.24. *The set $A_1 \cap L_2 \cap D \cap \mathbb{N}p^{k+1} = \emptyset$.*

Proof. We need only show that $L_2 \cap D = \emptyset$. Recall from the definition of L_2 that the elements of L_2 must have at least $2m$ nonzero digits in their base p expansion, whereas the members of D have exactly m nonzero digits in their base p expansion. Hence $A_1 \cap L_2 \cap D \cap \mathbb{N}p^{k+1} = \emptyset$. □

THEOREM 1.25. *Let m and n be in $\mathbb{N}$ with $m > n$. Then there is a partition $\mathbb{N} = N \cup M$ such that whenever $\langle x_t\rangle_{t=1}^{\infty}$ is a sequence in $\mathbb{N}$ with $SP_m(\langle x_t\rangle_{t=1}^{\infty})[SP_n(\langle x_t\rangle_{t=1}^{\infty})]$ monochrome with respect to $\{N, M\}$ one has $SP_m(\langle x_t\rangle_{t=1}^{\infty}) \subset M[SP_n(\langle x_t\rangle_{t=1}^{\infty}) \subset \mathbb{N}]$.*

Proof. Now let $B_1 = \mathbb{N}\backslash\mathbb{N}p^k, B_2 = A_0 \cap \mathbb{N}p^{k+1}, B_3 = A_2 \cap \mathbb{N}p^{k+1}, B_4 = A_3 \cap \mathbb{N}p^{k+1}, B_5 = A_4 \cap \mathbb{N}\backslash L_1 \cap \mathbb{N}p^{k+1}, B_6 = A_1 \cap L_2 \cap D \cap \mathbb{N}p^{k+1}$ and $B_7 = A_1 \cap \mathbb{N}\backslash L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}, B_8 = \mathbb{N}p^k\backslash\mathbb{N}p^{k+1}, B_9 = A_4 \cap L_1 \cap \mathbb{N}p^{k+1}, B_{10} = A_1 \cap L_2 \cap \mathbb{N}\backslash D \cap \mathbb{N}p^{k+1}$, and $B_{11} = A_1 \cap \mathbb{N}\backslash L_2 \cap D \cap \mathbb{N}p^{k+1}$. Let $N = B_1 \cup B_2 \cup B_3 \cup B_4 \cup B_5 \cup B_6 \cup B_7$ and let $M' = B_8 \cup B_9 \cup B_{10} \cup B_{11}$. Let $M = (M'\backslash\{n\}) \cup \{m\}$ and $N = (N'\backslash\{m\}) \cup \{n\}$. Let sequences $\langle x_t\rangle_{t=1}^{\infty}$ and $\langle y_t\rangle_{t=1}^{\infty}$ be given with $SP_m(\langle x_t\rangle_{t=1}^{\infty})$ and $SP_n(\langle y_t\rangle_{t=1}^{\infty})$ both monochrome with respect to $\{M, N\}$.

If infinitely often (or in fact at least m times) $x_1 = 1$, then $m \epsilon SP_m(\langle x_t \rangle_{t=1}^{\infty}) \cap M$; so, since $SP_m(\langle x_t \rangle_{t=1}^{\infty})$ is monochrome, $SP_m(\langle x_t \rangle_{t=1}^{\infty}) \subseteq M$. Similarly, if infinitely often $y_t = 1$, one has $SP_n(\langle x_t \rangle_{t=1}^{\infty}) \subseteq N$. Thus we may assume, by passing to subsystems, that both $\langle x_t \rangle_{t=1}^{\infty}$ and $\langle y_t \rangle_{t=1}^{\infty}$ are increasing sequences.

By [2, Corollary 2.5] we may assume (by passing to subsystems) that we have i and j in $\{1, 2, \ldots, 11\}$ such that $SP_m(\langle x_t \rangle_{t=1}^{\infty}) \subset B_i$ and $SP_n(\langle y_t \rangle_{t=1}^{\infty}) \subseteq B_j$. We need to show that $i \epsilon \{8, 9, 10, 11\}$ and $j \epsilon \{1, 2, 3, 4, 5, 6, 7\}$.

Now i is not any of $1, 2, 3$, or 4 by Lemmas 1.11, 1.6, 1.9, and 1.10, respectively. Also $i \neq 5$ by Lemma 1.18 and Theorem 1.4, $i \neq 7$ by Lemma 1.21 and Theorem 1.4, and $i \neq 6$ by Lemma 1.24.

We have $j \neq 8$ by Lemma 1.13. Also $j \neq 9$ by Lemma 1.16 and Theorem 1.4, $j \neq 10$ by Lemma 1.20 and Theorem 1.4, and $j \neq 11$ by Lemma 1.23 and Theorem 1.4. □

REFERENCES

[1] N. Hindman, *Partitions and sums and products—Two counterexamples*, J. Combin. Theory Ser. A, 29 (1980), pp. 113–120.

[2] G. Smith, *Partitions and (m and n) sums of products*, J. Combin. Theory Ser. A, 72 (1995), pp. 77–94.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 370–380

MINIMUM SPAN OF NO-HOLE $(r+1)$-DISTANT COLORINGS*

GERARD J. CHANG†, JUSTIE SU-TZU JUAN‡, AND DAPHNE DER-FEN LIU§

Abstract. Given a nonnegative integer r, a no-hole $(r+1)$-distant coloring, called N_r-coloring, of a graph G is a function that assigns a nonnegative integer (color) to each vertex such that the separation of the colors of any pair of adjacent vertices is greater than r, and the set of the colors used must be consecutive. Given r and G, the minimum N_r-span of G, $\mathrm{nsp}_r(G)$, is the minimum difference of the largest and the smallest colors used in an N_r-coloring of G if there exists one; otherwise, define $\mathrm{nsp}_r(G) = \infty$. The values of $\mathrm{nsp}_1(G)$ $(r = 1)$ for bipartite graphs are given by Roberts [*Math. Comput. Modelling*, 17 (1993), pp. 139–144]. Given $r \geq 2$, we determine the values of $\mathrm{nsp}_r(G)$ for all bipartite graph with at least $r-2$ isolated vertices. This leads to complete solutions of $\mathrm{nsp}_2(G)$ for bipartite graphs.

Key words. vertex-coloring, no-hole $(r+1)$-distant coloring, minimum span, bipartite graphs

AMS subject classification. 05C78

PII. S0895480198339456

1. Introduction.

The T-coloring of graphs models the *channel assignment problem* introduced by Hale [6] in communication networks. In the channel assignment problem, several transmitters and a forbidden set T (called *T-set*) of nonnegative integers with $0 \in T$ are given. We assign a nonnegative integral channel to each transmitter under the constraint that if two transmitters interfere, the difference of their channels does not fall within the given T-set. Two transmitters may interfere due to various reasons such as geographic proximity and meteorological factors. To formulate this problem, we construct a graph G such that each vertex represents a transmitter, and two vertices are adjacent if their corresponding transmitters interfere.

Thus, we have the following definition. Given a T-set and a graph G, a *T-coloring* of G is a function $f : V(G) \to Z^+ \cup \{0\}$ such that

$$|f(x) - f(y)| \notin T \text{ if } xy \in E(G).$$

Note that if $T = \{0\}$, then T-coloring is the same as ordinary vertex-coloring.

A *no-hole T-coloring* of a graph G is a T-coloring f of G such that the set $\{f(v) : v \in V(G)\}$ is consecutive (the no-hole assumption). When $T = \{0, 1\}$ and $T = \{0, 1, 2, \ldots, r\}$, a no-hole T-coloring is also called an *N-coloring* [16] and an *N_r-coloring* (or *no-hole $(r+1)$-distant coloring*) [17], respectively. That is, an N_r-coloring of a graph G is a vertex coloring $f : V(G) \to Z^+ \cup \{0\}$ such that the following two conditions are satisfied:

- $|f(x) - f(y)| \geq r + 1$ if $uv \in E(G)$;

*Received by the editors May 27, 1998; accepted for publication (in revised form) June 7, 2001; published electronically August 29, 2001.

http://www.siam.org/journals/sidma/14-3/33945.html

†Department of Applied Mathematics, National Chiao Tung University, Hsinchu 30050, Taiwan (gjchang@math.nctu.edu.tw). The research of this author was supported in part by the National Science Council under grant NSC87-2115-M009-007 and the Lee and MTI Center for Networking Research at NCTU.

‡Department of Computer Science and Information Engineering, National Chi Nan University, 1, University Road, Puli, Nantou 545, Taiwan (jsjuan@csie.ncnu.edu.tw).

§Department of Mathematics, California State University, Los Angeles, CA 90032 (dliu@calstatela.edu). The research of this author was supported in part by the National Science Foundation under grant DMS-9805945.

• the set $\{f(v) : v \in V(G)\}$ is consecutive.

In terms of efficiency of the usage of the channels (colors), the variable T-span has been considered. The *span* of a T-coloring f is the difference of the largest and the smallest colors used in $f(V)$; the *T-span* of a graph G, $\mathrm{sp}_T(G)$, is the minimum span among all T-colorings of G.

The T-spans for different families of graphs and for different T-sets have been studied extensively (see [3, 4, 5, 10, 11, 12, 14, 15, 18]). It is known [3, 10] that if T is an *r-initial set*, that is, $T = \{0, 1, 2, \ldots, r\} \cup A$ where A is a set of integers without multiples of $(r+1)$, then the following holds for all graphs:

$$(*) \qquad \mathrm{sp}_T(G) = (\chi(G) - 1)(r+1),$$

where $\chi(G)$, the *chromatic number* of G, is the minimum number of colors to properly color vertices of G.

It is known [3] and not difficult to learn that for any given T-set and any graph G, a T-coloring always exists. However, a no-hole T-coloring does not always exist. For instance, as $T = \{0, 1\}$, then K_n, the complete graph with n vertices, does not have a no-hole T-coloring for any $n \geq 2$.

The minimum span of a no-hole T-coloring for a graph G is denoted by $\mathrm{nsp}_T(G)$. If there does not exist a no-hole T-coloring for G, then $\mathrm{nsp}_T(G) = \infty$. If $T = \{0, 1, 2, \ldots, r\}$, denote $\mathrm{nsp}_T(G)$ by $\mathrm{nsp}_r(G)$.

A no-hole T-coloring is also a T-coloring. Hence by $(*)$, a natural lower bound for $\mathrm{nsp}_r(G)$ is $(\chi(G)-1)(r+1)$. Roberts [16] and Sakai and Wang [17] studied N-coloring and N_r-coloring, respectively. Among the findings in [16, 17] are the results about the existence of an N-coloring and an N_r-coloring for several families of graphs including paths, cycles, bipartite graphs, and 1-unit sphere graphs. The authors also compare the span of such a coloring (if there exists one) with the lower bound $(\chi(G)-1)(r+1)$. The N-colorings and N_r-colorings studied in [16, 17] are not necessarily optimal; i.e., the spans are not always the minimum.

This article focuses on the exact values of the minimum N_r-span, $\mathrm{nsp}_r(G)$, especially for bipartite graphs, i.e., graphs with $\chi(G) \leq 2$. In section 2, we give preliminary results for general graphs. In section 3, we explore the values of $\mathrm{nsp}_r(G)$ for bipartite graphs. The solutions of $\mathrm{nsp}_1(G)$ for bipartite graphs are given by Roberts [16]. We determine the values of $\mathrm{nsp}_r(G)$ for any bipartite graph G with at least $r-2$ isolated vertices. This result also leads to a complete description of the values of $\mathrm{nsp}_2(G)$ for all bipartite graphs.

2. Preliminary results. In this section, we present several results about the minimum N_r-span for general graphs. We show a number of upper and lower bounds of $\mathrm{nsp}_r(G)$ for different types of graphs. In order to find the minimum span, without loss of generality, we assume that the color 0 is always used in any N_r-coloring.

If $|V(G)| = n$ and $\mathrm{nsp}_T(G) < \infty$, then by definition a trivial upper bound for $\mathrm{nsp}_T(G)$ is $n-1$. On the other hand, any no-hole T-coloring is also a T-coloring, hence we have the following proposition.

PROPOSITION 2.1. *For any T-set and any graph G with n vertices,* $\mathrm{sp}_T(G) \leq \mathrm{nsp}_T(G)$*; and* $\mathrm{nsp}_T(G) \leq n-1$ *if* $\mathrm{nsp}_T(G) < \infty$.

Combining Proposition 2.1 and $(*)$, we have the following proposition.

PROPOSITION 2.2. *For any $r \in Z^+$ and any graph G with chromatic number $\chi(G)$,* $(\chi(G)-1)(r+1) \leq \mathrm{nsp}_r(G)$.

With the following result, we show a lower bound of $\mathrm{nsp}_r(G)$ in terms of r and the number of isolated vertices in G.

THEOREM 2.3. *Suppose $r \in Z^+$ and G is a graph with i isolated vertices, $i \geq 0$, and at least one edge. Then* $\mathrm{nsp}_r(G) \geq \max\{2r - i + 1, r + 1\}$.

Proof. It suffices to show the result when $\mathrm{nsp}_r(G)$ is finite. Because G has at least one edge, $\mathrm{nsp}_r(G) \geq r + 1$. Thus the lemma holds if $i \geq r$.

Suppose $i < r$. Let f be an optimal N_r-coloring of G. By the no-hole assumption of an N_r-coloring, the colors $r, r-1, \ldots, 2, 1, 0$, must be used by some vertices. Since G has only i isolated vertices and $i < r$, there exists a nonisolated vertex u with $r - i \leq f(u) \leq r$. Because u is nonisolated, there exists some vertex v such that $uv \in E(G)$. Then $f(v) \geq f(u)$, for otherwise $0 \leq f(u) - f(v) \leq r$, a contradiction to $uv \in E(G)$. Therefore, we have

$$f(v) \geq f(u) + r + 1 \geq r - i + r + 1 = \max\{2r - i + 1, r + 1\}.$$

This implies $\mathrm{nsp}_r(G) \geq \max\{2r - i + 1, r + 1\}$. □

The *union* of two *vertex-disjoint* graphs G and H, denoted by $G \cup H$, is the graph with vertex set $V(G \cup H) = V(G) \cup V(H)$ and edge set $E(G \cup H) = E(G) \cup E(H)$. For the case in which H has exactly one vertex x, $G \cup H$ is denoted by $G \cup \{x\}$.

The inequality $\mathrm{nsp}_r(G) \leq \mathrm{nsp}_r(G \cup H)$ does not always hold. For instance, if $G = K_2$, then $\mathrm{nsp}_1(G) = \infty$, while $\mathrm{nsp}_1(G \cup \{x\}) = 2$. In the rest of the section, we present several results on unions of graphs.

THEOREM 2.4. *Suppose G is a graph with at least one edge; then* $\mathrm{nsp}_{r+1}(G \cup \{x\}) \geq \mathrm{nsp}_r(G) + 1$.

Proof. It suffices to show the result when $\mathrm{nsp}_{r+1}(G \cup \{x\})$ is finite. Suppose f is an N_{r+1}-coloring of $G \cup \{x\}$. Define a coloring g on $V(G)$ by

$$g(v) = \begin{cases} f(v) & \text{if } f(v) < f(x) \text{ or } f(v) = 0, \\ f(v) - 1 & \text{if } f(v) \geq f(x) \text{ and } f(v) > 0. \end{cases}$$

It is straightforward to verify that g is an N_r-coloring of G and the span of g is one less than the span of f. Therefore, $\mathrm{nsp}_{r+1}(G \cup \{x\}) \geq \mathrm{nsp}_r(G) + 1$. □

THEOREM 2.5. *Suppose G is a graph with* $\mathrm{nsp}_r(G) = q(r + 1) + j$, *where $q \geq 1$ and $0 \leq j \leq r$, and H is a graph with q vertices. Then* $\mathrm{nsp}_{r+1}(G \cup H) \leq \mathrm{nsp}_r(G) + q$.

Proof. It suffices to show the result when $\mathrm{nsp}_r(G) < \infty$. Let f be an optimal N_r-coloring of G and $f(V(G)) = \{0, 1, \ldots, \mathrm{nsp}_r(G)\}$. Suppose $V(H) = \{x_1, x_2, \ldots, x_q\}$. Define a coloring g on $G \cup H$, $g : V(G \cup H) \to Z^+ \cup \{0\}$, by

$$g(v) = \begin{cases} \lfloor \frac{(r+2)f(v)}{r+1} \rfloor & \text{if } \ v \in V(G), \\ k(r + 2) - 1 & \text{if } \ v = x_k \in V(H). \end{cases}$$

It is enough to show that g is an N_{r+1}-coloring for $G \cup H$. Because f is onto, therefore $g(V(G \cup H))$ is a consecutive set; indeed $g(V(G \cup H)) = \{0, 1, 2, \ldots, \mathrm{nsp}_r(G) + q\}$. If $uv \in E(G \cup H)$, then either $uv \in E(G)$ or $uv \in E(H)$. If $uv \in E(H)$, then $|g(u) - g(v)| \geq r + 2$. If $uv \in E(G)$, without loss of generality, assume $f(u) > f(v)$. Since $f(u) - f(v) \geq r + 1$, we have $\frac{(r+2)f(u)}{r+1} - \frac{(r+2)f(v)}{r+1} \geq r + 2$, so $g(u) - g(v) \geq r + 2$. Hence g is an N_{r+1}-coloring with span $\mathrm{nsp}_r(G) + q$. This completes the proof. □

Note that the result in Theorem 2.5 is not always true if the assumption $\mathrm{nsp}_r(G) = q(r + 1) + j$ does not hold. For instance, let $G = K_2 \cup rK_1$ and $H = K_3$; then $\mathrm{nsp}_r(G) = r + 1$ for any r. However, $\mathrm{nsp}_{r+1}(G \cup H) = \infty$ for any $r \geq 4$.

COROLLARY 2.6. *If G is a graph with $r + 1 \leq \mathrm{nsp}_r(G) \leq 2r + 1$, then* $\mathrm{nsp}_{r+1}(G \cup \{x\}) = \mathrm{nsp}_r(G) + 1$.

Proof. The corollary follows from Theorems 2.4 and 2.5. ∎

Consider the graph G in Figure 2.1. According to Theorem 2.3, $\mathrm{nsp}_1(G) \geq 3$ and so the labeling in the figure gives that $\mathrm{nsp}_1(G) = 3$. According to Corollary 2.6, we have $\mathrm{nsp}_2(G \cup \{x\}) = \mathrm{nsp}_1(G) + 1 = 4$.

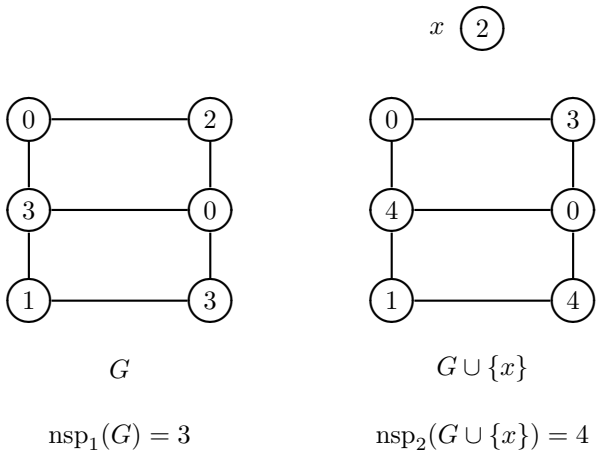


FIG. 2.1. *Optimal N-coloring for G and optimal N_2-coloring for $G \cup \{x\}$.*

3. Main results. In this section, we explore the minimum N_r-span for bipartite graphs. It turns out that the number of isolated vertices in a bipartite graph plays a key role for this problem. We give the values of $\mathrm{nsp}_r(G)$ for all bipartite graphs G with at least $r-2$ isolated vertices. This result leads to complete solutions of $\mathrm{nsp}_2(G)$ for all bipartite graphs G.

In this section, a bipartite graph is conventionally denoted by $G = (A, B, I, E)$, where I is the set of all isolated vertices and (A, B) is a *bipartition* of all nonisolated vertices such that each edge in G has one end in A and the other in B. A vertex v is called an *A-, B- or I-vertex* if $x \in A, B$, or I, respectively.

The *bipartite-complement* $\widehat{G}$ of a bipartite graph $G = (A, B, I, E)$ with $E \neq \emptyset$ is the bipartite graph $\widehat{G}$ with vertex set $V(\widehat{G}) = A \cup B$ and edge set

$$E(\widehat{G}) = \{ab : a \in A, b \in B, ab \notin E\}.$$

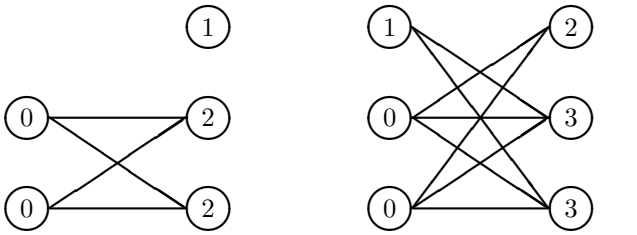
Note that the set of isolated vertices in $\widehat{G}$ is not specified in the notation. Moreover, we shall denote B' the set of all B-vertices not adjacent to any A-vertex in $\widehat{G}$. If G is a bipartite graph, then $\widehat{G}$ is a subgraph of G^c, the *complement* of G (i.e., $V(G^c) = V(G)$ and $E(G^c) = \{uv : u \neq v \text{ and } uv \notin E(G)\}$).

The N_1-coloring for bipartite graphs has been studied by Roberts [16]. Although the concept of the *minimum* N_1-span was not introduced explicitly in [16], the following theorem, which completely determines the values of $\mathrm{nsp}_1(G)$ for bipartite graphs, can be generated from [16].

THEOREM 3.1 (see Roberts [16]). *If $G = (A, B, I, E)$ is a bipartite graph with $E(G) \neq \emptyset$, then*

$$\mathrm{nsp}_1(G) = \begin{cases} 2 & \text{if } |I| \geq 1, \\ 3 & \text{if } |I| = 0 \text{ and } E(\widehat{G}) \neq \emptyset, \\ \infty & \text{if } |I| = 0 \text{ and } E(\widehat{G}) = \emptyset. \end{cases}$$

As examples to Theorem 3.1, consider the graphs G_1 and G_2 in Figure 3.1. As $|I| \geq 1$ for G_1, we have $\mathrm{nsp}_1(G_1) = 2$. For G_2, the facts $|I| = 0$ and $E(\widehat{G}) \neq \emptyset$ imply $\mathrm{nsp}_2(G_2) = 3$.



$G_1 : |I| \geq 1$ $\qquad$ $G_2 : |I| = 0$ and $E(\widehat{G}) \neq \emptyset$

$\mathrm{nsp}_1(G_1) = 2$ $\qquad$ $\mathrm{nsp}_1(G_2) = 3$

FIG. 3.1. *Two examples of optimal N-colorings for bipartite graphs.*

Sakai and Wang [17] characterize the existence of an N_r-coloring by using the Hamiltonian r-path. The *d-path* on n vertices, $v_1, v_2, \ldots, v_n$, has the edge set $\{v_i v_j : 1 \leq |i-j| \leq d\}$. Figure 3.2 shows a 2-path with seven vertices. A 1-path on n vertices is an ordinary path denoted as P_n. A *Hamiltonian d-path* of a graph G is a d-path covering each vertex of G exactly once.

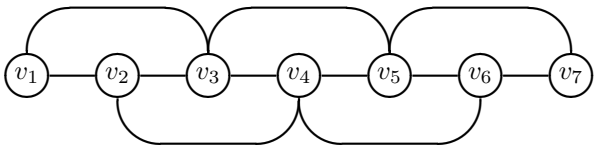


FIG. 3.2. *A 2-path with seven vertices.*

THEOREM 3.2 (see Sakai and Wang [17]). *G has an N_r-coloring if and only if G^c has a Hamiltonian r-path. Indeed, if f is an N_r-coloring such that $f(v_1) \leq f(v_2) \leq \ldots \leq f(v_n)$, then $v_1, v_2, \ldots, v_n$ is a Hamiltonian r-path in G^c.*

If the lower bound of $\mathrm{nsp}_r(G)$ in Theorem 2.3 is attained by some graph G, according to Proposition 2.2, G must be bipartite. Such graphs do exist. In the next theorem, we show a sufficient condition for such graphs.

THEOREM 3.3. *Suppose $G = (A, B, I, E)$ is a bipartite graph with at least one edge. If $|I| \geq r$, then $\mathrm{nsp}_r(G) = r+1$; if $|I| \leq r-1$ and there exist $\{a_1, a_2, \ldots, a_{r-|I|}\} \subseteq A$ and $\{b_1, b_2, \ldots, b_{r-|I|}\} \subseteq B$ such that $a_j b_k \notin E(G)$ for $j + k \geq r - |I| + 1$, then $\mathrm{nsp}_r(G) = 2r - |I| + 1$.*

Proof. It is obvious that $\mathrm{nsp}_r(G) \geq r + 1$, since $E(G) \neq \emptyset$.

If $|I| \geq r$, coloring A-vertices with 0, B-vertices with $r + 1$, and I-vertices with $1, 2, \ldots, r$ gives an N_r-coloring. Therefore, $\mathrm{nsp}_r(G) = r + 1$.

If $|I| \leq r - 1$, by Theorem 2.3, $\mathrm{nsp}_r(G) \geq 2r - |I| + 1$. Hence it suffices to find

an N_r-coloring with span at most $2r-|I|+1$. Define a coloring by the following four steps:

(1) color $a_1, a_2, \ldots, a_{r-|I|}$ with $1, 2, \ldots, r-|I|$, respectively;
(2) color I-vertices with $r-|I|+1, r-|I|+2, \ldots, r$;
(3) color $b_{r-|I|}, b_{r-|I|-1}, \ldots, b_1$ with $r+1, r+2, \ldots, 2r-|I|$, respectively; and
(4) color all the remaining A-vertices with 0 and B-vertices with $2r-|I|+1$.

By the assumption that $a_j b_k \notin E(G)$ for $j+k \geq r-|I|+1$, it is easy to verify that the coloring defined above is an N_r-coloring with span at most $2r-|I|+1$. □

COROLLARY 3.4. *Let $G=(A,B,I,E)$ be a bipartite graph with at least one edge.*

(a) *If $|I| \leq r-1$ and $E(\widehat{G})=\emptyset$, then $\mathrm{nsp}_r(G)=\infty$.*
(b) *If $|I|=r-1$, then $\mathrm{nsp}_r(G)=r+2$ if and only if $E(\widehat{G})\neq\emptyset$.*
(c) *If $|I|=r-2$ and there exists a P_4 in $\widehat{G}$, then $\mathrm{nsp}_r(G)=r+3$.*

Proof. We need only to show (a), since (b) and (c) follow from Theorem 3.3.

Suppose $|I| \leq r-1$ and $E(\widehat{G})=\emptyset$. Then, $G-I$ is a complete bipartite graph $K_{|A|,|B|}$. Combining this with the assumption that $|I| \leq r-1$, G does not admit any N_r-coloring, so $\mathrm{nsp}_r(G)=\infty$. □

Combining Theorem 3.3 and Corollary 3.4(b), the values of $\mathrm{nsp}_r(G)$ for bipartite graphs with at least $r-1$ isolated vertices are settled. In the rest of the article, we shall focus on the N_r-coloring for bipartite graphs $G=(A,B,I,E)$ with at most $r-2$ isolated vertices. By Corollary 3.4(a), we may assume $2 \leq |A| \leq |B|$. In the rest of the section, we search for the exact value of $\mathrm{nsp}_r(G)$ to complete the case as $|I|=r-2$. By Corollary 3.4(c), it suffices to consider the case that $\widehat{G}$ contains no P_4. We first show a lemma which is a key to settle this problem.

For any real number x, denote $\max\{x,0\}$ by x^+. For any two integers a and b, $a \leq b$, let $[a,b]$ denote the set $\{a, a+1, a+2, \ldots, b\}$.

LEMMA 3.5. *Let $G=(A,B,I,E)$ be a bipartite graph with $2 \leq m=|A| \leq |B|$, $|I| \leq r-2$, and $\widehat{G}$ contains no P_4. If $\mathrm{nsp}_r(G)<\infty$, then the following are all true:*

(a) *In the graph $\widehat{G}$, every B-vertex is adjacent to at most one A-vertex.*
(b) *There exist an arrangement $\Pi=(A_1,A_2,\ldots,A_m)$ of A and nonnegative integers $d_1=0, c_1, d_2, c_2, d_3, \ldots, d_m, c_m=0$ such that $\deg_{\widehat{G}}(A_k)=d_k+c_k$ for $1 \leq k \leq m$ and $|I| \geq q(\Pi):=\sum_{k=1}^{m-1} q_k$, where $q_k=\max\{(r-c_k)^+,(r-d_{k+1})^+\}$.*
(c) *$\mathrm{nsp}_r(G) \geq (m-1)(2r+1)-|I|$.*
(d) *If $B' \neq \emptyset$, then $|I|-q(\Pi) \geq q'(\Pi):=\min_{1\leq k\leq m-1} q'_k$, where $q'_k=\min\{(r-c_k)^+,(r-d_{k+1})^+\}$.*
(e) *If $B' \neq \emptyset$, then $\mathrm{nsp}_r(G) \geq \max\{2r+2,(m-1)(2r+1)-|I|+s(\Pi)+1\}$, where $s(\Pi)=\min_{1\leq k\leq m-1}\{q_k : q'_k \leq |I|-q(\Pi)\}$.*

Proof. Suppose f is an optimal N_r-coloring for G. According to Theorem 3.2, G^c has a Hamiltonian r-path $P=v_1,v_2,\ldots,v_{|V(G)|}$ with $0=f(v_1) \leq f(v_2) \leq \cdots \leq f(v_{|V(G)|})$. Without loss of generality, we assume the order of A-vertices on the r-path P is $\Pi=(A_1,A_2,\ldots,A_m)$. We call this an *arrangement* of A. Hence $f(A_1) \leq f(A_2) \leq \cdots \leq f(A_m)$.

On P, let an A- (or B-) *run* be a maximal interval of consecutive $A \cup I$- (or $B \cup I$-) vertices, starting and ending with A- (or B-) vertices. Note that there may exist some I-vertices within one run or between two consecutive runs; and the runs are alternating between A and B.

It is impossible to have two consecutive runs with at least two vertices in each. For if it is possible, then there exist $x,y \in A$ and $z,w \in B$ whose order in P is (x,y,z,w), and the vertices between x and w, other than y and z, are I-vertices.

Because $|I| \le r-2$, $(x-z-y-w)$ forms a P_4 in $\widehat{G}$, a contradiction.

Analogously it is impossible to have two consecutive singleton runs (except possibly the first run and the last run). For if it is possible, then we get a P_4 in $\widehat{G}$ by connecting the two consecutive singleton A-run and B-run with the B-vertex and A-vertex before and after them.

We conclude that either all A-runs or all B-runs are singletons. As $|A| \le |B|$, all A-runs are singletons and each B-run (except possibly the first run and/or the last run) contains at least two vertices. Therefore between any A_k and A_{k+1} on P, there are only B- or I-vertices. Since $|I| \le r-2$ and P is an Hamiltonian r-path in G^c, there exist at least two B-vertices between A_k and A_{k+1} that are adjacent to A_k.

To prove (a), suppose to the contrary that there exists $v \in B$ such that $vA_k, vA_\ell \in E(\widehat{G})$ for some $k < \ell$. Then between A_k and A_ℓ on P there exists $u \in B - \{v\}$ adjacent to A_k in $\widehat{G}$. Thus $(u - A_k - v - A_\ell)$ forms a P_4 in $\widehat{G}$, a contradiction. This proves (a).

Claim. For all $1 \le k \le m-1$, we have $f(A_{k+1}) - f(A_k) \ge r+2$.

Proof of claim. Suppose $f(A_{k+1}) - f(A_k) \le r+1$ for some k. Then the B-vertices between A_k and A_{k+1} on P are adjacent to both A_k and A_{k+1} in $\widehat{G}$, contradicting (a).

Note that if $A_1 = v_i$, then $P' = v_i, v_{i-1}, \ldots, v_2, v_1, v_{i+1}, v_{i+2}, \ldots, v_{|V(G)|}$ is also a Hamiltonian r-path in G^c, or, equivalently, f' defined by $f'(v_j) = f(v_{1+i-j})$ for $1 \le j \le i$ and $f'(v_j) = f(v_j)$ for $i < j \le |V(G)|$ is also an optimal N_r-coloring of G. Therefore, without loss of generality, we may assume $A_1 = v_1$. Similarly, we may assume that $A_m = v_{|V(G)|}$. Put

$D_1 := \{y \in B : yA_1 \in E(\widehat{G}) \text{ and } f(y) < f(A_1)\}$ and $d_1 := |D_1|$,
$C_1 := \{x \in B : xA_1 \in E(\widehat{G}) \text{ and } f(A_1) \le f(x)\}$ and $c_1 := |C_1|$,
$D_k := \{y \in B : yA_k \in E(\widehat{G}) \text{ and } f(y) \le f(A_k)\}$ and $d_k := |D_k|$ for $2 \le k \le m$,
$C_k := \{x \in B : xA_k \in E(\widehat{G}) \text{ and } f(A_k) < f(x)\}$ and $c_k := |C_k|$ for $2 \le k \le m$,
$I_k := \{z \in I : f(A_k) < f(z) < f(A_{k+1})\}$ and $i_k := |I_k|$ for $1 \le k \le m-1$,
$I'_k := \{z \in I : f(A_k) < f(z) \le f(A_k) + r\}$ and $i'_k := |I'_k|$ for $1 \le k \le m-1$,
$I''_k := \{z \in I : f(A_{k+1}) - r \le f(z) < f(A_{k+1})\}$ and $i''_k := |I''_k|$ for $1 \le k \le m-1$.

Then $d_1 = c_m = 0$ and $\deg_{\widehat{G}}(A_k) = d_k + c_k$ for $1 \le k \le m$. By (a), the C_i's and D_j's are all disjoint. By the claim, for any $1 \le k \le m$, $I'_k \cup I''_k \subseteq I_k$ (while I'_k and I''_k are not necessarily disjoint). Furthermore, it is clear that for any $1 \le k \le m-1$, $f^{-1}[f(A_k)+1, f(A_k)+r] \subseteq C_k \cup I'_k$, since if $f(A_k) < f(x) \le f(A_k)+r$, then $x \in C_k \cup I'_k$. Similarly, $f^{-1}[f(A_{k+1})-r, f(A_{k+1})-1] \subseteq D_{k+1} \cup I''_k$. Hence we have $c_k + i'_k \ge r$ and $d_{k+1} + i''_k \ge r$, implying that $i_k \ge \max\{i'_k, i''_k\} \ge \max\{(r-c_k)^+, (r-d_{k+1})^+\} = q_k$ for $1 \le k \le m-1$. Therefore,

$$(**) \qquad |I| \ge \sum_{k=1}^{m-1} i_k \ge \sum_{k=1}^{m-1} q_k = q(\Pi).$$

This completes the proof of (b).

Now we have $f^{-1}[f(A_k)+1, f(A_k)+r] \subseteq C_k \cup I'_k \subseteq C_k \cup I_k$ and $f^{-1}[f(A_{k+1}) - r, f(A_{k+1})-1] \subseteq D_{k+1} \cup I''_k \subseteq D_{k+1} \cup I_k$. Because $C_k \cap D_{k+1} = \emptyset$, at least $r - i_k$ colors of $[f(A_{k+1})-r, f(A_{k+1})-1]$ are not in $[f(A_k)+1, f(A_k)+r]$. Thus $f(A_{k+1}) - f(A_k) \ge r + (r-i_k) + 1 = 2r+1-i_k$ for $1 \le k \le m-1$. Summing up, we get (c): $\mathrm{nsp}_r(G) \ge f(A_m) - f(A_1) \ge (m-1)(2r+1) - |I|$.

Now consider the case that $B' \ne \emptyset$; i.e., there exists some $w \in B$ such that $wA_k \notin E(\widehat{G})$ for all $1 \le k \le m$. Hence $|f(w) - f(A_k)| \ge r+1$ for all $1 \le k \le m$. Assume

$f(A_p) < f(w) < f(A_{p+1})$ for some $1 \le p \le m-1$. Then $f(A_{p+1}) - f(A_p) \ge 2r+2$, so $I'_p \cap I''_p = \emptyset$, implying that $i_p \ge i'_p + i''_p \ge (r-c_p)^+ + (r-d_{p+1})^+ = q_p + q'_p$. Replacing $i_p \ge q_p + q'_p$ to the last summation in $(**)$, we get $|I| \ge q(\Pi) + q'_p \ge q(\Pi) + q'(\Pi)$. This proves (d).

Because $f(A_{p+1}) - f(A_p) \ge 2r+2 \ge 2r+1-i_p+q_p+1$, we have, from the first inequality, $\mathrm{nsp}_r(G) \ge f(A_{p+1}) - f(A_p) \ge 2r+2$. Using the second inequality, similar to the proof of (c), one can get $\mathrm{nsp}_r(G) \ge (m-1)(2r+1) - |I| + q_p + 1 \ge (m-1)(2r+1) - |I| + s(\Pi) + 1$. This proves (e). □

In the next result, we complete the solution of $\mathrm{nsp}_r(G)$ for bipartite graphs $G = (A, B, I, E)$ with $|I| = r-2$. Let $s(G) = \min s(\Pi)$, where Π runs over all arrangements of A satisfying Lemma 3.5(b) and (d).

THEOREM 3.6. *Suppose $G = (A, B, I, E)$ is a bipartite graph with $2 \le m = |A| \le |B|$, $0 \le |I| = r-2$, and $\widehat{G}$ has no P_4. Then, $\mathrm{nsp}_r(G) < \infty$ if and only if $\widehat{G}$ satisfies Lemma* 3.5(a), (b), *and* (d). *In this case,*

$$\mathrm{nsp}_r(G) = \begin{cases} (2r+1)(m-1) - r + 2 & \text{if } B' = \emptyset, \\ 2r+2 & \text{if } B' \ne \emptyset \text{ and } m = 2, \\ (2r+1)(m-1) - r + s(G) + 3 & \text{if } B' \ne \emptyset \text{ and } m \ge 3. \end{cases}$$

Proof. The necessity follows from Lemma 3.5. For the sufficiency, suppose $\Pi = (A_1, A_2, \ldots, A_m)$ is an arrangement of A satisfying Lemma 3.5(a), (b), and (d). Moreover, assume $s(\Pi) = s(G)$ when $B' \ne \emptyset$. By Lemma 3.5(a), any two A-vertices have disjoint sets of neighbors in $\widehat{G}$. Then by Lemma 3.5(b), we can label the neighbors of A_k in $\widehat{G}$ by $C_{k,1}, C_{k,2}, \ldots, C_{k,c_k}$ and $D_{k,1}, D_{k,2}, \ldots, D_{k,d_{k+1}}$, respectively, for $1 \le k \le m$. In addition, since $|I| \ge \sum_{k=1}^{m-1} q_k$, there exist distinct I-vertices $I_{k,1}, I_{k,2}, \ldots, I_{k,q_k}$ for all k.

We shall complete the proof by considering the three cases.

Case 1. $B' = \emptyset$. That is, B is the union of all the C-and D-vertices. It suffices to find an N_r-coloring of G with span $(2r+1)(m-1) - r + 2$. (Then we not only prove that $\mathrm{N}_r(G) < \infty$ but also confirm that the span is optimal by Lemma 3.5(c).) We first replace q_{m-1} by $|I| - \sum_{j=1}^{m-2} q_j$. Then $q_{m-1} \ge \max\{(r-c_{m-1})^+, (r-d_m)^+\}$ and $|I| = \sum_{j=1}^{m-1} q_j$. Indeed, letting B represent the C- and D-vertices and I for I-vertices (without indicating the indices), we can line up all vertices of G as an Hamiltonian r-path in G^c as

$$P = A_1 \underbrace{BB\cdots B}_{c_1} \underbrace{II\cdots I}_{q_1} \underbrace{BB\cdots B}_{d_2} A_2 \cdots A_{m-1} \underbrace{BB\cdots B}_{c_{m-1}} \underbrace{II\cdots I}_{q_{m-1}} \underbrace{BB\cdots B}_{d_m} A_m.$$

Note that $d_1 = c_m = 0$. Define a coloring on G by the following three steps. (The idea is to use each I-vertex to reduce the span by 1.)

(1) A-vertices: $f(A_1) = 0$ and $f(A_{k+1}) = f(A_k) + 2r + 1 - q_k$ for $1 \le k \le m-1$.

(2) B-vertices: for all $1 \le k \le m-1$,

$$f(C_{k,j}) = \begin{cases} f(A_k) + j & \text{for } 1 \le j \le r - q_k - 1, \\ f(A_k) + r - q_k & \text{for } r - q_k \le j \le c_k, \end{cases}$$

$$f(D_{k+1,j}) = \begin{cases} f(A_k) + r + j & \text{for } 1 \le j \le r - q_k - 1, \\ f(A_k) + 2r - q_k & \text{for } r - q_k \le j \le d_{k+1}. \end{cases}$$

(3) I-vertices: $f(I_{k,j}) = f(A_k) + r - q_k + j$ for all $q_k > 0$ and $1 \le j \le q_k$.

One can easily verify that f is an N_r-coloring for G with span $(2r+1)(m-1)-|I| = (2r+1)(m-1)-r+2$.

Case 2. $B' \neq \emptyset$ and $m = 2$. Similar to Case 1, by Lemma 3.5(e), it suffices to find an N_r-coloring of G with span $\mathrm{nsp}_r(G) = 2r+2$. Define a coloring by $f(A_1) = 0$, $f(A_2) = 2r+2$, and $f(z) = r+1$ for all vertices z in B'. Since $|I| \geq q(\Pi) + q'(\Pi) = q_1 + q_1' = (r-c_1)^+ + (r-d_2)^+$, there are enough I-vertices to use the colors between 0 and $2r+2$. Thus one can verify that this is an N_r-coloring of G with span $2r+2$.

Case 3. $B' \neq \emptyset$ and $m \geq 3$. Again, by Lemma 3.5(e), it suffices to find an N_r-coloring with span $(2r+1)(m-1)-|I|+s(G)+1$. Suppose $s(\Pi) = q_p$ for some $1 \leq p \leq m-1$ with $q_p' \leq |I| - q(\Pi)$. As before, we replace q_i by $q_i + |I| - q(\Pi) - q_p'$ for some $i \neq p$. Then $|I| = q_1 + q_2 + \cdots + q_{p-1} + (r-c_p)^+ + (r-d_{p+1})^+ + q_{p+1} + \cdots + q_{m-1}$. All the C-, D-, and I-vertices are labeled the same as before, except the I-vertices between A_p and A_{p+1} are labeled as $I'_{p,1}, I'_{p,2}, \ldots, I'_{p,(r-c_p)^+}, I''_{p,1}, I'_{p,2}, \ldots, I'_{p,(r-d_{p+1})^+}$. Apply the same three-step coloring method used for the Case 1, except the colors for the vertices between A_p and A_{p+1} are defined by $f(I'_{p,j}) = f(A_p) + r - (r-c_p)^+ + j$ for $1 \leq j \leq (r-c_p)^+$; $f(w) = f(A_p) + r + 1$ for all $w \in B'$; $f(I''_{p,j}) = f(A_p) + r + 1 + j$ for $1 \leq j \leq (r-d_{p+1})^+$; $f(A_{p+1}) = f(A_p) + 2r + 2$; and

$$f(C_{p,j}) = \begin{cases} f(A_p) + j & \text{for } 1 \leq j \leq r - (r-c_p)^+ - 1, \\ f(A_p) + r - (r-c_p)^+ & \text{for } r - (r-c_p)^+ \leq j \leq c_p, \end{cases}$$

$$f(D_{k,j}) = \begin{cases} f(A_p) + r + 1 + (r-d_{p+1})^+ + j & \text{for } 1 \leq j \leq r - (r-d_{p+1})^+ - 1, \\ f(A_p) + 2r + 1 & \text{for } r - (r-d_{p+1})^+ \leq j \leq d_{p+1}. \end{cases}$$

This gives an N_r-coloring for G with span $(2r+1)(m-1)-|I|+s(G)+1 = (2r+1)(m-1)-r+s(G)+3$. □

Based on Lemma 3.5, using a similar process in the proof of Theorem 3.6, we can also completely settle the case that $I = \emptyset$ and $r \geq 2$. In this case, Lemma 3.5(b) means that $q_k = 0$ for all k, or, equivalently, that $\widehat{G}$ has two A-vertices of degree at least r and the rest $(m-2)$ A-vertices of degree at least $2r$. Furthermore, Lemma 3.5(d) holds automatically, and $s(\Pi) = 0$. This implies that the lower bound in Lemma 3.5(e) is simply $(m-1)(2r+1)+1$. Hence the same labeling procedure used in Theorem 3.6 gives the following result.

THEOREM 3.7. *Let $G = (A, B, I, E)$ be a bipartite graph with $2 \leq m = |A| \leq |B|$, $I = \emptyset$, and $\widehat{G}$ contains no P_4. If $r \geq 2$, then $\mathrm{nsp}_r(G) < \infty$ if and only if Lemma* 3.5(a) *holds and $\widehat{G}$ has two A-vertices of degree at least r and the other $(m-2)$ A-vertices of degree at least $2r$. In this case,*

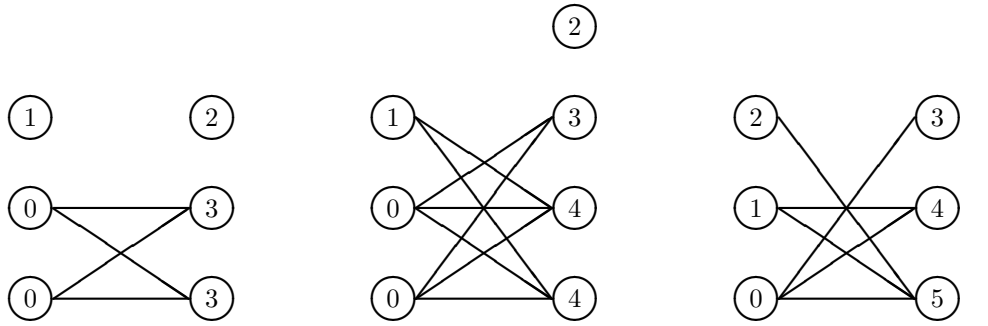
$$\mathrm{nsp}_r(G) = \begin{cases} (2r+1)(m-1) & \text{if } B' = \emptyset, \\ (2r+1)(m-1)+1 & \text{if } B' \neq \emptyset. \end{cases}$$

By Corollary 3.4 and Theorems 3.3 and 3.7, we obtain the complete solutions of $\mathrm{nsp}_2(G)$ for bipartite graphs.

THEOREM 3.8. *If $G = (A, B, I, E)$ is a bipartite graph with at least one edge and $1 \le m = |A| \le |B|$, then*

$$\mathrm{nsp}_2(G) = \begin{cases} 3 & \text{if } |I| \ge 2; \\ 4 & \text{if } |I| = 1 \text{ and } E(\widehat{G}) \ne \emptyset; \\ 5 & \text{if } |I| = 0 \text{ and } \widehat{G} \text{ has a } P_4; \\ 5m-5 & \text{if } |I| = 0,\ B' = \emptyset, \text{ and } \widehat{G} \text{ is a disjoint union of } m \\ & \text{stars, centered at } A \text{ except that two of the stars have} \\ & \text{at least 2 edges, each star has at least 4 edges;} \\ 5m-4 & \text{same as the above, except } B' \ne \emptyset; \\ \infty & \text{other than any of the above.} \end{cases}$$

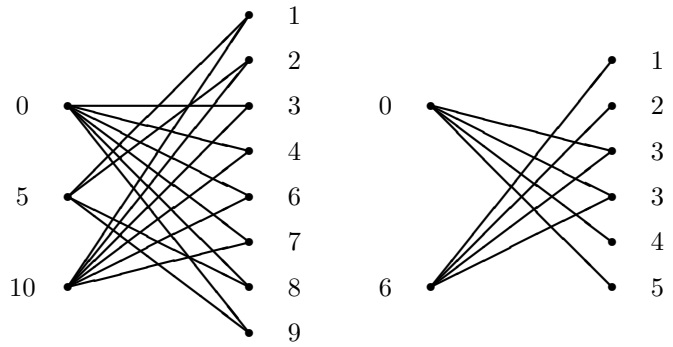
Figure 3.3 shows examples of Theorem 3.8.



G_1: example for Case 1 — $\mathrm{nsp}_2(G_1) = 3$

G_2: example for Case 2 — $\mathrm{nsp}_2(G_2) = 4$

G_3: example for Case 3 — $\mathrm{nsp}_2(G_3) = 5$



G_4: example for Case 4 — $\mathrm{nsp}_2(G_4) = 10$

G_5: example for Case 5 — $\mathrm{nsp}_2(G_5) = 6$

FIG. 3.3. *Five examples for Theorem* 3.8.

Remark. This article is aimed at computing the values of $\mathrm{nsp}_T(G)$ for bipartite graphs when $T = \{0, 1, \ldots, r\}$. Another article by Chang, Juan, and Liu [1] deals with the values of $\mathrm{nsp}_T(G)$ for unit-interval graphs when $T = \{0, 1\}$. The no-hole T-colorings for some other T-sets and different families of graphs were studied by Liu and Yeh [13]. It was proved [13] that if T is r-initial or $T = [a, b]$, $1 \le a \le b$, then for any large n, there exists some graph on n vertices such that $\mathrm{nsp}_T(G)$ equals the upper bound $n - 1$.

Acknowledgment. The authors are grateful to the two anonymous referees for valuable comments.

REFERENCES

[1] G. J. Chang, S. Juan, and D. Liu, *No-hole 2-distant colorings for unit interval graphs*, Ars Combinatoria, to appear.
[2] G. J. Chang, D. D.-F. Liu, and X. Zhu, *Distance graphs and T-coloring*, J. Combin. Theory Ser. B, 75 (1999), pp. 259–269.
[3] M. B. Cozzens and F. S. Roberts, *T-colorings of graphs and the channel assignment problem*, Congr. Numer., 35 (1982), pp. 191–208.
[4] M. B. Cozzens and F. S. Roberts, *Greedy algorithms for T-colorings of complete graphs and the meaningfulness of conclusions about them*, J. Combin. Inform. System Sci., 16 (1991), pp. 286–299.
[5] J. R. Griggs and D. D.-F. Liu, *The channel assignment problem for mutually adjacent sites*, J. Combin. Theory Ser. A, 68 (1994), pp. 169–183.
[6] W. K. Hale, *Frequency assignment: Theory and applications*, Proc. IEEE, 68 (1980), pp. 1497–1514.
[7] S. J. Hu, S. T. Juan, and G. J. Chang, *T-Colorings and T-edge spans of graphs*, Graphs Combin., 15 (1999), pp. 295–301.
[8] S. T. Juan, *The No-Hole T-Coloring Problem*, Master Thesis, Department of Applied Math., National Chiao Tung University, Hsinchu, Taiwan, 1996.
[9] T. A. Lanfear, *Radio Frequency Assignment and Graph Coloring*, presented at the Third Advanced Research Institute in Discrete Applied Mathematics, RUTCOR, Rutgers University, New Brunswick, NJ, 1988.
[10] D. D.-F. Liu, *T-colorings of graphs*, Discrete Math., 101 (1992), pp. 203–212.
[11] D. D.-F. Liu, *On a conjecture of T-colorings*, Congr. Numer., 103 (1994), pp. 27–31.
[12] D. D.-F. Liu, *T-graphs and the channel assignment problem*, Discrete Math., 161 (1996), pp. 197–205.
[13] D. D.-F. Liu and R. Yeh, *Graph homomorphism and no-hole T-coloring*, Congr. Numer., 138 (1999), pp. 39–48.
[14] J. H. Rabinowitz and V. K. Proulx, *An asymptotic approach to the channel assignment problem*, SIAM J. Algebraic Discrete Methods, 6 (1985), pp. 507–518.
[15] A. Raychaudhuri, *Further results on T-coloring and frequency assignment problems*, SIAM J. Discrete Math., 7 (1994), pp. 605–613.
[16] F. S. Roberts, *No-hole 2-distant colorings*, Math. Comput. Modelling, 17 (1993), pp. 139–144.
[17] D. Sakai and C. Wang, *No-hole $(r+1)$-distant colorings*, Discrete Math., 119 (1993), pp. 175–189.
[18] B. A. Tesman, *T-Colorings, List T-Colorings and Set T-Colorings of Graphs*, Ph.D. Dissertation, Department of Math., Rutgers University, New Brunswick, NJ, 1989.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 381–407

TRANSFORMATIONS ON REGULAR NONDOMINATED COTERIES AND THEIR APPLICATIONS*

KAZUHISA MAKINO† AND TIKO KAMEDA‡

Abstract. A coterie under an underlying set U is a family of subsets of U such that every pair of subsets has at least one element in common, but neither is a subset of the other. A coterie C under U is said to be *nondominated* (ND) if there is no other coterie D under U such that, for every $Q \in C$, there exists $Q' \in D$ satisfying $Q' \subseteq Q$.

We introduce the operation σ which transforms a ND coterie to another ND coterie. A *regular* coterie is a natural generalization of a *vote-assignable* coterie. We show that any regular ND coterie C can be transformed to any other regular ND coterie D by judiciously applying the σ operation to C at most $|C| + |D| - 2$ times.

As another application of the σ operation, we present an incrementally polynomial-time algorithm for generating all regular ND coteries. We then introduce the concept of a g-*regular* functional as a generalization of availability. We show how to construct an optimum coterie C with respect to a g-regular functional in $O(n^3|C|)$ time, where $n = |U|$. Finally, we discuss the structures of optimum coteries with respect to a g-regular functional.

Key words. coterie, nondominatedness, regular coterie, availability, mutual exclusion, positive self-dual Boolean function, regular self-dual Boolean function, g-regular functional

AMS subject classifications. 68M14, 68M15, 68P15, 68Q25, 68R05

PII. S0895480100371110

1. Introduction.

A *coterie* C under an underlying set $U = \{1, 2, \ldots, n\}$ is a family of subsets (called *quorums*) of U satisfying the *intersection property* (i.e., for any pair $S, R \in C$, $S \cap R \neq \emptyset$ holds) and *minimality* (i.e., no quorum in C contains any other quorum in C) [18, 23]. The concept of a coterie has applications in diverse areas, such as mutual exclusion in distributed systems [13, 18, 23], data replication protocols [14], name servers [27], selective dissemination of information [39], and distributed access control and signatures [30].

For example, to achieve mutual exclusion in a distributed system, let the elements in U represent the sites in the distributed system. A process is allowed to enter a critical section only if it can get permissions from all the members of a quorum $Q \in C$, where each site is allowed to issue at most one permission at a time. By the intersection property, it is guaranteed that at most one process can enter the critical section at any time.

A coterie C under U is said to *dominate* another coterie D ($\neq C$) under U if, for each quorum $Q \in D$, there is a quorum $Q' \in C$ satisfying $Q' \subseteq Q$. A coterie which is not dominated by any other coterie is said to be *nondominated* (ND) [18]. ND coteries are important in practical applications, since they have maximal "efficiency" in some sense [4, 18, 21].

*Received by the editors April 17, 2000; accepted for publication (in revised form) June 6, 2001; published electronically August 29, 2001. An extended abstract of this paper appears in *Proceedings of the Nineteenth ACM Symposium on Principles of Distributed Computing* (*PODC* 2000), Portland, OR, 2000, pp. 279–288. This work was supported in part by the Scientific Grant in Aid, by the Ministry of Education, Science, Sports, and Culture of Japan, and in part by the Natural Sciences and Engineering Research Council of Canada.

http://www.siam.org/journals/sidma/14-3/37111.html

†Division of Systems Science, Graduate School of Engineering Science, Osaka University, Toyonaka, Osaka, 560-8531, Japan (makino@sys.es.osaka-u.ac.jp)

‡School of Computing Science, Faculty of Applied Sciences, Simon Fraser University, Burnaby, British Columbia, V5A 1S6 Canada (tiko@cs.sfu.ca)

Given a family C of subsets of U, which is not necessarily a coterie, we define a *positive* (i.e., monotone) Boolean function f_C such that $f_C(x) = 1$ if the Boolean vector $x \in \{0,1\}^n$ is greater than or equal to the characteristic vector of some subset[1] in C, and 0 otherwise, where $n = |U|$. It was shown in [20] that C is a coterie if and only if f_C is *dual-minor*, and C is ND if and only if f_C is *self-dual*. (See section 2.2.) Based on this characterization, the methods developed in the rich field of Boolean functions can be exploited to derive various properties of coteries and ND coteries.

A coterie C is said to be *vote-assignable* if there exist a *vote assignment* $w : U \mapsto \mathbb{R}^+$ and a *threshold* $t \in \mathbb{R}^+$ such that $w(S) \geq t$ if and only if $S \supseteq Q$ for some $Q \in C$ [18, 19, 37], where $\mathbb{R}^+$ is the set of nonnegative real numbers and $w(S) = \sum_{i \in S} w(i)$. It is easy to see that there is a one-to-one correspondence between vote-assignable coteries (resp., ND coteries) C and dual-minor (resp., self-dual) threshold Boolean functions f_C. (For the definition of a threshold Boolean function, see section 2.) The vote-assignable coteries are important and have been used in many practical problems, since they can be handled efficiently (see, e.g., [18, 19, 37, 38]). We assume in this paper that a vote assignment w satisfies $w(i) \geq w(j)$ for all $i < j$, since we are interested in coteries which are nonequivalent under permutation on U. A coterie C is *equivalent to* a coterie C' *under permutation* if C can be transformed into C' by permuting the elements of U. For example, $C = \{\{1,2\},\{1,3\}\}$ is equivalent to $C' = \{\{2,3\},\{2,1\}\}$ under permutation. A coterie C is said to be *regular* if, for each $Q \in C$ and every pair $(i,j) \in U \times U$ with $i < j$, $i \notin Q$ and $j \in Q$, there exists $Q' \in C$ such that $Q' \subseteq (Q \setminus \{j\}) \cup \{i\}$.[2] By definition (and the discussion in section 2), a vote-assignable coterie C is always regular, though in general the converse is not true. The regular Boolean functions were defined as a generalization of the threshold functions [28]. It is known that most regular coteries are vote-assignable [28]; in particular, all regular ND coteries under U with $n = |U| \leq 9$ are vote-assignable.

Among the important problems regarding coteries are the following:

(i) decide whether a given coterie is ND (equivalently, whether a given positive dual-minor function is self-dual);

(ii) construct "optimal" ND coteries according to a certain criterion, such as availability and load [29] (equivalently, construct "optimal" positive self-dual functions); and

(iii) generate all ND coteries (equivalently, all positive self-dual functions) systematically.

Unfortunately, the complexity of problem (i) is still unknown [8, 16, 22], although a result by Fredman and Khachiyan [17] suggests that it is unlikely that the problem is NP-hard. [8, 16] give a number of interesting equivalent problems which arise in various fields of applications. However, it is known that if we restrict ourselves to regular coteries, (i) is polynomially solvable [6, 32].

Although (i) is an interesting problem, we do not consider (i) further in this paper. Instead, we focus on problems (ii) and (iii). As for (ii), let us consider the availability of a coterie, where the concept of availability has been extensively studied under different names in reliability theory (see, e.g., [35]). Assume that each element can be in either of two states, *operational* or *inoperational*, and takes on its state randomly and independently, element i being operational (resp., inoperational) with probability p_i (resp., $1 - p_i$). Given operational probabilities p_i, $i \in U$, where we assume without loss of generality that $1 \geq p_1 \geq p_2 \geq \cdots \geq p_n \geq 0$, the *availability* of a coterie C is

[1] The ith component of the characteristic vector is 1 (0) if $i \in U$ is (not) contained in the subset.

[2] This definition was motivated by the definition of *regular* Boolean functions. See section 2.3.

the probability that the set of operational elements contains at least one quorum in C. Availability is undoubtedly an important concept in practical applications, and hence it is natural to construct a coterie with the maximum availability.

The availability of coteries has been studied extensively [1, 5, 15, 33, 36, 38]. It is known [1, 36] that the elements $i \in U$ with $p_i < 1/2$ can be ignored; i.e., there exists a maximum-availability coterie C such that no quorum in C contains i. (In the case where $p_i < 1/2$ holds for all i, $C = \{\{1\}\}$ has the maximum availability [1, 15, 33].) Thus, we shall assume that

$$(1 \geq)\ p_1 \ \geq\ p_2 \ \geq\ \cdots \ \geq\ p_n \ \geq\ 1/2.$$

It is also known that if either $p_1 = 1$ or $p_1 \leq 1/2$, then $C = \{\{1\}\}$ has the maximum availability. If $1 \neq p_1 > 1/2$, on the other hand, it is demonstrated in [36, 38] that the coterie C_{max}, given below, maximizes availability. First define the weight for $i \in U$ by

$$w^*(i) = \log_2(p_i/(1-p_i)) \tag{1}$$

and introduce the notation $w^*(S) = \sum_{i \in S} w^*(i)$ for $S \subseteq U$. Now, $Q \in C_{max}$ if

(a) $w^*(Q)$ $(= w^*(U \setminus Q)) = w^*(U)/2$ and $1 \in Q$ (1 is an element of U), or

(b) Q is a minimal subset of U satisfying $w^*(Q) > w^*(U)/2$, and Q does not contain any quorum of type (a).

Since this coterie C_{max} is vote-assignable, Amir and Wool [1], Spasojevic and Berman [36], and Tong and Kain [38] proposed algorithms to compute a vote assignment w from w^*, called *tie-breaking* algorithm, in order to remove case (a). An exponential algorithm is proposed in [38] to find the "optimal" tie-breaking rule, while Amir and Wool [1] and Spasojevic and Berman [36] present polynomial-time approximation algorithms for it. The main problem with the above definition of C_{max} is that there may exist a subset $S \subseteq U$ such that $w^*(S) = w^*(U \setminus S)$ (case (a)), because of which a simple vote assignment w (showing that C_{max} is vote-assignable) is not easily obtainable, and that the weight $w^*(i)$ is, in general, not a rational number; hence we cannot compute $w^*(S) = \sum_{i \in S} w^*(i)$ in polynomial time. For the above reasons, no polynomial algorithm for constructing maximum-availability coteries was known. In this paper, we present a polynomial-time algorithm for it. More precisely, we define a "g-regular" functional as a generalization of availability (see section 6) and then show that, given a g-regular functional Φ, we can compute a coterie C which maximizes Φ in $O(n^3|C|)$ time, where $|C|$ is the number of quorums in C.

Problem (iii) is known to be useful to solve (ii) [9, 18]. To solve (ii), we first enumerate all (or some) ND coteries efficiently and select the best one under a certain criterion, which is not easily computable. This procedure is useful when n is small or when we have enough time to compute it. We feel that (iii) is mathematically interesting, giving us an insight into the structure of ND coteries (or, equivalently, self-dual Boolean functions).

The generation of all ND coteries in a certain subclass of vote-assignable ND coteries was discussed in [28], which is used to give a lower bound on the number of all vote-assignable ND coteries. However, the procedure is not polynomial and computes a proper subclass of vote-assignable ND coteries. Garcia-Molina and Barbara [18] proposed an algorithm to generate all ND coteries in a certain superclass of regular ND coteries. However, it is also not polynomial. Bioch and Ibaraki [9] later came up with a polynomial-time algorithm to generate all ND coteries, and compiled a list containing all ND coteries of up to seven elements, which are essentially different

(i.e., nonequivalent under permutation). We remark here that their algorithm is not polynomial if equivalent duplicates are to be deleted from the output. In fact, they compiled a list of all ND coteries under seven or fewer elements by first running their algorithm and then selecting nonequivalent representatives from among them. In this paper, we present a polynomial algorithm to generate all *regular* ND coteries. Since no regular ND coterie C is equivalent to any other regular ND coterie C' $(\neq C)$ under permutation (see Lemma 2.2), our algorithm does not output ND coteries which are equivalent under permutation. Although our algorithm outputs only regular ND coteries, it is practically useful because all ND coteries under $n = 5$ or fewer elements are all regular (if we consider their representatives), and when n is relatively small, a large fraction of ND coteries are regular [28]. Moreover, if the objective function of (ii) cited above is g-regular (e.g., the availability of a coterie), then we can restrict our attention to regular coteries.

After defining necessary terminology in section 2 (we use Boolean terminology, which is simpler than that of set theory), we discuss in section 3 two operations, named ρ and σ, which transform the positive self-dual function f (representing a ND coterie) into another positive self-dual function (representing another ND coterie) by making a minimal change in the set of minimal true vectors of f. The ρ operation was introduced in [9], and σ was implicitly introduced in [18], where it is called *coterie transformation*.

Section 4 shows that any regular self-dual function f (representing a regular ND coterie) can be transformed into any other regular self-dual function g (representing any other regular ND coterie) by judiciously applying the σ operation to f at most $|\min T(f)| + |\min T(g)| - 2$ times. (For the definition of $\min T(f)$, see section 2.) In sections 5 and 6, we consider the problems of generating all regular self-dual functions and of computing an optimal self-dual function with respect to a g-regular functional Φ (see the definition of g-regularity in section 6) as applications of the above transformation.

In addition to the theory of coteries, the concepts of self-duality and regularity play important roles in diverse areas such as computational learning theory (e.g., identification of positive Boolean functions [8, 10, 24, 25]), threshold logic [28], operations research [6, 11, 31, 32], clutters in set theory [7], minimal transversals in hypergraphs [16], and coherent systems of reliability theory [35]. The results of this paper are relevant to all these problems.

2. Definitions and basic properties. A *Boolean function*, or a *function* in short, of n variables is a mapping $f : \{0,1\}^n \to \{0,1\}$, where $v \in \{0,1\}^n$ is called a *Boolean vector* (a *vector* in short). If $f(v) = 1$ (resp., 0), then v is called a *true* (resp., *false*) vector of f. The set of all true vectors (resp., false vectors) of f is denoted by $T(f)$ (resp., $F(f)$). Throughout this paper, the constant functions with $T(f) = \emptyset$ (empty) and $F(g) = \emptyset$ are denoted by $f = \perp$ and $g = \top$, respectively. For any two functions f and g, we write $f \leq g$ if $T(f) \subseteq T(g)$. For a vector $v = (v_1, v_2, \ldots, v_n)$, we define $ON(v) = \{j \mid v_j = 1\}$ and $OFF(v) = \{j \mid v_j = 0\}$.

The argument x of function f is represented as a vector $x = (x_1, x_2, \ldots, x_n)$, where each x_i is a Boolean *variable*. A variable x_i is said to be *relevant* if there exist two vectors v and w such that $f(v) \neq f(w)$, $v_i \neq w_i$, and $v_j = w_j$ for all $j \neq i$; otherwise, it is said to be *irrelevant*. The set of all relevant variables of a function f is denoted by $V_f \subseteq V = \{x_1, x_2, \ldots, x_n\}$. A *literal* is either a variable x_i or its complement $\overline{x}_i$, which are referred to as a *positive* or *negative* literal, respectively. The *complement* of vector $x = (x_1, x_2, \ldots, x_n)$ is defined by $\overline{x} = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)$. A

term t is a conjunction $(\bigwedge_{i\in P(t)} x_i) \wedge (\bigwedge_{j\in N(t)} \overline{x}_j)$ of literals such that $P(t), N(t) \subseteq \{1, 2, \ldots, n\}$ and $P(t) \cap N(t) = \emptyset$. For example, $t_1 = x_1x_4\overline{x}_5x_6$ is a term, while $t_2 = x_2x_4\overline{x}_2$ is not. In particular, the term t with $P(t) = N(t) = \emptyset$ represents $\top$. A *disjunctive normal form* (DNF) is a disjunction of distinct terms. It is easy to see that any function f can be represented in DNF whose variable set is V_f. We sometimes do not distinguish a formula (e.g., DNF) from the function it represents if no confusion arises.

2.1. Positive functions. For a pair of vectors $v, w \in \{0,1\}^n$, we write $v \leq w$ if $v_j \leq w_j$ holds for all $j \in V$, and $v < w$ if $v \leq w$ and $v \neq w$, where we define $0 < 1$. For a set of vectors $S \subseteq \{0,1\}^n$, $\min_{\geq} S$ (resp., $\max_{\geq} S$) denotes the set of all minimal (resp., maximal) vectors in S with respect to $\geq$. For example, for a function f, $\min_{\geq} T(f)$ denotes the set of all minimal true vectors of f, and $\max_{\geq} F(f)$ denotes the set of all maximal false vectors of f. We sometimes use $\min S$ (resp., $\max S$) instead of $\min_{\geq} S$ (resp., $\max_{\geq} S$) if no confusion arises. A function f is said to be *positive* or *monotone* if $v \leq w$ always implies $f(v) \leq f(w)$. A *prime implicant* of a function f is a term (i.e., monomial) t such that $t \leq f$, but $t' \not\leq f$ for any proper subterm t' of t. There is a one-to-one correspondence between $\min T(f)$ and the set of all prime implicants of f such that a vector v corresponds to the term t_v defined by $t_v = x_{i_1}x_{i_2}\cdots x_{i_k}$ if $v_{i_j} = 1, j = 1, 2, \ldots, k$, and $v_i = 0$ otherwise. For example, the vector $v = (1010)$ corresponds to the term $t_v = x_1x_3$. In particular, if $v = (00\cdots0)$, then $t_v = \top$. Note that $t_v \leq t_w$ (as functions) holds if and only if $v \geq w$. We also use the notation $t_{\overline{v}}$ to denote the term $x_{j_1}x_{j_2}\cdots x_{j_l}$, where $\{j_1, j_2, \ldots, j_l\} = \{1, 2, \ldots, n\} \setminus \{i_1, i_2, \ldots, i_k\}$. For the above $v = (1010)$, we have $t_{\overline{v}} = x_2x_4$.

It is known that a positive function f is uniquely determined by $\min T(f)$ (hence a positive function f can be represented by a string of length $\leq n|\min T(f)|$) and that f has the unique *minimal disjunctive normal form* (MDNF), consisting of all the prime implicants of f, where $N(t) = \emptyset$ for each prime implicant t. In this paper, we sometimes represent the MDNF of a positive function such as $f = x_1x_2 + x_2x_3 + x_3x_1$ in a simplified form $f = 12 + 23 + 31$, using only the subscripts of the literals. The set of minimal true vectors of this function is $\min T(f) = \{(110), (011), (101)\}$ if f is a 3-variable function. Coteries can be conveniently modeled by Boolean functions based on the fact that $\min T(f)$ can represent a family of subsets, none of which includes the other. For example, the above $\min T(f)$ represents a coterie $C = \{\{1, 2\}, \{2, 3\}, \{3, 1\}\}$, while $T(f)$ represents the family of all subsets that contain a member of C.

2.2. Dual-comparable functions. The *dual* of a function f, denoted f^d, is defined by

$$f^d(x) = \overline{f}(\overline{x}),$$

where $\overline{f}$ and $\overline{x}$ denote the complement of f and x, respectively. As is well known, f^d is obtained from f by interchanging $+$ (OR) and $\cdot$ (AND), as well as the constants 0 and 1. Recall that for any two functions f and g, we write $f \leq g$ if $T(f) \subseteq T(g)$, and $f < g$ if $f \leq g$ and $f \neq g$. We say that f *is covered by* g if $f \leq g$. It is easy to see that $(f + g)^d = f^dg^d$, $(fg)^d = f^d + g^d$, $f \leq g$ if and only if $f^d \geq g^d$, and so on. A function is called *dual-minor* if $f \leq f^d$, *dual-major* if $f \geq f^d$, and *self-dual* if $f = f^d$. It is known [20] that

1. f is dual-minor if and only if at most one of v and $\overline{v}$ belongs to $T(f)$ for any $v \in \{0,1\}^n$;

2. f is dual-major if and only if at least one of v and $\overline{v}$ belongs to $T(f)$ for any $v \in \{0,1\}^n$; and
3. f is self-dual if and only if exactly one of v and $\overline{v}$ belongs to $T(f)$ for any $v \in \{0,1\}^n$.

For example, $f = 123$ is dual-minor since $f^d = 1 + 2 + 3$ satisfies $f \leq f^d$. The dual of $f = 12 + 23 + 31$ is

$$f^d = (1+2)(2+3)(3+1) \ = \ 12 + 23 + 31.$$

This function f is self-dual and is called the *basic majority function*; it is known to be the only positive self-dual function of three relevant variables. There is no positive self-dual function of exactly two relevant variables. However, each function $f = x_i$ is a positive self-dual function of one variable.

If f is positive, then f^d is also positive. In this case, an alternative definition of f^d is given by the condition that $v \in T(f^d)$ if and only if v is a *transversal* of $\min T(f)$; i.e., it satisfies $ON(v) \cap ON(w) \neq \emptyset$ for all $w \in \min T(f)$. Let

$\mathcal{C}_{SD}(n)$: the class of all positive self-dual functions of n variables,
$\mathcal{C}_{DMA}(n)$: the class of all positive dual-major functions of n variables,
$\mathcal{C}_{DMI}(n)$: the class of all positive dual-minor functions of n variables.

Note that in these definitions functions may have some irrelevant variables.

2.3. Regular, 2-monotonic, and threshold functions. A positive function f is said to be *regular* if, for every $v \in \{0,1\}^n$ and every pair (i,j) with $i < j$, $v_i = 0$ and $v_j = 1$, the following condition holds:

$$f(v) \leq f(v + e^{(i)} - e^{(j)}), \tag{2}$$

where $e^{(k)}$ denotes the unit vector which has a 1 in its kth position and 0 in all other positions.

In order to define an important partial order on $\{0,1\}^n$, we first define the concept of the *profile* of a vector $v \in \{0,1\}^n$ as follows:

$$prof_v(k) = \sum_{j \leq k} v_j,$$

where $k = 1, 2, \ldots, n$. If $v, w \in \{0,1\}^n$, where $v \neq w$, satisfy $prof_v(k) \leq prof_w(k)$ for all k, then we write $v \prec w$ (or $w \succ v$), and we say that w *majorizes* v. If $v \prec w$ or $v = w$, then we write $v \preceq w$ (or $w \succeq v$). It is helpful to visualize the profile as in Figure 2.1.

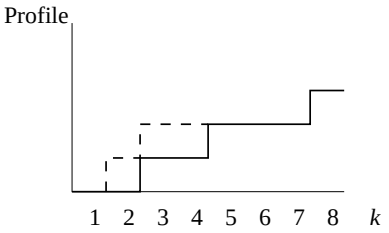


FIG. 2.1. *The profiles $prof_v(k)$ (solid lines) and $prof_w(k)$ (dashed lines).*

In Figure 2.1 the profiles of $v = (00101001)$ and $w = (01100001)$ are depicted in the solid staircase and dashed staircase, respectively. The dashed staircase is not visible where it overlaps with the solid staircase. It is easy to see that if v is majorized by w, then the profile of w does not go below the profile of v.

It is clear from the above definition that $v \prec w$ if and only if $\overline{v} \succ \overline{w}$, since $prof_{\overline{v}}(k) = k - prof_v(k)$. Note that $v \leq w$ implies $v \preceq w$, but the converse is not always true. A function f is said to be *profile-monotone* if $v \prec w$ implies $f(v) \leq f(w)$. The following lemma is proved in [28].

LEMMA 2.1 (see Muroga [28]). *A function f is regular if and only if f is* profile-monotone.

For two functions f and g, we say that f is *equivalent to g under permutation* if permuting variables of f produces g.

LEMMA 2.2. *Two different regular functions are not equivalent under permutation.*

Proof. Let f and g be regular functions such that g can be obtained from f by a permutation π, where we regard π as the permutation on indices; i.e., we write $\pi(i) = j$ instead of $\pi(x_i) = x_j$. We claim that $f = g$, which proves the lemma. Let i and j be indices satisfying $i < j$ and $\pi(i) > \pi(j)$. Note that if there exist no such indices, then π is the identity permutation, implying $f = g$. By the regularity of f, we have

$$f(v) \leq f(v + e^{(i)} - e^{(j)}) \tag{3}$$

for every $v \in \{0,1\}^n$ with $v_i = 0$ and $v_j = 1$, and by the regularity of g, we have $g(w) \leq g(w + e^{\pi(j)} - e^{\pi(i)})$ for every $w \in \{0,1\}^n$ with $w_{\pi(j)} = 0$ and $w_{\pi(i)} = 1$, that is,

$$f(v) \geq f(v + e^{(i)} - e^{(j)}) \tag{4}$$

for every $v \in \{0,1\}^n$ with $v_i = 0$ and $v_j = 1$. By combining (3) and (4),

$$f(v) = f(v + e^{(i)} - e^{(j)})$$

holds for every $v \in \{0,1\}^n$ with $v_i = 0$ and $v_j = 1$. This means that f is symmetric in variables x_i and x_j. Namely, the function f' obtained from f by the permutation

$$\pi'(k) = \begin{cases} i & \text{if } k = j, \\ j & \text{if } k = i, \\ k & \text{otherwise} \end{cases}$$

is identical to f (i.e., $f' = f$). Since π can be obtained by a concatenation of such permutations π', it follows by induction that $f = g$. □

For a set of vectors $S \subseteq \{0,1\}^n$, $\min_{\succeq} S$ (resp., $\max_{\succeq} S$) denotes the set of all minimal (resp., maximal) vectors in S with respect to $\succeq$. For any set of vectors $S \subseteq \{0,1\}^n$, we have $\min_{\succeq} S \subseteq \min S\,(= \min_{\geq} S)$ and $\max_{\succeq} S \subseteq \max S\,(= \max_{\geq} S)$, since $v \geq w$ implies $v \succeq w$. In particular, we have $\min_{\succeq} T(f) \subseteq \min T(f)$; i.e., any element of $\min T(f) \setminus \min_{\succeq} T(f)$ majorizes an element of $\min_{\succeq} T(f)$. It follows from Lemma 2.1 that a regular function f is uniquely determined by $\min_{\succeq} T(f)$.

A positive function f is called 2-*monotonic* if there exists a linear ordering on V for which f is regular. The 2-monotonicity and related concepts have been studied in various contexts in fields such as threshold logic [6, 12, 28, 32], game theory

[35], hypergraph theory [11], and learning theory [10, 24, 25]. The 2-monotonicity was originally introduced in conjunction with threshold functions (e.g., [28]), where a positive function f is a *threshold* function if there exist n nonnegative real numbers (weights) $w_1, w_2, \ldots, w_n$ and a nonnegative real number (threshold) t such that

$$f(x) = \begin{cases} 1 & \text{if } \sum w_i x_i \geq t, \\ 0 & \text{if } \sum w_i x_i < t. \end{cases}$$

As this f satisfies (2) by permuting variables so that $w_i > w_j$ implies $i < j$, a threshold function is always 2-monotonic, although the converse is not true [28].

It is known [18] that there are $\Omega(2^{2^{cn}})$ self-dual functions, where c denotes some positive constant, but only $O(2^{n^2})$ threshold self-dual functions. It is not known if 2-monotonic self-dual functions are substantially more than threshold self-dual functions.

3. The operations ρ and σ. Let f be a positive function of n variables. Throughout this paper, we assume that f is *nontrivial* in the sense that $f \neq \bot, \top$ and $n \geq 1$. Given a vector $v \in \min T(f)$, the operation ρ_v applied to f removes v from $T(f)$ and then adds $\overline{v}$ to $T(f)$ [9]. More precisely, while adding $\overline{v}$, all the vectors larger than $\overline{v}$ are also added to $T(f)$. Therefore,

$$T(\rho_v(f)) = (T(f) \setminus \{v\}) \cup T_{\geq}(\overline{v}), \tag{5}$$

where

$$T_{\geq}(\overline{v}) = \{w \in \{0,1\}^n \mid w \geq \overline{v}\}.$$

An equivalent definition is

$$\rho_v(f) = f_{\setminus v} + t_{\overline{v}} + t_v t_{\overline{v}}^d, \tag{6}$$

where $f_{\setminus v}$ denotes the function defined by all the prime implicants of f except t_v, and $t_{\overline{v}}^d$ denotes the dual of $t_{\overline{v}}$. We note that if $t_v = x_{i_1} x_{i_2} \cdots x_{i_k}$ and $t_{\overline{v}} = x_{j_1} x_{j_2} \cdots x_{j_l}$, then

$$t_v t_{\overline{v}}^d = x_{i_1} x_{i_2} \cdots x_{i_k} (x_{j_1} + x_{j_2} + \cdots + x_{j_l})$$

represents all the vectors larger than v. As seen in Example 3.1, the expression (6) is not necessarily in MDNF, even if $f_{\setminus v}$ is represented by its MDNF, because some of the prime implicants in $t_{\overline{v}} + t_v t_{\overline{v}}^d$ may cover or may be covered by some prime implicants in $f_{\setminus v}$.

Let us note that the operation ρ is essentially the same as the coterie transformation (CT) in [18] except that CT assumes the following additional conditions: (i) $|OFF(v)| \geq 2$, and (ii) there is at least one prime implicant in $t_v t_{\overline{v}}^d$ which is not covered by $f_{\setminus v}$. In this sense, CT is a special case of the ρ operation.

Given a vector $v \in \min T(f)$ and a variable set I with $V_f \subseteq I \subseteq V$, we define the operation $\sigma_{(v;I)}$ by

$$\sigma_{(v;I)}(f) = f_{\setminus v} + t_{\overline{v[I]}} + t_{v[I]} t_{\overline{v[I]}}^d, \tag{7}$$

where $v[I]$ denote the *projection* of v on I; e.g., if $v = (1100)$, $I_1 = \{x_1, x_2, x_3\}$, and $I_2 = \{x_2, x_3\}$, then $v[I_1] = (110)$ and $v[I_2] = (10)$. By definition, we have $\sigma_{(v;V)} = \rho_v$. This operation $\sigma_{(v;I)}$ is implicitly used in [18].

Example 3.1. Consider a positive function of $n = 7$ variables,

$$f = 12 + 13 + 145 + 234 + 235.$$

For this function, we have $V_f = \{1, 2, 3, 4, 5\}$.[3] For $v = (1100000)$ and $w = (0111000)$, we show below how operations ρ and σ are applied.

$$\begin{aligned}
\rho_v(f) &= 13 + 145 + 234 + 235 + \mathbf{34567} + 12(\mathbf{3} + \mathbf{4} + \mathbf{5} + \mathbf{6} + \mathbf{7}) \\
&= 124 + 125 + 126 + 127 + 13 + 145 + 234 + 235 + 34567, \\
\rho_w(f) &= 12 + 13 + 145 + 235 + \mathbf{1567} + 234(\mathbf{1} + \mathbf{5} + \mathbf{6} + \mathbf{7}) \\
&= 12 + 13 + 1567 + 2346 + 2347 + 235, \\
\sigma_{(v;V_f)}(f) &= 13 + 145 + 234 + 235 + \mathbf{345} + 12(\mathbf{3} + \mathbf{4} + \mathbf{5}) \\
&= 124 + 125 + 13 + 145 + 234 + 235 + 345, \\
\sigma_{(w;V_f)}(f) &= 12 + 13 + 145 + 235 + \mathbf{15} + 234(\mathbf{1} + \mathbf{5}) \\
&= 12 + 13 + 15 + 235.
\end{aligned}$$

Let f be a function on the variable set $V = \{1, 2, \ldots, n\}$. For a variable set $I \subseteq V$, the *projection* of f on I, denoted by $Proj_I(f)$, is the function on I obtained from f by fixing $x_i = 0$ for all $x_i \in V \setminus I$, i.e.,

$$Proj_I(f)(x_1, x_2, \ldots, x_{|I|}) = f(x_1, x_2, \ldots, x_{|I|}, 0, 0, \ldots, 0)$$

if $I = \{x_1, x_2, \ldots, x_{|I|}\}$. For a variable set $J \supseteq V$, the *expansion* of f to J, denoted by $Exp_J(f)$, is the function on J obtained from f by adding irrelevant variables $x_i \in J \setminus V$. By definition, f and its expansion can be represented by the same DNF.

For $I \supseteq V_f$, we have

$$\sigma_{(v;I)}(f) = Exp_V(\rho_{v[I]}(Proj_I(f))). \tag{8}$$

Thus σ has properties similar to those of ρ. See, for example, Theorem 3.2 below.

Now, for a specified class $\mathcal{C}(n)$ of positive functions of n variables, we say that ρ (resp., σ) *preserves* $\mathcal{C}(n)$ if $\rho_v(f) \in \mathcal{C}(n)$ holds for all $f \in \mathcal{C}(n)$ and $v \in \min T(f)$ (resp., $\sigma_{(v;I)}(f) \in \mathcal{C}(n)$ holds for all $f \in \mathcal{C}(n)$, $v \in \min T(f)$, and $I \supseteq V_f$).

THEOREM 3.2. *The operations ρ and σ defined above preserve the classes $\mathcal{C}_{SD}(n)$, $\mathcal{C}_{DMA}(n)$, and $\mathcal{C}_{DMI}(n)$.*

Proof. This theorem is proved for ρ in [9]. Consider any function $f \in \mathcal{C}_{SD}(n)$ and any set I satisfying $V_f \subseteq I \subseteq V$. If $f = f^d$, then clearly $Proj_I(f) = Proj_I(f^d)$. We thus have $Proj_I(f) \in \mathcal{C}_{SD}(|I|)$, and hence $\rho_v(Proj_I(f)) \in \mathcal{C}_{SD}(|I|)$ by the above-cited result in [9]. It is clear that, for any $g \in \mathcal{C}_{SD}(|I|)$, we have $Exp_V(g) \in \mathcal{C}_{SD}(n)$. Thus by (5), $\sigma_{(v;I)}$ preserves $\mathcal{C}_{SD}(n)$, similarly for $\mathcal{C}_{DMA}(n)$ and $\mathcal{C}_{DMI}(n)$. □

Note that if f is self-dual, then $\rho_v(f)$, $v \in \min T(f)$, is specified simply by

$$T(\rho_v(f)) = (T(f) \setminus \{v\}) \cup \{\overline{v}\}, \tag{9}$$

i.e., by interchanging v with $\overline{v}$ in $T(f)$. This follows from (5) and the fact that $\rho_v(f) \in \mathcal{C}_{SD}(n)$, hence $|T(\rho_v(f)| = |T(f)| = 2^{n-1}$. To see the effect of $\sigma_{(v;I)}$ on $T(f)$, where $V_f \subseteq I \subseteq V$, define

$$v[I]^{*} = \{u \in \{0,1\}^n \mid u[I] = v[I]\}.$$

[3] We sometimes represent a variable set as an index set; e.g., $\{x_1, x_2\}$ is represented as $\{1, 2\}$.

It is easy to see that

$$T(\sigma_{(v;I)}(f)) = (T(f) \setminus v[I]_{\underline{*}}) \cup \overline{v}[I]_{\underline{*}}. \tag{10}$$

To see the difference between (9) and (10), refer to Example 3.1, where $I = V_f$.

Now consider a sequence of transformations from a positive self-dual function f to another positive self-dual function g,

$$\begin{array}{ccccccccc} f_0\,(=f) & \longrightarrow & f_1 & \longrightarrow & f_2 & \longrightarrow & \cdots & \longrightarrow & f_{m_1}\,(=g), \\ g_0\,(=f) & \longrightarrow & g_1 & \longrightarrow & g_2 & \longrightarrow & \cdots & \longrightarrow & g_{m_2}\,(=g), \end{array}$$

where $f_{i+1} = \rho_{v^{(i)}}(f_i)$, $v^{(i)} \in \min T(f_i)$, $g_{i+1} = \sigma_{(w^{(i)};I_i)}(g_i)$, $w^{(i)} \in \min T(g_i)$, and $I_i \supseteq V_{g_i}$. We can see that $m_1, m_2 \geq |\min T(f) \setminus \min T(g)|$ and $m_1 \geq |T(f) \setminus T(g)|$. The latter implies that m_1 might be exponential in n and $|\min T(f)|$, while m_2 might be small. In the next section, we consider the ρ and σ operations for regular self-dual functions, and give a transformation algorithm between any two regular self-dual functions f and g of n variables, which satisfy

$$m_2 \leq |\min T(f)| + |\min T(g)| - 2.$$

4. Transformation of regular self-dual functions. The goal of this section is to present an efficient algorithm, TRANS-REG-SD, which transforms a given regular self-dual function f to the one-variable regular self-dual function $g = x_1$. It applies a sequence of σ operations to f, generating a sequence of regular self-dual functions in the process. As we will show, this algorithm can be used to transform a given regular self-dual function of n variables to any other regular self-dual function of n variables, some of which may be irrelevant. We need to prove a number of lemmas to achieve this goal.

We start with the following lemma, which shows that ρ_v preserves profile-monotonicity (i.e., regularity) if v satisfies a certain condition. (We have already seen that ρ_v preserves self-duality.)[4] Recall that $\rho_v(f)$ is specified by (9), and therefore in the proof we concentrate on the vectors v and $\overline{v}$.

LEMMA 4.1. *Let f be a regular self-dual function, and let $v \in \min T(f)$. $\rho_v(f)$ is regular if and only if $v \in \min_{\geq} T(f)$ and $\overline{v} \not\prec v$.*

Proof. By definition, $\rho_v(f)$ is regular (i.e., profile-monotone) if

$$\rho_v(f)(u) \leq \rho_v(f)(w) \text{ for any } u \prec w. \tag{11}$$

Let us first consider the only-if part. Recall that $\rho_v(f)$ can be specified by (9). Thus we have $\rho_v(f)(v) = 0$ and $\rho_v(f)(\overline{v}) = 1$, which, together with (11), implies $\overline{v} \not\prec v$. Moreover, since $v \in \min T(f)$, if $v \notin \min_{\geq} T(f)$, then there exists a vector $u \in \min_{\geq} T(f)$ majorized by v, i.e., $u \prec v$. (See the paragraph after Lemma 2.1.) Now we have $\rho_v(f)(v) = 0$ and $\rho_v(f)(u) = 1$, which contradicts (11) with $w = v$. Thus $v \in \min_{\geq} T(f)$ holds.

We now turn to the proof of the if part and show that $\rho_v(f)$ is profile-monotone. Equation (11) clearly holds if $u, w \notin \{v, \overline{v}\}$, since f is profile-monotone. (See (9).) If $u = v$ in (11), then the left-hand side becomes $\rho_v(f)(v) = 0$ by (9), and (11) holds for any w. Similarly, if $w = \overline{v}$ in (11), then the right-hand side becomes $\rho_v(f)(\overline{v}) = 1$ by (9), and (11) holds for any u. Now assume that $u = \overline{v}$ $(\prec w)$, in which case

[4] As we commented before, the ρ operation is a special case of the σ operation.

$w \neq v$ by the condition in the lemma. Then we have $v \succ \overline{w}$, and $v \in \min_{\geq} T(f)$ implies $f(\overline{w}) = 0$, hence $f(w) = \rho_v(f)(w) = 1$. Thus (11) holds. Finally, assume that $w = v$ ($\succ u$), in which case $u \neq \overline{v}$ by the condition in the lemma. $v \in \min_{\geq} T(f)$ implies $f(u) = 0$, hence $\rho_v(f)(u) = 0$. Thus (11) again holds. □

The following lemma shows how to choose v to be used in $\rho_v(f)$ to guarantee that $\rho_v(f)$ is regular.

LEMMA 4.2. *Let f be a regular self-dual function of $n\,(\geq 2)$ variables. If $v \in \min_{\geq} T(f)$ and $v_n = 1$, then $\rho_v(f)$ is regular.*

Proof. By Lemma 4.1 we have only to show $\overline{v} \not\prec v$. We have $f(v - e^{(n)}) = 0$ from $v \in \min_{\geq} T(f)$. Thus, the self-duality of f implies $f(\overline{v} + e^{(n)}) = 1$, which, together with $v \in \min_{\geq} T(f)$, in turn implies $\overline{v} + e^{(n)} \not\prec v$. It follows from $\overline{v} + e^{(n)} \not\prec v$ and $v_n = 1$ that $\overline{v} \not\prec v$. □

Interestingly, the existence of a vector v satisfying the condition in Lemma 4.2 is equivalent the relevance of x_n to f, as proved in the following lemma.

LEMMA 4.3. *For a regular function f, x_n is relevant to f if and only if there exists a vector $v \in \min_{\geq} T(f)$ such that $v_n = 1$.*

Proof. If such a vector v exists, then we have $f(v) = 1$ and $f(v - e^{(n)}) = 0$ (by $v \in \min_{\geq} T(f)$). Thus x_n is relevant to f.

Conversely, if x_n is relevant, then there exists a vector $w \in \min T(f)$ such that $w_n = 1$, since, otherwise, the MDNF of f does not contain variable x_n, and hence x_n is irrelevant. The proof is complete if we show $w \in \min_{\geq} T(f)$. Assume that $w \notin \min_{\geq} T(f)$. Then there exists $u \in \min_{\geq} T(f)$ such that $u \prec w$. Note that $w \in \min T(f)$ and $u \in \min_{\geq} T(f)$ imply $u \not\leq w - e^{(n)}$. Otherwise, by the profile-monotonicity of f, we would have $f(w - e^{(n)}) \geq f(u) = 1$, a contradiction to $w \in \min T(f)$. From $u \not\leq w - e^{(n)}$ and $u \prec w$, it follows that $u_n = 1$ (possible only if $n \geq 2$), implying the only-if part. □

Lemma 4.2 deals with the case where x_n is relevant to f. Before dealing with the case where x_n is irrelevant to f, we first prove the following proposition.

PROPOSITION 4.4. *Let f be a regular function. For any $i, j \in V$ such that $i < j$, if x_j is relevant to f, then so is x_i.*

Proof. Assume that x_j is relevant, but x_i is not. Then there must be two vectors, v and w, such that $f(v) > f(w)$, where $v_k = w_k$ for all k $(1 \leq k \leq n)$ except $k = j$, $v_j = 1$, and $w_j = 0$. Now define two vectors, $v' = (v_1, \ldots, v_{i-1}, 0, v_{i+1}, \ldots, v_n)$ and $w' = (w_1, \ldots, w_{i-1}, 1, w_{i+1}, \ldots, w_n)$. We thus have $v' \prec w'$. Since x_i is irrelevant, we should have

$$f(v') = f(v) > f(w) = f(w'),$$

a contradiction to the profile-monotonicity (regularity) of f. □

The above proposition implies that x_i is relevant to f if and only if $V_f \supseteq \{1, 2, \ldots, i\}$; in particular, x_n is relevant to f if and only if $V_f = \{1, 2, \ldots, n\} = V$. Corollary 4.5 generalizes Lemma 4.2 to the case where x_n may be irrelevant to f.

COROLLARY 4.5. *Let f be a regular self-dual function such that $|V_f| = i$ (≥ 2). If $v \in \min_{\geq} T(f)$ and $v_i = 1$, then $\sigma_{(v;V_f)}(f)$ is regular.*

Proof. Let $I = V_f$ in (8). Then $\rho_v(Proj_{V_f}(f))$ is a regular function on V_f by Lemma 4.2. This completes the proof, since $Exp_V()$ preserves regularity. □

We now have the theoretical foundation for TRANS-REG-SD. By Lemma 4.2 and Corollary 4.5, if x_n is relevant to a given f, we can use transformation $\rho_v(f)$, with some v, to generate a new regular self-dual function and repeat this procedure as long

as x_n is relevant. Once x_n becomes irrelevant to the newly generated function, f', we use the σ operation with respect to $V_{f'}$, and so forth.

What remains is the discussion of data we need to keep track of in implementing a sequence of σ transformations. It will be used later in computing the complexity of TRANS-REG-SD. To represent the sequence of regular self-dual functions $\{f'\}$ that TRANS-REG-SD generates, we represent each such function f' in terms of $\min T(f')$ and $\min_{\succeq} T(f')$ (see Lemma 4.8). The following proposition and corollary will prepare us for Lemma 4.8. For a vector v, let us introduce the following notation:

$$T_{\succ}(v) = \{w \mid w \succ v\} \quad \text{and} \quad T_{\prec}(v) = \{w \mid w \prec v\}.$$

PROPOSITION 4.6. *$u \in T_{\succ}(v)$ is a minimal member with respect to $\prec$ in $T_{\succ}(v)$ if and only if $prof_u(i) = prof_v(i) + 1$ for some i, $1 \le i \le n$, and $prof_u(k) = prof_v(k)$ for all $k \ne i$.*

Proof. For simplicity, we present an informal "picture proof" using Figure 4.1. In Figure 4.1 (a) and (b), it is clear that the vector v whose profile is represented by the solid staircase is majorized by the vector u whose profile is represented by the dashed staircase and that u is a minimal vector with respect to $\prec$ in $T_{\succ}(v)$. The dashed staircase in Figure 4.1 (c) shows a nonminimal vector w with $prof_w(n) = |ON(w)| = prof_v(n) = |ON(v)|$. w is nonminimal, since it majorizes another vector $u \in T_{\succ}(v)$, whose profile satisfies $prof_u(3) = 1$ and $prof_u(4) = 2$. Similarly, it is easy to see that any member of $T_{\succ}(v)$ violating the conditions of this proposition majorizes another member of $T_{\succ}(v)$. □

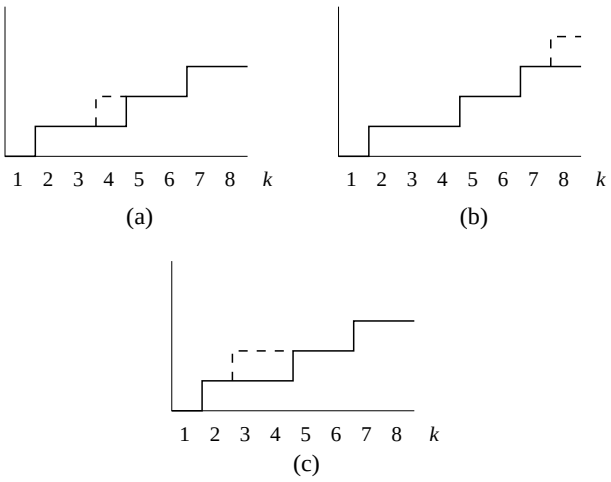


FIG. 4.1. *A "picture proof" of Proposition* 4.6.

Formula (12) in the following corollary follows immediately from the above proposition. Formula (13) is dual to (12).

COROLLARY 4.7.

$$\text{(12)}\quad \min_{\succeq} T_{\succ}(v) = \begin{cases} \{v + e^{(j)} - e^{(j+1)} \mid v_j = 0, v_{j+1} = 1, 1 \le j \le n-1\} & \\ \qquad \cup \{v + e^{(n)}\} & \text{if } v_n = 0, \\ \{v + e^{(j)} - e^{(j+1)} \mid v_j = 0, v_{j+1} = 1, 1 \le j \le n-1\} & \text{if } v_n = 1. \end{cases}$$

$$\max_{\succeq} T_{\prec}(v) = \begin{cases} \{v - e^{(j)} + e^{(j+1)} \mid v_j = 1, v_{j+1} = 0, 1 \le j \le n-1\} \\ \qquad \cup \{v - e^{(n)}\} & \text{if } v_n = 1, \\ \{v - e^{(j)} + e^{(j+1)} \mid v_j = 1, v_{j+1} = 0, 1 \le j \le n-1\} & \text{if } v_n = 0. \end{cases} \tag{13}$$

We now show the effect of operation ρ_v on $\min T(f)$ and $\min_{\succeq} T(f)$.

LEMMA 4.8. *Let f be a regular self-dual function of $n\,(\ge 2)$ variables, and let $v \in \min_{\succeq} T(f)$ with $v_n = 1$. Then we have*

$$\min T(\rho_v(f)) = \min T(f) \setminus (\{v\} \cup \{\overline{v} + e^{(j)} \mid \max OFF(v) < j \le n\}) \cup \{\overline{v}\}, \tag{14}$$

$$\min_{\succeq} T(\rho_v(f)) = \min_{\succeq} T(f) \setminus (\{v\} \cup \min_{\succeq} T_{\succ}(\overline{v})) \cup \{\overline{v}\} \cup \{u \in \min_{\succeq} T_{\succ}(v) \mid u \not\succ z \text{ for all } z \in (\min_{\succeq} T(f) \setminus \{v\}) \cup \{\overline{v}\}\}. \tag{15}$$

Proof. By (9), we need to consider only the influence of (i) adding $\overline{v}$ to $T(f)$ and (ii) removing v from $T(f)$. To follow this proof, it will be helpful to have the following example: For a function $f = 12+13+145+234+235$ of five variables, if $v = (10011)$, then $\overline{v} = (01100)$, $\max OFF(v) = 2$, $\rho_v(f) = 12 + 13 + 234 + 235 + 23 + 145(2 + 3) = 12{+}13{+}23$, $\min_{\succeq} T(f) = \{(10100), (10011), (01101)\}$, and $\min_{\succeq} T(\rho_v(f)) = \{(01100)\}$.

(14)(i). By the self-duality of f, we have $\overline{v} \in \max F(f)$, hence $\overline{v} \in \min T(\rho_v(f))$. (See, e.g., the prime implicant 23 of $\rho_v(f)$ in Example 3.1.) Let us next consider vectors of the form $\overline{v} + e^{(j)}$, $j \in ON(v)\,(= OFF(\overline{v}))$, which may cease to be a minimal member as a result of operation (i). Note that these vectors belong to $T(f)$, since $\overline{v} \in \max F(f)$. We claim that $\overline{v} + e^{(j)} \in \min T(f)$ if and only if $j > \max OFF(v)$. (See then the prime implicants 234 and 235 of f in Example 3.1.) If $j > \max OFF(v)$, $\overline{v} + e^{(j)} \in \min T(f)$ holds, since otherwise there exists a vector $w \in \min T(f)$ such that $w < \overline{v} + e^{(j)}$. Since $w \ne \overline{v}$ and $w < \overline{v} + e^{(j)}$, $w \le \overline{v} + e^{(j)} - e^{(i)}$ holds for some $i \in OFF(v)\,(= ON(\overline{v}))$. Since $i < j$, we have $w \prec \overline{v}$. Since $f(\overline{v}) = 0$ by the self-duality of f, it follows from Lemma 2.1 that $f(w) = 0$ holds, a contradiction to $w \in \min T(f)$. This proves the if part of our claim.

Let $j \in ON(v)$ $(= OFF(\overline{v}))$ with $j \le \max OFF(v)$ and consider the vector $w = \overline{v} + e^{(j)} - e^{(\max OFF(v))}$. Since $j \in ON(v)$, we have $j < \max OFF(v)$. Thus $w \succ \overline{v}$, and hence $v \succ \overline{w}$ holds. If $f(w) = 0$, then $f(\overline{w}) = 1$ holds by the self-duality of f. Since $v \succ \overline{w}$, we have $v \notin \min_{\succeq} T(f)$, a contradiction. Thus $f(w) = 1$, implying that $\overline{v} + e^{(j)} \notin \min T(f)$ holds in this case.

(14)(ii). Since $v \notin \min T(\rho_v(f))$, there may be a $j \in OFF(v)$ such that $u = v + e^{(j)}$ belongs to $\min T(\rho_v(f))$. Since $f(v) = 1$, $f(v + e^{(j)} - e^{(n)}) = 1$, i.e., $v + e^{(j)} - e^{(n)} \in T(f)$, follows from the regularity of f. Thus no vector of the form $v + e^{(j)}$ belongs to $\min T(\rho_v(f))$, since there obviously exists a $w \in \min T(f)$ satisfying $w \le v + e^{(j)} - e^{(n)} < v + e^{(j)}$, a contradiction.

(15)(i). It is easy to see that $\overline{v} \in \min_{\succeq} T(\rho_v(f))$ holds, since otherwise $f(\overline{v}) = 0$ and there exists a vector w such that $w \prec \overline{v}$ and $f(w) = 1$, a contradiction to the regularity of f. Now consider any vector $w \in \min_{\succeq} T(f)$ majorizing $\overline{v}$, i.e., satisfying $w \succ \overline{v}$. Such a w must be removed from $\min_{\succeq} T(f)$ to construct $\min_{\succeq} T(\rho_v(f))$. Clearly, each such w is contained in $\min_{\succeq} T_{\succ}(\overline{v})$ (e.g., $w = (01110) \in \min_{\succeq} T(f)$ corresponding to the prime implicant 234 of f in Example 3.1).

(15)(ii). As noted in (14)(ii), we have $v \notin \min T(\rho_v(f))$, a fortiori, $v \notin \min_{\succeq} T(\rho_v(f))$. Let us consider the vectors in $\min_{\succeq} T_{\succ}(v)$, given by (12), since, besides $\overline{v}$, only they may be contained in $\min_{\succeq} T(\rho_v(f)) \setminus \min_{\succeq} T(f)$. Note that a vector $u \in \min_{\succeq} T_{\succ}(v)$ belongs to $\min_{\succeq} T(\rho_v(f))$, provided there is no vector $z \in \min_{\succeq} T(f) \setminus \{v\} \cup \{\overline{v}\}$ such that $z \prec u$. This is because $\min_{\succeq} T(f) \setminus (\{v\} \cup \min_{\succeq} T_{\succ}(\overline{v})) \cup \{\overline{v}\} \subseteq \min_{\succeq} T(\rho_v(f))$ (see (15)(i) above) and all vectors in $\min_{\succeq} T_{\succ}(\overline{v})$ majorizing $\overline{v}$. ∎

From the proof of Lemma 4.8 (case (14)(i)), we can see that $\overline{v} + e^{(n)} \in \min T(f)$. Since $v_n = 1$ implies $n > \max OFF(v)$, $\{\overline{v} + e^{(j)} \mid \max OFF(v) < j \leq n\}$ is nonempty, and (14) implies the following lemma.

LEMMA 4.9. *Let f be a regular self-dual function of $n\,(\geq 2)$ variables, and let $v \in \min_{\succeq} T(f)$ with $v_n = 1$. Then*

$$|\min T(\rho_v(f))| \leq |\min T(f)| - 1, \tag{16}$$

$$\min T(\rho_v(f))_{x_n=1} \cup \{v, \overline{v} + e^{(n)}\} = \min T(f)_{x_n=1}, \tag{17}$$

where $S_{x_n=1}$ denotes the set $\{v \in S \mid v_n = 1\}$.

We are now ready to describe the transformation algorithm, TRANS-REG-SD. If we repeatedly apply the ρ_v operation (with different v's, of course) to a regular self-dual function f, until there is no vector $v \in \min_{\succeq} T(f)$ with $v_n = 1$, then by Lemmas 4.2, 4.3, and 4.9 we have a regular self-dual function f' to which x_n is irrelevant. This, together with (17), implies that $|\min T(f)_n|$ is even. Note that f' is not unique; i.e., it depends on the sequence of vectors $v \in \min_{\succeq} T(f)$ with $v_n = 1$ that are used in ρ_v. For example, consider a function $f = 12+13+145+234+235$ of five variables. For vectors $v = (10011)$ and $w = (01101)$, $\rho_v(f) = 12 + 13 + 23$ and $\rho_w(f) = 12 + 13 + 14 + 234$, respectively.

Now $V_{f'} = \{1, 2, \ldots, j_1\}$ holds for some $j_1 \leq n - 1$. If $j_1 = 1$, we have $f' = x_1$ and we are done. If $j_1 \neq 1$, on the other hand, we apply $\sigma_{(v;V_{f'})}$ operations to f' instead of $\sigma_{(v;V_f)}\,(= \rho_v)$ until there is no vector $v \in \min_{\succeq} T(f')$ with $v_{j_1} = 1$. Since all the lemmas presented in this section are still valid for $\sigma_{(v;V_{f'})}$ and $v_{j_1} = 1$ in place of $\sigma_{(v;V_f)}\,(= \rho_v)$ and $v_n = 1$, we obtain a regular self-dual function f'' whose relevant variable set is $V_{f''} = \{1, 2, \ldots, j_2\}$ with $j_2 < j_1$. By repeating this argument, we reach the one-variable regular self-dual function x_1. Formally, this sequence of transformations can be stated as follows.

Algorithm TRANS-REG-SD

Input: $\min T(f)$, where f is a regular self-dual function.

Output: Regular self-dual functions $f_0\,(= f)$, f_1, $f_2, \ldots,$ $f_m\,(= x_1)$.

Step 0: Let $i = 0$ and $f = f_0$.

Step 1: Output f_i. If $f_i = x_1$, then halt.

Step 2: $f_{i+1} = \sigma_{(v^{(i)};V_{f_i})}(f_i)$, where $v^{(i)} \in \min_{\succeq} T(f_i)$ and $v^{(i)}_{\max V_{f_i}} = 1$. $i := i+1$. Return to Step 1.

By (16), the number m in the output from algorithm TRANS-REG-SD satisfies $m \leq |\min T(f)| - 1$. Since every self-dual function f satisfies $\rho_{\overline{v}}(\rho_v(f)) = f$ (see (9)), we can transform x_1 into any regular self-dual function g by repeatedly applying the σ operation to x_1 at most $|\min T(g)| - 1$ times. Thus we have the following theorem.

THEOREM 4.10. *Let f and g be any two regular self-dual functions. Then f can be transformed into g by repeatedly applying σ operations to f at most $|\min T(f)| + |\min T(g)| - 2$ times.*

In the subsequent sections, we consider the problems of generating all regular self-dual functions and of computing an optimum self-dual function with respect to a "g-regular" functional Φ (for the definition of g-regularity, see section 6) as applications of algorithm TRANS-REG-SD.

5. Generation of all regular self-dual functions. Let $\mathcal{C}_{R\text{-}SD}(n)$ denote the class of all regular self-dual functions of n variables. We present in this section an algorithm to generate all functions in $\mathcal{C}_{R\text{-}SD}(n)$ by applying the operation σ. The algo-

rithm is *incrementally polynomial* [22] in the sense that the ith function $\phi_i \in \mathcal{C}_{R\text{-}SD}(n)$ is output in polynomial time in n and $\sum_{j=0}^{i-1} |\min T(\phi_j)|$ for $i = 1, 2, \ldots, |\mathcal{C}_{R\text{-}SD}|$.

To visualize the algorithm, we first define an undirected graph $G_n = (\mathcal{C}_{R\text{-}SD}(n), E)$, where $(g, f) \in E$, if there exists a vector $v \in \min_{\geq} T(g)$ such that $\sigma_{(v;I)}(g) = f$ for some $I \supseteq V_g$.

Example 5.1. Figure 5.1 shows the graph G_5.[5] (Ignore the arrows on some edges for now.)

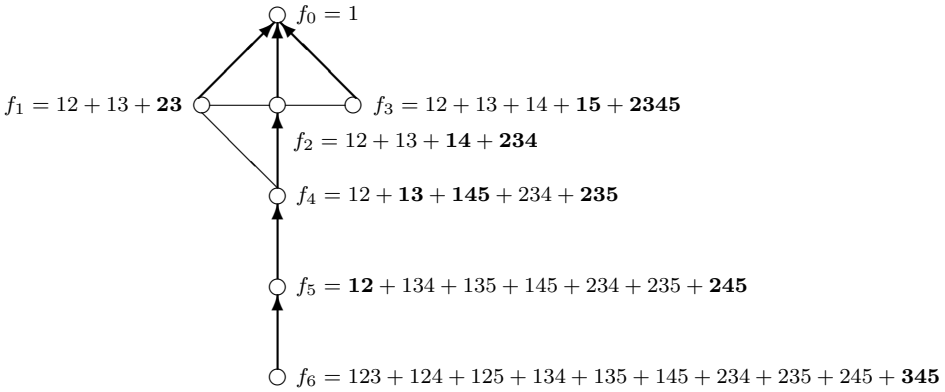


FIG. 5.1. *The graph G_5: the prime implicants corresponding to the vectors in $\min_{\geq} T(f_i)$ are set in boldface.*

Theorem 4.10 implies that G_n is connected. Moreover, the condition $(g, f) \in E$ holds if and only if $(f, g) \in E$; i.e., G_n is indeed undirected, as shown by the following proposition. For a vector $v \in \{0, 1\}^n$ and $I \subseteq V$, let $v[I]\underline{0}$ denote the vector u defined by $ON(u) = ON(v) \cap I$; i.e., $u[I] = v[I]$ and the remaining components of u, if any, are all set to 0's.

PROPOSITION 5.2. *Let $f \in \mathcal{C}_{R\text{-}SD}(n)$ and $g = \sigma_{(w;I)}(f)$ for $w \in \min_{\geq} T(f)$ such that $I \supseteq V_f$ and $\overline{w}[I] \not\leq w[I]$. Then $g \in \mathcal{C}_{R\text{-}SD}(n)$ and $f = \sigma_{(u;I)}(g)$, where $u = \overline{w}[I]\underline{0} \in \min_{\geq} T(g)$.*

Proof. $g \in \mathcal{C}_{R\text{-}SD}(n)$ follows from Theorem 3.2 and Lemma 4.1. From (10), we have

$$T(g) = (T(f) \setminus w[I]\underline{*}) \cup \overline{w}[I]\underline{*}.$$

Let $f' = \sigma_{(u;I)}(g)$. Then

$$T(f') = (T(g) \setminus \overline{w}[I]\underline{*}) \cup w[I]\underline{*}.$$

We thus have $T(f') = T(f)$, hence $f' = f$. $u = \overline{w}[I]\underline{0} \in \min_{\geq} T(g)$ follows from Lemma 4.1 and the fact that f is regular. ∎

[5] Note that in this section the subscripts of the functions $\{f_i\}$ are reversed from those used in TRANS-REG-SD; for example, f_0 now denotes the function x_1.

Example 5.3. For example, consider the function $f = f_2 = 12 + 13 + 14 + 234$ in Figure 5.1, where we assume $n = 5$. We have $\min_{\succeq} T(f_2) = \{(10010), (01110)\}$. Pick $w = (10010) \in \min_{\succeq} T(f_2)$, which satisfies $\overline{w}[I] \not\succ w[I]$ for $I = V_{f_4}$. It is easy to see that $g = \sigma_{(w;I)}(f_2) = 12+13+14(2+3+5)+235+234 = 12+13+145+234+235 = f_4$. For $u = \overline{w} = (01101) \in \min_{\succeq} T(g)$, we have $\sigma_{(u;I)}(g) = 12+13+234+235(1+4)+14 = f_2$. Note that $\overline{w}[I]\underline{0} = \overline{w}[I]$ since $I = V$.

For two distinct vectors $u = (u_1, \ldots, u_n)$ and $v = (v_1, \ldots, v_n)$, we say that u is *lexicographically smaller* than v, written $u \lessdot v$, if for some k $(1 \le k \le n)$ $u_k < v_k$ and $u_i = v_i$ for all i $(1 \le i < k)$. Thus among the vectors in $\{0,1\}^3$, for example, we have $(000) \lessdot (001) \lessdot (010) \lessdot (011) \lessdot (100) \lessdot (101) \lessdot (110) \lessdot (111)$.

Let $f_0 = x_1$ be the designated function in $\mathcal{C}_{R\text{-}SD}(n)$ and consider the problem of transforming an arbitrary function $g \in \mathcal{C}_{R\text{-}SD}(n)$ to f_0 by repeatedly applying the σ operation as in algorithm TRANS-REG-SD. Note that the transformation path from a given g to f_0 is not unique. Thus, to make the path unique, we choose for the σ operation the lexicographically smallest vector $\tilde{v} \in \min_{\succeq} T(g)$ such that $\tilde{v}_{\max V_g} = 1$. Let μ be such an operation, i.e.,

$$\mu(g) = \sigma_{(\tilde{v};V_g)}(g). \tag{18}$$

In this way, we define a directed spanning tree of G_n, $RT_n = (\mathcal{C}_{R\text{-}SD}(n), A_{RT})$, such that (g, f) is a directed arc in A_{RT} if and only if $\mu(g) = f$. If $\mu()$ is applied recursively to g, we eventually reach a function h such that $V_h \subset V_g$ (see (17)). For example, in Figure 5.1, we have $V_{f_2} \subset V_{f_4}$. Thus, RT_n is an in-tree rooted at $f_0 = x_1$. In Figure 5.1, A_{RT} is indicated by the thick arcs.

Our algorithm GEN-REG-SD presented later in this section, which generates all regular self-dual functions of n variables for a given n, will traverse RT_n from f_0 in a depth-first manner in the reverse direction of the arcs in A_{RT}, outputting each regular self-dual function f when it first visits f. This type of enumeration is called *reverse search* in [2, 3] and has been applied to many other enumeration problems such as the extreme points of a convex polyhedron, the arrangements of hyperplanes, the triangulations of a polygon, matroid bases, and so on. When RT_n is traversed from f_0, for each arc $(g, f) \in A_{RT}$, the end node f (nearer the root) is visited before the end node g (farther from the root). Unfortunately, when we first visit node f we cannot identify the incoming arcs (in A_{RT}) towards node f from among the edges in E (of G_n). In other words, knowing f, we cannot find g such that $(g, f) \in A_{RT}$. Note that (18) computes f given g, not the other way around. In Lemma 5.5 below, we find the "inverse" of (18) in the sense that u in Proposition 5.2 coincides with $\tilde{v}$ in (18).

The following lemma is essentially a restatement of Lemma 4.8 in a slightly generalized form.

LEMMA 5.4. *Let $f, g \in \mathcal{C}_{R\text{-}SD}(n)$ satisfy $f = \mu(g) = \sigma_{(\tilde{v};V_g)}(g)$, where $\tilde{v}$ is the lexicographically smallest vector in $\min_{\succeq} T(g)_{x_{\max V_g}=1}$. With $w = \overline{(\tilde{v})}[V_g]\underline{0}$, we have*

$$\min T(f) = \min T(g) \setminus (\{\tilde{v}\} \cup \{w + e^{(j)} \mid \max(OFF(\tilde{v}) \cap V_g) < j \le \max V_g\}) \cup \{w\}, \tag{19}$$

$$\min T(g) = (\min T(f) \setminus \{w\}) \cup \{\tilde{v}\} \cup \{w + e^{(j)} \mid \max(ON(w) \cap V_g) < j \le \max V_g\}, \tag{20}$$

$$\min_{\succeq} T(f) = \min_{\succeq} T(g) \setminus (\{\tilde{v}\} \cup \min_{\succeq} T_{\succ}(w; V_g)) \cup \{w\} \cup \{u \in \min_{\succeq} T_{\succ}(\tilde{v}; V_g) \mid u \not\succ z \text{ for all } z \in (\min_{\succeq} T(g) \setminus \{\tilde{v}\}) \cup \{w\}\}, \tag{21}$$

$$\min_{\succeq} T(g) = \min_{\succeq} T(f) \setminus (\{w\} \cup \min_{\succeq} T_{\succ}(\tilde{v}; V_g)) \cup \{\tilde{v}\} \cup \{u \in \min_{\succeq} T_{\succ}(w; V_g) \mid u \not\succ z \text{ for all } z \in (\min_{\succeq} T(f) \setminus \{w\}) \cup \{\tilde{v}\}\}, \tag{22}$$

where $T_{\succ}(v; I) = \{u \mid u \succ v, ON(u) \subseteq I\}$ *for a vector* v *and an index set* $I \subseteq V$.

Proof. The proof follows from Lemma 4.8, which is a special case of this lemma ($V_g = V$). □

Note that by (20) we have $w + e^{(\max V_g)} \in \min T(g)$, implying $\tilde{v} \lessdot w + e^{(\max V_g)}$, hence $\tilde{v}_1 = 0$ and $w_1 = 1$, since $\overline{w}_1 = \tilde{v}_1$. In Lemma 5.4, conceptually, $\tilde{v}$ was explicitly chosen first, and w was specified in terms of $\tilde{v}$. The following lemma will enable us to choose w explicitly, so that $\tilde{v}$ is chosen implicitly. Note that condition (c) in Lemma 5.5 involves the lexicographic order in $\min_{\succeq} T(f)$, which one can compute given f, while $\tilde{v}$ in (18) is defined in terms of the lexicographically smallest vector in $\min_{\succeq} T(g)$, which one can compute given g. Note also that $\tilde{v}$ is unique for a given regular self-dual function g, but vector w which satisfies the conditions of Lemma 5.5 is in general not unique for a given regular self-dual function f. This reflects the fact that a nonroot node in graph RT_n has one parent but in general has more than one child node.

LEMMA 5.5. *Let* $f \in \mathcal{C}_{R\text{-}SD}(n)$ *and* $g = \sigma_{(w;V_g)}(f)$ *for* $w \in \min_{\succeq} T(f)$[6] *such that* $\overline{w}[V_g] \not\leq w[V_g]$ *and* $V_g \supseteq V_f$. *Then* $f = \mu(g)$ $(= \sigma_{(\tilde{v};V_g)}(g))$ *i.e.,* $(g, f) \in A_{RT}$ *(the arc set of* RT_n*), if and only if*

(a) $w_{\max V_g} = 0$,
(b) $w_1 = 1$, *and*
(c) $\overline{w}[V_g]\underline{0}$ *is lexicographically smaller than any vector in* $\min_{\succeq} T(f)_{x_{\max V_g}=1}$.

Proof. Let us first consider the only-if part, assuming $f = \sigma_{(\tilde{v};V_g)}(g)$, where $\tilde{v}$ is the lexicographically smallest vector in $\min_{\succeq} T(g)_{x_{\max V_g}=1}$ as in Lemma 5.4. Then $g = \sigma_{(w;V_g)}(f)$ implies that $w = \overline{(\tilde{v})}[V_g]\underline{0}$. Since $\tilde{v}_{\max V_g} = 1$ by definition, we have $w_{\max V_g} = 0$, hence (a) holds.

(b) was shown above in the comment immediately after Lemma 5.4. To prove (c), by (21), it suffices to show that $\tilde{v} \lessdot w$ and that $\tilde{v}$ is lexicographically smaller than any vector in the set $\min_{\succeq} T_{\succ}(\tilde{v}; V_g)$, since any vector u in the first term in (21) with $u_{\max V_g} = 1$ must be lexicographically larger than $\tilde{v}$ by definition of $\tilde{v}$. The former follows immediately from (b) and $\tilde{v} = \overline{w}[V_g]\underline{0}$, and the latter is obvious, since $u \prec v$ implies that $u \lessdot v$ for any vectors u and v.

To prove the if part, let $\tilde{v} = \overline{w}[V_g]\underline{0}$. We want to show that $\tilde{v}$ is lexicographically the smallest in $\min_{\succeq} T(f)_{x_{\max V_g}=1}$. By (c) and (22), it suffices to show that $\tilde{v}$ is lexicographically smaller than any vector in $\min_{\succeq} T_{\succ}(w; V_g)$ (the last term in (22)). This is obvious since $w_1 = 1$ and $\tilde{v}_1 = 0$. □

Example 5.6. Using Lemma 5.5, we can construct $RT_n = (\mathcal{C}_{R\text{-}SD}(n), A_{RT})$. $RT_6 = (\mathcal{C}_{R\text{-}SD}(6), A_{RT})$ is shown in Figure 5.2.

The function numbers in the figure denote those regular self-dual functions of six variables shown in the following table, which was derived from the work by Bioch and Ibaraki [8]. (Some function numbers have been changed.)

[6] This means that $(g, f) \in E$ (the edge set of G_n). g depends on the choice of w.

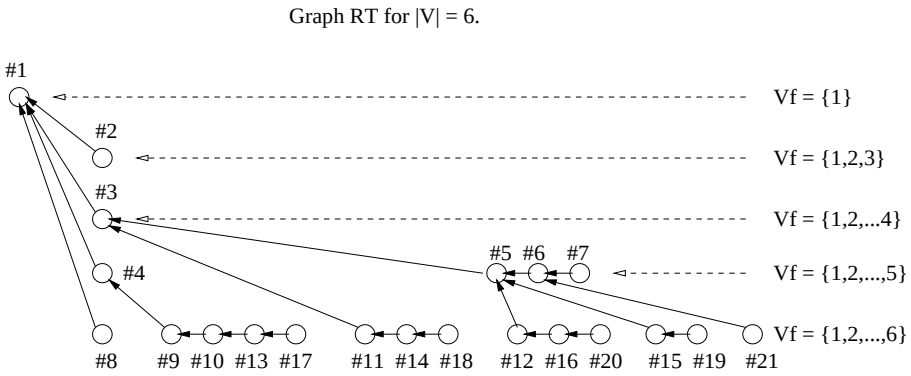


FIG. 5.2. $RT_6 = (\mathcal{C}_{R\text{-}SD}(6), A_{RT})$.

#	$\lvert V_f\rvert$	Function
1	1	$\mathbf{1}^*$
2	3	12+13+$\mathbf{23}^\dagger$
3	4	12+13+$\mathbf{14}^*$+$\mathbf{234}^\dagger$
4	5	12+13+14+$\mathbf{15}^*$+$\mathbf{2345}^\dagger$
5		12+$\mathbf{13}^*$+$\mathbf{145}^*$+234+$\mathbf{235}^\dagger$
6		$\mathbf{12}^*$+134+135+145+234+235+$\mathbf{245}^\dagger$
7		123+124+125+134+135+145+234+235+245+$\mathbf{345}^\dagger$
8	6	12+13+14+15+$\mathbf{16}$+$\mathbf{23456}^\dagger$
9		12+13+$\mathbf{14}^*$+$\mathbf{156}$+2345+$\mathbf{2346}^\dagger$
10		12+$\mathbf{13}^*$+145+146+$\mathbf{156}$+2345+2346+$\mathbf{2356}^\dagger$
11		12+$\mathbf{13}^*$+145+$\mathbf{146}$+$\mathbf{234}$+$\mathbf{2356}^\dagger$
12		12+$\mathbf{13}^*$+234+235+$\mathbf{236}^\dagger$ +$\mathbf{1456}$
13		$\mathbf{12}^*$+134+135+136+145+146+$\mathbf{156}$+2345+2346+2356 +$\mathbf{2456}^\dagger$
14		$\mathbf{12}^*$+134+135+136+145+$\mathbf{146}$+$\mathbf{234}$+2356+$\mathbf{2456}^\dagger$
15		$\mathbf{12}^*$+134+135+$\mathbf{136}$+$\mathbf{145}$+234+$\mathbf{235}$+$\mathbf{2456}^\dagger$
16		$\mathbf{12}^*$+134+135+136+234+235+$\mathbf{236}$+1456+$\mathbf{2456}^\dagger$
17		123+124+125+126+134+135+136+145+146+$\mathbf{156}$+2345+2346+2356 +2456+$\mathbf{3456}^\dagger$
18		123+124+125+126+134+135+136+145+$\mathbf{146}$+$\mathbf{234}$+2356+2456+$\mathbf{3456}^\dagger$
19		123+124+125+126+134+135+$\mathbf{136}$+$\mathbf{145}$+234+$\mathbf{235}$+2456+$\mathbf{3456}^\dagger$
20		123+124+125+126+134+135+136+234+235+$\mathbf{236}$+1456+2456+$\mathbf{3456}^\dagger$
21		123+124+125+$\mathbf{126}$+134+135+145+234+235+$\mathbf{245}$+$\mathbf{3456}^\dagger$

In the table, the prime implicants corresponding to the vectors in $\min_{\succeq} T(f)$ are shown in boldface. Let t_w denote the prime implicant corresponding to vector $w \in \min T(f)$; i.e., t_w contains the variable x_i as a factor if and only if $i \in ON(w)$. In other words, $i \in P(t_w)$ if and only if $i \in ON(w)$ and $N(t_w) = \emptyset$. (For the definition of $P()$ and $N()$, see the beginning of section 2.) * indicates a prime implicant t_w such that vector w satisfies the conditions of Lemma 5.5. When each function is called g, the prime implicant t_v with † corresponds to the unique vector $\tilde{v}$.

In general, each vector w with $w_{\max V_f} = 1$ implies that f has $(n - \max V_f)$ children due to w in RT_n, one each for $|V_g| = |V_f| + 1, \ldots, n$. Similarly, each vector w with $w_{\max V_f} = 0$ implies that f has $(n - \max V_f + 1)$ children due to w, one each for $|V_g| = |V_f|, |V_f| + 1, \ldots, n$. For example, it is observed in Figure 5.2 that $t_w = 14$ in function #3 gives rise to $6-4 = 2$ children, while $t_w = 15$ in function #4 gives rise

to just 6−5=1 child. Similarly, $t_w = 145$ in function #5 gives rise to just one child, while $t_w = 13$ in function #5 gives rise to $6-5+1 = 2$ children, and so forth.

Note that if $V_g \neq V_f$ (i.e., $V_g \supset V_f$), Lemma 5.5 implies that $f = \mu(g)$ if and only if $w_1 = 1$, since $V_g \supset V_f$ and $w \in \min_{\geq} T(f)$ imply conditions (a) and (c) are vacuous in this case. Thus, for an index set $I \supset V_f$, any vector $w \in \min_{\geq} T(f)$ that satisfies $w_1 = 1$ and $\overline{w}[I] \not\succ w[I]$ produces $g = \sigma_{(w;I)}(f)$ such that $f = \mu(g)$.

We now discuss the data structures for $\min T(f)$ and $\min_{\geq} T(f)$. The set $\min T(f)$ is represented by a *binary tree*, denoted by $B(\min T(f))$, of height n, in which the left edge (resp., right edge) from a node at depth $j-1$ (the root is at depth 0) represents the case $x_j = 1$ (resp., $x_j = 0$). A leaf node t of $B(S)$ at depth n stores the vector $v \in S\,(\subseteq \{0,1\}^n)$, the components of which correspond to the edges of the path from the root to t. In order to have a compact representation, the edges with no descendant leaves are removed from $B(S)$.

Example 5.7. Figure 5.3 shows $B(\min T(f))$ for $\min T(f) = \{v^{(1)} = (110000), v^{(2)} = (101000), v^{(3)} = (100110), v^{(4)} = (011100), v^{(5)} = (011010)\}$ (i.e., $f = 12 + 13 + 145 + 234 + 235$).

With this data structure, it is easy to see that we can apply operations MEMBER (i.e., check if $v \in S$), INSERT (i.e., update $S := S \cup \{v\}$), and DELETE (i.e., update $S := S \setminus \{v\}$) all in $O(n)$ time. Moreover, since the rightmost (resp., leftmost) path in $B(S)$ represents the lexicographically smallest (resp., largest) vector in S, we can output from $B(S)$ the lexicographically smallest/largest vector in S in $O(n)$ time.

Let $V_i = \{1, 2, \ldots, i\}$ and define $\alpha : \{0,1\}^n \to \mathbb{Z}^+$ by

$$\alpha(v) = \min(\{\, i \mid \overline{v}[V_i] \not\succ v[V_i],\ ON(v) \subseteq V_i \subseteq V\,\} \cup \{n+1\}). \tag{23}$$

By definition, if no i satisfies the condition on the right-hand side, $\alpha(v) = n+1$ holds. We can easily see that $\overline{v}[V_i] \not\succ v[V_i]$ holds for all $i \geq \alpha(v)$, if $\alpha(v) < n+1$. Based on this $\alpha(v)$ and $\max ON(v)$, we decompose $\min_{\geq} T(f)$ into many subsets as follows: $\min_{\geq} T(f) = \bigcup_{\substack{j \leq i \leq n+1 \\ j=1,2,\ldots,n}} \min_{\geq} T(f)_{(i,j)}$, where

$$\min_{\geq} T(f)_{(i,j)} = \{v \in \min_{\geq} T(f) \mid \alpha(v) = i,\ \max ON(v) = j\}.$$

For the above example function $f = 12 + 13 + 145 + 234 + 235$, we have $\min_{\geq} T(f) = \{v^{(2)} = (101000), v^{(3)} = (100110), v^{(5)} = (011010)\}$. From this, we get $\min_{\geq} T(f)_{(5,3)} = \{v^{(2)}\}$, $\min_{\geq} T\ (f)_{(5,5)} = \{v^{(3)}, v^{(5)}\}$, and $\min_{\geq} T(f)_{(i,j)} = \emptyset$, otherwise.

Our algorithm keeps $\min_{\geq} T(f)$ as $\bigcup_{\substack{j \leq i \leq n+1 \\ j=1,2,\ldots,n}} B(\min_{\geq} T(f)_{(i,j)})$ and $\bigcup_{j=1}^{n} B(\min_{\geq} T(f)_j)$, where $\min_{\geq} T(f)_j = \bigcup_{i=j}^{n+1} \min_{\geq} T(f)_{(i,j)}$.

We start depth-first search from the root f_0. Note that $\min_{\geq} T(f_0)_{(1,1)} = (10\ldots0)$ and $V_{f_0} = \{1\}$. During the depth-first search, when we visit node f, we first set up $I := V_f$ as the index set. In the order $(i,j) = (1,1), \ldots, (1, \max I - 1), (2,1), \ldots, (2, \max I - 1), \ldots, (\max I, 1), \ldots, (\max I, \max I - 1)$, we then check if the lexicographically largest $w^{(i,j)}$ in $\min_{\geq} T(f)_{(i,j)}$ satisfies (b) $w_1^{(i,j)} = 1$ and (c) that the vector $v^{(i,j)}$, defined by $ON(v^{(i,j)}) = OFF(w^{(i,j)}) \cap I$, is lexicographically smaller than any vector in $\min_{\geq} T(f)_{\max I}$ (i.e., $\min_{\geq} T(f)_{x_{\max I}=1}$). If there exists such a vector $w^{(i,j)}$, we move to $g = \sigma_{(w^{(i,j)};I)}(f)$. Since $w^{(i,j)}$ satisfies conditions (a), (b), and (c) of Lemma 5.5, we have $f = \mu(g)$. Moreover, if $w^{(i,j)}$ does not satisfy (b) (resp., (c)), then no vector in $u \in \min_{\geq} T(f)_{(i,j)}$ satisfies (b) (resp., (c)). This means that we do not have to check the vectors in $\min_{\geq} T(f)_{(i,j)}$ other than $w^{(i,j)}$. Thus, if no $w^{(i,j)}$ $(i = 1, 2, \ldots, \max I,$

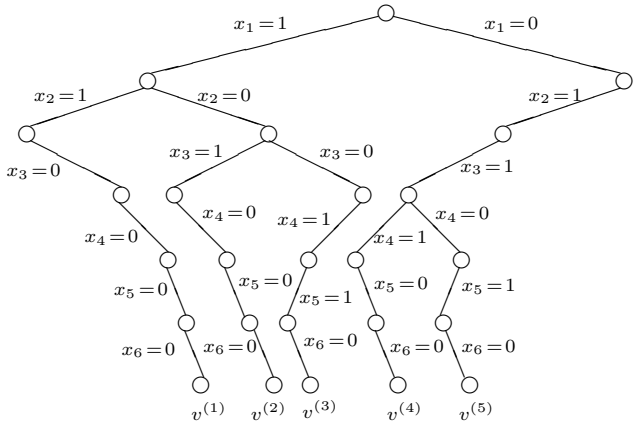


FIG. 5.3. *A binary tree $B(\min T(f))$, where $f = 12 + 13 + 145 + 234 + 235$ (a function of six variables).*

$j = 1, 2, \ldots, \max I - 1$) satisfies (b) and (c), then we check if $\max I = n$. If so, we return to $h = \mu(f)$ from f (if $f = f_0 = x_1$, then halt); otherwise, we update $I := I \cup \{\max I + 1\}$ and again check if $w^{(i,j)}$ satisfies (b) and (c).

When the depth-first search returns to f from g by backtracking (i.e., $f = \mu(g) = \sigma_{(v^{(i^*,j^*)};V_g)}(g)$), we set $I := V_g$ and move on to the vector w' which lexicographically follows $w^{(i^*,j^*)}$ in $\min_{\succeq} T(f)_{(i^*,j^*)}$. If this w' satisfies conditions (b) and (c) of Lemma 5.5, then we move to $g' = \sigma_{(w';I)}(f)$; otherwise, check if $w^{(i,j)}$ satisfies (b) and (c), according to the ordering $(i,j) = (i^*, j^*+1), \ldots, (i^*, \max I - 1), (i^*+1, 1), \ldots, (i^*+1, \max I - 1), \ldots, (\max I, 1), \ldots, (\max I, \max I - 1)$.

This procedure has the advantage that there is no need to maintain the data of the entire search tree RT_n but only the information about the current function is sufficient. Our algorithm can be stated formally as follows:

Algorithm GEN-REG-SD

Input: A positive integer n.

Output: All regular self-dual functions of n variables.

Step 0: Let $f := f_0$, $I := V_f$, $w := (11\cdots1)$, and output f. /* Note that $(11\cdots1)$ is the special vector, indicating that no σ operation is applied to f.*/

Step 1: If $w = (11\cdots1)$, then go to Step 2. /* In this case, no σ operation was applied to f.*/

Else if $I = V_f$, then go to Step 3. /* In this case, $V_g = V_f$ holds for $g = \sigma_{(w,I)}(f)$; i.e., we have applied the σ operation with $I = V_f$ previously.*/

Else, go to Step 4. /* In this case, $V_g \supset V_f$ holds for $g = \sigma_{(w,I)}(f)$; i.e., we have applied the σ operation with $I \supset V_f$ previously, and hence we already applied the σ operation for with V_i, where $V_f \subseteq V_i \subset I$.*/

Step 2: Try to find the lexicographically largest $w' = w^{(i,j)}$ in $\min_{\succeq} T(f)_{(i,j)}$ satisfying the two conditions of Lemma 5.5, (b) $w_1^{(i,j)} = 1$ and (c) the vector $v^{(i,j)}$, defined by $ON(v^{(i,j)}) = OFF(w^{(i,j)}) \cap I$, is lexicographically smaller than any vector in $\min_{\succeq} T(f)_{\max I}$, in the order $(i,j) = (1,1), \ldots, (1, \max I - 1), (2,1), \ldots, (2, \max I - 1), \ldots, (\max I, 1), \ldots, (\max I, \max I - 1)$. /* Recall

$I = V_f$. First apply σ with the index set $I = V_f$.*/

Then consider the following four cases:

(i) There is a w' satisfying the above conditions: Let $f := \sigma_{(w';I)}(f)$ and $w := (11\cdots1)$, output f, and return to Step 1 (downward move).

(ii) There is no such w' and $\max I < n$:[7] Try to find the lexicographically largest $w'' = w^{(i,j)}$ in $\min_{\succeq} T(f)_{(i,j)}$ satisfying (b) $w_1^{(i,j)} = 1$, according to the ordering $(i,j) = (1,1),\ldots,(1,\max V_f),(2,1),\ldots,(2,\max V_f),\ldots,(n,1),\ldots,(n,\max V_f)$.

If there is no such w'', then go to either case (iii) or (iv). Else, let $w'' \in \min_{\succeq} T(f)_{(i^*,j^*)}$. Then let $I := V_{i^*}$ $(= \{1,2,\ldots,i^*\})$ if $i^* \geq \max I + 2$; otherwise, let $I := I \cup \{\max I + 1\}$, $f := \sigma_{(w'';I)}(f)$, and $w := (11\cdots1)$, output f, and return to Step 1 (downward move).

(iii) There is no such w' or w'', and $f = f_0$: Halt (all functions have been output).

(iv) There is no such w' or w'', and $f \neq f_0$: Let v be the lexicographically smallest vector in $\min_{\succeq} T(f)_{\max V_f}$. Update f, I, and w by letting $I := V_f$, $f := \mu(f) = \sigma_{(v;V_f)}(f)$, and $ON(w) = OFF(v) \cap I$, respectively. Return to Step 1 (backtrack).

Step 3: Let $w \in \min_{\succeq} T(f)_{(i^*,j^*)}$. Try to find the vector w' satisfying conditions (b) and (c) of Lemma 5.5, according to the ordering (1) the vector that lexicographically follows w in $\min_{\succeq} T(f)_{(i^*,j^*)}$, followed by (2) $w^{(i,j)}$, where $(i,j) = (i^*,j^*+1),\ldots,(i^*,\max I - 1),(i^*+1,1),\ldots,(i^*+1,\max I-1),\ldots,(\max I,1),\ldots,(\max I,\max I-1)$. /*Try to apply σ with the current I with $I = V_f$.*/

Then consider the four cases of Step 2.

Step 4: Let $w \in \min_{\succeq} T(f)_{(i^*,j^*)}$. Try to find the vector w' satisfying (b) $w'_1 = 1$, according to the ordering (1) the vector that lexicographically follows w in $\min_{\succeq} T(f)_{(i^*,j^*)}$, followed by (2) $w^{(i,j)}$, where $(i,j) = (i^*,j^*+1),\ldots,(i^*,\max V_f),(i^*+1,1),\ldots,(i^*+1,\max V_f),\ldots,(\max I,1),\ldots,(\max I,\max V_f)$. /*Try to apply σ with the current I with $I \supset V_f$.*/

Then consider the four cases of Step 2.

In case (ii), we check if there exist a set $I'' \supset I$ and a vector $w'' \in \min_{\succeq} T(f)$ such that $f = \mu(g)$, where $g = \sigma_{(w'';I'')}(f)$. Since $I'' \supset I \supseteq V_f$, we just check if $w''_1 = 1$, as noted in the second paragraph after Example 5.6. According to the orderings on w' and w'', algorithm GEN-REG-SD traverses RT_n depth-first; i.e., each arc in RT_n is traversed only twice, downward and upward.

To analyze the time complexity of GEN-REG-SD, we need two more lemmas.

LEMMA 5.8. *Given the data structures* $B(\min T(f))$, $B(\min_{\succeq} T(f)_{(i,j)})$ $(j = 1,2,\ldots,n,\ i = j,j+1,\ldots,n+1)$ *and* $B(\min_{\succeq} T(f)_j)$ $(j = 1,2,\ldots,n)$, *each iteration of Steps* $1 \sim 4$ *in algorithm* GEN-REG-SD *computes either* w' *or* w'' (*or concludes that no such vector exists*) *in* $O(n^3)$ *time.*

Proof. Since we can check if a given vector u satisfies (b) and (c) of Lemma 5.5 in $O(n)$ time, and since there are $2n^2$ candidates for either w' or w'', each iteration requires $O(n \times n^2) = O(n^3)$ time. □

LEMMA 5.9. *Let* $f, g \in \mathcal{C}_{R\text{-}SD}(n)$ *satisfy* $f = \mu(g)$. *Let* $f = \sigma_{(v;I)}(g)$ *and* $g = \sigma_{(w;I)}(f)$. *Given the data structures for* g (*i.e.,* $B(\min T(g))$, $B(\min_{\succeq} T(g)_{(i,j)})$ $(i = 1,2,\ldots,n+1,\ j = 1,2,\ldots,n)$, *and* $B(\min_{\succeq} T(g)_j)$ $(j = 1,2,\ldots,n)$), v, *and* I,

[7] Note that $I \supseteq V_f$, and hence $\min_{\succeq} T(f)_j = \emptyset$ holds for all $j \geq \max I + 1$.

we can compute data structures of f in $O(n^3)$ time. Similarly, given data structures of f, w, and I, we can compute data structures of g in $O(n^3)$ time.

Proof. We shall prove only the first assertion of the lemma, since the second assertion can be proved analogously. It directly follows from (19) in Lemma 5.4 that we can compute $B(\min T(f))$ in $O(n^2)$ time.

As for the rest, it suffices to show that, given a vector u, the membership $u \in \min_{\succeq} T(f)$ can be checked in $O(n^2)$ time, since at most n vectors are deleted from or added to $\min_{\succeq} T(g)$ to construct $\min_{\succeq} T(f)$ by (21) and (12). Note that a vector u is contained in $\min_{\succeq} T(f)$ if and only if $f(z) = 0$ holds for all vectors $z \in \max_{\succeq} T_{\prec}(u)$. From (13), we have at most n such vectors z. Moreover, we can check if a given vector z satisfies $f(z) = 1$ in $O(n)$ time if $B(\min T(f))$ is prepared [34]. This completes the proof. □

By Lemmas 5.8 and 5.9, each iteration of Step 1 can be carried out in $O(n^3)$ time, except for the outputting of function f. Since each arc $(g, f) \in A_{RT}$ is traversed twice, algorithm GEN-REG-SD requires $O(n^3|\mathcal{C}_{R\text{-}SD}(n)|)$ time, plus the time for outputting all functions in $\mathcal{C}_{R\text{-}SD}(n)$, i.e., $O(nM_{sum})$ time, where

$$M_{\text{sum}} = \sum_{f \in \mathcal{C}_{R\text{-}SD}(n)} |\min T(f)|.$$

Algorithm GEN-REG-SD clearly requires $O(nM_{\max})$ space, where

$$M_{\max} = \max_{f \in \mathcal{C}_{R\text{-}SD}(n)} |\min T(f)|.$$

Thus we have the following theorem.

THEOREM 5.10. *Algorithm* GEN-REG-SD *generates all functions in $\mathcal{C}_{R\text{-}SD}(n)$ and is incrementally polynomial. It requires $O(n^3|\mathcal{C}_{R\text{-}SD}(n)| + nM_{sum})$ time and $O(nM_{\max})$ space.*

COROLLARY 5.11. *All functions in $\mathcal{C}_{R\text{-}SD}(n)$ can be scanned in $O(n^3|\mathcal{C}_{R\text{-}SD}(n)|)$ time.*

By Lemma 2.2, the regular functions are all *representatives of equivalence classes under permutation*; i.e., there is no regular function f that is equivalent to another regular function $g\,(\neq f)$ under permutation. Therefore, our algorithm generates the nonequivalent functions. Let us remark that the algorithms in [9, 18] are not polynomial if we try to output only nonequivalent functions.

It is known that the positive self-dual functions of up to $n = 5$ variables are all threshold functions (and hence regular if we consider the representatives), but there are many nonregular self-dual functions for $n \geq 6$, even if we consider the representatives (see Example 5.6). Moreover, it is known [28] that all regular positive self-dual functions for $n \leq 9$ are threshold functions.

6. Optimum self-dual function for regular functional Φ. Let φ be a *pseudo-Boolean function*; i.e., φ is a mapping from $\{0,1\}^n$ to the set of real numbers $\mathbb{R}$. A pseudo-Boolean function φ is said to be g-*regular* if $\varphi(v) \geq \varphi(w)$ holds for all pairs of vectors v and w with $v \succeq w$. For a Boolean function f, let

$$\Phi(f) = \sum_{v \in T(f)} \varphi(v), \tag{24}$$

where φ is a pseudo-Boolean function. The functional Φ is also said to be g-*regular* if φ is g-regular. As an example of a g-regular pseudo-Boolean functional of interest,

we cite the *availability* of a Boolean function, defined as follows. Each element $i \in V$ has the operational probability p_i $(0 \leq p_i \leq 1)$; i.e., the ith element is *operational* with the probability p_i, while it is *failed* with the probability $1 - p_i$. We assume that each element takes on its state independently. Then the *availability* $A(f)$ of a Boolean function f is defined by

$$A(f) = \sum_{v \in T(f)} \left(\prod_{i \in ON(v)} p_i \prod_{i \in OFF(v)} (1 - p_i) \right). \tag{25}$$

If we interpret $T(f)$ as the set of states in which the n-element system defined by the Boolean function f is working, then $A(f)$ represents the probability that the system represented by f is working. The availability has been studied extensively [35], especially in the case where f represents ND coteries (i.e., f is positive self-dual) [1, 5, 15, 33, 36, 38]. It is known [1, 36] that any element i with $p_i < 1/2$ can be ignored; i.e., x_i is irrelevant for all positive self-dual functions f having the maximum availability. The only exception is when all the elements have $p_i < 1/2$. In this case, it is known [1] that $f = x_j$ has the maximum availability, where j is the element such that $p_j \geq p_i$ for all i. Thus, we can assume that $p_i \geq 1/2$ holds for all i. Moreover, we assume without loss of generality that

$$p_1 \ \geq \ p_2 \ \geq \ \cdots \ \geq \ p_n \ \geq \ 1/2.$$

Now, let $\varphi(v) = \prod_{i \in ON(v)} p_i \prod_{i \in OFF(v)} (1 - p_i)$. Then we have $\Phi(f) = A(f)$. It follows from the assumption on the order of probabilities that $A(f)$ is g-regular.

In this section, we consider the functions f that maximize g-regular functional Φ among all self-dual functions.

LEMMA 6.1. *Given a* g*-regular function φ, let Φ be a* g*-regular functional defined by* (24)*. Then the following statements regarding f are equivalent.*

(i) *$\Phi(f)$ is maximum among all self-dual functions.*
(ii) *All vectors $v \in T(f)$ satisfy $\varphi(v) \geq \varphi(\overline{v})$.*
(iii) *All vectors $v \in \min_{\succeq} T(f)$ satisfy $\varphi(v) \geq \varphi(\overline{v})$.*

Proof. Let us prove (i) $\Longrightarrow$ (ii) $\Longrightarrow$ (iii) $\Longrightarrow$ (i). Since (ii) $\Longrightarrow$ (iii) is obvious, we show (i) $\Longrightarrow$ (ii) and (iii) $\Longrightarrow$ (i).

(i) $\Longrightarrow$ (ii): If there exists a vector $v \in T(f)$ such that $\varphi(v) < \varphi(\overline{v})$, then the function g defined by $T(g) = (T(f) \setminus \{v\}) \cup \{\overline{v}\}$ satisfies $\Phi(g) > \Phi(f)$. Since g is self-dual, $\Phi(f)$ is not maximum among all self-dual functions.

(iii) $\Longrightarrow$ (i): Assume that $\Phi(f)$ is not maximum among all self-dual functions. Then there exists a self-dual function g such that $\Phi(g) > \Phi(f)$. Since $\Phi(g) > \Phi(f)$, some vector $v \in T(f) \setminus T(g)$ satisfies $\varphi(v) < \varphi(\overline{v})$. For this v, let w be a vector in $\min_{\succeq} T(f)$ such that $w \preceq v$. By the g-regularity of φ, $\varphi(w) \leq \varphi(v)$ holds. Moreover, since $\overline{w} \succeq \overline{v}$ by $w \preceq v$, $\varphi(\overline{w}) \geq \varphi(\overline{v})$ holds. Thus, we have

$$\varphi(w) \leq \varphi(v) < \varphi(\overline{v}) \leq \varphi(\overline{w}).$$

This means that this w satisfies $w \in \min_{\succeq} T(f)$ and $\varphi(w) < \varphi(\overline{w})$. ∎

THEOREM 6.2. *Let $\Phi(f)$ be a* g*-regular functional defined by* (24)*. Then there exists a* regular *self-dual function f which maximizes $\Phi(f)$ among all self-dual functions.*

Proof. Let f be a regular self-dual function that maximizes Φ among all *regular* self-dual functions. We claim that f in fact maximizes Φ among all self-dual functions.

If not, by Lemma 6.1, there exists a vector $v \in \min_{\succeq} T(f)$ such that $\varphi(v) < \varphi(\overline{v})$. Note that $v \not\succeq \overline{v}$ holds, since otherwise (i.e., $v \succeq \overline{v}$) $\varphi(v) \geq \varphi(\overline{v})$, a contradiction. Thus, it follows from Lemma 4.1 that $\rho_v(f)$ is regular and self-dual. Moreover, by (9), we have $\Phi(\rho_v(f)) > \Phi(f)$, which contradicts the assumption. □

However, there may be nonregular functions f that also maximize $\Phi(f)$. For example, let $p_1 = p_2 = 1/2$. Then $f_1 = x_1, f_2 = x_2, f_3 = \overline{x}_1$, and $f_4 = \overline{x}_2$ have the maximum availability, and, clearly, f_2, f_3, and f_4 are not regular. (f_3 and f_4 are not even positive.)

A pseudo-Boolean function φ is said to be *properly* g*-regular* if $\varphi(v) > \varphi(w)$ holds for all pairs of vectors v and w with $v \succ w$. The functional Φ defined by (24) is also said to be *properly* g*-regular* if φ is properly g-regular.

The next theorem is a weak uniqueness result for Theorem 6.2, showing that any optimal coterie is regular if $\Phi(f)$ is properly g-regular.

THEOREM 6.3. *Let $\Phi(f)$ be a* properly g*-regular functional defined by* (24)*. Then any function f that maximizes $\Phi(f)$ among all self-dual functions is regular.*

Proof. Assume that a nonregular function f maximizes $\Phi(f)$ among all self-dual functions. Since f is nonregular, there exists two vector v and w with $v \succ w$ such that $f(v) = 0$ and $f(w) = 1$. By Lemma 6.1, we have $\varphi(w) \geq \varphi(\overline{w})$. This, together with $v \succ w$ (and $\overline{w} \succ \overline{v}$), implies that $\varphi(v) > \varphi(\overline{v})$, which contradicts the assumption by Lemma 6.1. □

The above theorem directly implies the following corollary.

COROLLARY 6.4. *Let p_i, $i = 1, 2, \ldots, n$, be the operational probability of the ith element. If $p_1 > p_2 > \cdots > p_n > 1/2$, then any function f that maximizes the availability $A(f)$ among all self-dual functions is regular.*

Algorithm OPT-REG-SD

Input: A membership oracle of g-regular function φ.

Output: A regular self-dual function f that maximizes $\Phi(f)$ among all self-dual functions.

Step 0: Let $i := 1$ and $f := x_1$.

Step 1: While $\exists v \in \min_{\succeq} T(f)$ such that $v_i = 0$, $v[V_i] \not\succeq \overline{v}[V_i]$ and $\varphi(v') < \varphi(\overline{v'})$ for $v' = v + \sum_{j=i+1}^{n} e^{(j)}$, let $f := \sigma_{(v;V_i)}(f)$, where $V_i = \{1, 2, \ldots, i\}$.

Step 2: If $i = n$, output f and halt. Otherwise, $i := i + 1$ and return to Step 1.

Note that the set $\min_{\succeq} T(f)$ in the "while" statement of Step 1 is updated as a result of applying the σ transformation to f in Step 2.

Example 6.5. Let us consider the availability of the 6-variable functions, when $p_1 = 9/10$, $p_2 = 6/7$, $p_3 = 4/5$, $p_4 = 7/10$, and $p_5 = 3/5$. Recall that $\Phi(f) = A(f)$ is given by (25). We apply algorithm OPT-REG-SD to this $\Phi(f)$.

Step 0: Let $i := 1$ and $f := x_1$ (thus, $\min_{\succeq} T(f) = \{(10000)\}$). Let $u = (10000)$.

First iteration of Steps 1-2: Since $u_1 = 1$, skip Step 1. Step 2 sets $i := 2$.

Second iteration of Steps 1-2: Since $u[V_2] = (10) \succeq \overline{u}[V_2] = (01)$ for the only vector u in $\min_{\succeq} T(f)$, skip Step 1. Step 2 sets $i := 3$.

Third iteration of Steps 1-2: Vector $u \in \min_{\succeq} T(f)$ satisfies $u_3 = 0$, $u[V_3] = (100) \not\succeq \overline{u}[V_3] = (011)$, but $\varphi(10011) = 9/10 \times 1/7 \times 1/5 \times 7/10 \times 3/5 = 189/17500 > \varphi(01100) = 1/10 \times 6/7 \times 4/5 \times 3/10 \times 2/5 = 144/17500$. (If we were to apply $\sigma_{(u;V_3)}$ to f, we would have $\Phi(\sigma_{(u;V_3)}(f) < \Phi(f)$.) Thus skip Step 1. Step 2 sets $i := 4$.

Fourth iteration of Steps 1-2: Vector u satisfies $u_4 = 0$, $u[V_4] = (1000) \not\succeq \overline{u}[V_4] = (0111)$. Moreover, we have $\varphi(10001) < \varphi(01110)$, since $\varphi(10001) = 9/10 \times 1/7 \times 1/5 \times 3/10 \times 3/5 = 81/17500$ and $\varphi(01110) = 1/10 \times 6/7 \times 4/5 \times 7/10 \times$

$2/5 = 336/17500$. Thus f is transformed to

$$f := \sigma_{(u;V_4)}(f) = 12 + 13 + 14 + 234.$$

For this new f, we have $\min_{\succeq} T(f) = \{u = (10010), v = (01110)\}$. Since $u_4 = v_4 = 1$, we skip Step 1. Step 2 sets $i := 5$.

Fifth iteration of Steps 1-2: u satisfies $u_5 = 0$, $u[V_5] = (10010) \not\succeq \overline{u}[V_5] = (01101)$. Moreover, we have $\varphi(10010) < \varphi(01101)$, since $\varphi(10010) = 9/10 \times 1/7 \times 1/5 \times 7/10 \times 2/5 = 126/17500$ and $\varphi(01101) = 1/10 \times 6/7 \times 4/5 \times 3/10 \times 3/5 = 216/17500$. Thus f is transformed to

$$\begin{aligned} f := \sigma_{(u;V_5)}(f) &= 12 + 13 + 14(2 + 3 + 5) + 234 + 235 \\ &= 12 + 13 + 145 + 234 + 235. \end{aligned} \tag{26}$$

As a result, we have $\min_{\succeq} T(f) = \{u = (10100), v = (10011), w = (01101)\}$. Since $v_5 = w_5 = 1$, we need to consider only $u = (10100)$. Since $u_5 = 0$, $u[V_5] = (10100) \not\succeq \overline{u}[V_5] = (01011)$, but $\varphi(10100) = 216/17500 > \varphi(01011) = 126/17500$, skip Step 1. Since $i = n = 5$, output function f given by (26) and halt.

As mentioned in the introduction, the weights $(\log_2 9, \log_2 6, 2, \log_2(7/3), \log_2(3/2))$ defined by (1) produce an optimal vote-assignable coterie, since tie-breaking is unnecessary here. However, some weights are irrational, and hence we cannot exactly compute the sum of the weights $w^*(S) = \sum_{i \in S} w^*(i)$.

Let f_i, $i = 1, 2, \ldots, n$, be the function f after the ith iteration of Step 1 of algorithm OPT-REG-SD has been completed. Then clearly $V_{f_i} \subseteq V_i\,(= \{1, 2, \ldots, i\})$ holds. Moreover, we have the following lemma.

LEMMA 6.6. *Let f_i, $i = 1, 2, \ldots, n$, be as defined above. For each $i = 1, 2, \ldots, n$, all vectors $v \in \min_{\succeq} T(f_i)$ with $v[V_i] \not\succeq \overline{v}[V_i]$ satisfy*

$$\varphi(v') \geq \varphi(\overline{v'}), \tag{27}$$

where $v' = v + \sum_{j=i+1}^{n} e^{(j)}$.

Proof. If $i = 1$, then $f_1 = x_1$, and the lemma holds in this case, since $\min_{\succeq} T(f_1) = \{v = (100 \cdots 0)\}$ and $v[V_1] \succeq \overline{v}[V_1]$.

Assuming it holds for $i = k$, consider the case where $i = k + 1$. Let us consider the vector $v \in \min_{\succeq} T(f_{k+1})$ with $v[V_{k+1}] \not\succeq \overline{v}[V_{k+1}]$. If $v_{k+1} = 0$, then v satisfies $\varphi(v') \geq \varphi(\overline{v'})$, where $v' = v + \sum_{j=k+2}^{n} e^{(j)}$, since otherwise, v would have been removed from $T(f_{k+1})$ in Step 1 by applying the operation $\sigma_{(v;V_{k+1})}$ to f_{k+1}. This contradicts the definition of f_{k+1}.

If $v_{k+1} = 1$, on the other hand, there are two possibilities: (1) $f_k(v) = 0$ and (2) $f_k(v) = 1$. In case (1), since $v \in T(f_{k+1}) \setminus T(f_k)$, f was updated by $f := \sigma_{(\overline{v'};V_{k+1})}(f)$ in the $(k+1)$st iteration of Step 1. Therefore, we have $\varphi(v') \geq \varphi(\overline{v'})$.

In case (2), assuming $\varphi(v') < \varphi(\overline{v'})$, we derive a contradiction. $V_{f_k} \subseteq V_k$ implies $f_k(v - e^{(k+1)}) = 1$. Since $v[V_{k+1}] \not\succeq \overline{v}[V_{k+1}]$, we have $(v - e^{(k+1)})[V_k] \not\succeq (\overline{v - e^{(k+1)}})[V_k]$. Note that $(v - e^{(k+1)})' = v - e^{(k+1)} + \sum_{j=k+1}^{n} e^{(j)} = v'$. Thus, if $v - e^{(k+1)} \in \min_{\succeq} T(f_k)$, by assumption, f_k would have been updated by $f_k := \sigma_{(v-e^{(k+1)};V_k)}(f_k)$. This contradicts the definition of f_k. Now, since $v - e^{(k+1)} \notin \min_{\succeq} T(f_k)$, there exists a vector $w \in \min_{\succeq} T(f_k)$ such that $w \preceq v - e^{(k+1)}$. We can easily see that this w satisfies $w[V_k] \not\succeq \overline{w}[V_k]$ and $\varphi(w') < \varphi(\overline{w'})$, where $w' = w + \sum_{j=k+1}^{n} e^{(j)}$. This implies that f_k would have been updated by $f_k := \sigma_{(w;V_k)}(f_k)$, again a contradiction. ∎

LEMMA 6.7. *Let f_n be as defined above. Then f_n maximizes Φ among all self-dual functions.*

Proof. Lemma 6.6 asserts (when $i = n$) that all vectors $v \in \min_{\succeq} T(f_n)$ with $v \not\succeq \overline{v}$ satisfy $\varphi(v) \geq \varphi(\overline{v})$. As for vectors $v \in \min_{\succeq} T(f_n)$ with $v \succeq \overline{v}$, the g-regularity of φ implies $\varphi(v) \geq \varphi(\overline{v})$. Together with Lemma 6.1, this completes the proof. □

THEOREM 6.8. *Algorithm* OPT-REG-SD *correctly outputs a regular self-dual function f that maximizes Φ among all self-dual functions in $O(n^3|\min T(f)|)$ time.*

Proof. Since the algorithm's correctness follows from Lemma 6.7, we consider only its time complexity.

Let us assume that OPT-REG-SD generates the following sequence of functions: $f_m\,(= x_1) \to f_{m-1} \to \ldots \to f_0$, where f_0 is the output of algorithm OPT-REG-SD. Then there must be a sequence of transformations, $f_0 \to f_1 \to \ldots \to f_m$, which can be generated by algorithm TRANS-REG-SD. This means that $m \leq |\min T(f_0)|$.

As in algorithm GEN-REG-SD, we use binary trees B as the data structures of $\min T(f)$ and $\min_{\succeq} T(f)$ in the following way: For a vector v, define $\beta(v)$ by

$$\beta(v) = \min\left(\left\{ i \geq \max ON(v) \mid \varphi(v') < \varphi(\overline{v'}) \text{ for } v' = v + \sum_{j=i+1}^{n} e^{(j)} \right\} \cup \{n+1\}\right).$$

It is clear that $\varphi(v') < \varphi(\overline{v'})$ holds for $v' = v + \sum_{j=i+1}^{n} e^{(j)}$ with $i \geq \beta(v)$, if $\beta(v) \leq n$. Algorithm OPT-REG-SD then prepares $B(\min T(f))$ and $\{B(\min_{\succeq} T(f)_{(i,j)}) \mid i = 1, 2, \ldots, n+1, j = 1, 2, \ldots, n\}$, where

$$\min_{\succeq} T(f)_{(i,j)} = \{v \in \min_{\succeq} T(f) \mid i = \max\{\alpha(v), \beta(v)\},\ j = \max ON(v)\}.$$

As in the time complexity analysis of algorithm GEN-REG-SD, we can prove that each iteration of Step 1 in algorithm OPT-REG-SD can be executed in $O(n^3)$ time. Thus it requires $O(n^3|\min T(f)|)$ time in total. □

Acknowledgment. The authors thank the anonymous referees for their helpful and constructive comments which improved the presentation of this paper.

REFERENCES

[1] Y. AMIR AND A. WOOL, *Optimal availability quorum systems: Theory and practice*, Inform. Process. Lett., 65 (1998), pp. 223–228.

[2] D. AVIS AND K. FUKUDA, *A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra*, Discrete Comput. Geom., 8 (1992), pp. 295–313.

[3] D. AVIS AND K. FUKUDA, *Reverse search for enumeration*, Discrete Appl. Math., 65 (1996), pp. 21–46.

[4] D. BARBARA AND H. GARCIA-MOLINA, *The vulnerability of vote assignment*, ACM Trans. Computer Systems, 4 (1986), pp. 187–213.

[5] D. BARBARA AND H. GARCIA-MOLINA, *The reliability of voting mechanisms*, IEEE Trans. Comput., 36 (1987), pp. 1197–1208.

[6] P. BERTOLAZZI AND A. SASSANO, *An $O(mn)$ time algorithm for regular set-covering problems*, Theoret. Comput. Sci., 54 (1987), pp. 237–247.

[7] L.J. BILLERA, *On the composition and decomposition of clutters*, J. Combinatorial Theory Ser. B, 11 (1971), pp. 234–245.

[8] J.C. BIOCH AND T. IBARAKI, *Complexity of identification and dualization of positive Boolean functions*, Inform. Comput., 123 (1995), pp. 50–63.

[9] J.C. BIOCH AND T. IBARAKI, *Generating and approximating positive non-dominated coteries*, IEEE Trans. Parallel Distrib. Systems, 6 (1995), pp. 905–914.

[10] E. BOROS, P.L. HAMMER, T. IBARAKI, AND K. KAWAKAMI, *Polynomial-time recognition of 2-monotonic positive Boolean functions given by an oracle*, SIAM J. Comput., 26 (1997), pp. 93–109.

[11] V. Chvátal and P.L. Hammer, *Aggregation of inequalities in integer programming*, Ann. Discrete Math., 1 (1977), pp. 145–162.
[12] Y. Crama, *Dualization of regular Boolean functions*, Discrete Appl. Math., 16 (1987), pp. 79–85.
[13] S.B. Davidson, *Replicated data and partition failures*, in Distributed Systems, S. Mullender, ed., Addison-Wesley, Reading, MA, 1989.
[14] S.B. Davidson, H. Garcia-Molina, and D. Skeen, Consistency in partitioned networks, ACM Comput. Surveys, 17 (1985), pp. 341–370.
[15] K. Diks, E. Kranakis, K. Krizanc, B. Mans, and A. Pelc, *Optimal coteries and voting schemes*, Inform. Process. Lett., 51 (1994), pp. 1–6.
[16] T. Eiter and G. Gottlob, *Identifying the minimal transversals of a hypergraph and related problems*, SIAM J. Comput., 24 (1995), pp. 1278–1304.
[17] M.L. Fredman and L. Khachiyan, *On the complexity of dualization of monotone disjunctive normal forms*, J. Algorithms, 21 (1996), pp. 618–628.
[18] H. Garcia-Molina and D. Barbara, *How to assign votes in a distributed system*, J. ACM, 32 (1985), pp. 841–860.
[19] D.K. Gifford, *Weighted voting for replicated data*, in Proceedings of the 7th Symposium on Operating System Principles, Pacific Grove, CA, ACM, 1979, pp. 150–162.
[20] T. Ibaraki and T. Kameda, *A theory of coteries: Mutual exclusion in distributed systems*, IEEE Trans. Parallel Distrib. Systems, 4 (1993), pp. 779–794.
[21] T. Ibaraki, H. Nagamochi, and T. Kameda, *Optimal coteries for rings and related networks*, in Proceedings of the 12th International Conference on Distributed Computing Systems, Yokohama, Japan, 1992, pp. 650–656.
[22] D.S. Johnson, M. Yannakakis, and C.H. Papadimitriou, *On generating all maximal independent sets*, Inform. Process. Lett., 27 (1988), pp. 119–123.
[23] L. Lamport, *Time, clocks, and the ordering of events in a distributed system*, Comm. ACM, 21 (1978), pp. 558–565.
[24] K. Makino and T. Ibaraki, *The maximum latency and identification of positive Boolean functions*, SIAM J. Comput., 26 (1997), pp. 1363–1383.
[25] K. Makino and T. Ibaraki, *A fast and simple algorithm for identifying 2-monotonic positive Boolean functions*, J. Algorithms, 26 (1998), pp. 291–305.
[26] K. Makino and T. Kameda, *Efficient generation of all regular non-dominated coteries*, in Proceedings of the Nineteenth ACM Symposium on Principles of Distributed Computing (PODC 2000), Portland, OR, 2000, pp. 279–288.
[27] S.J. Mullender and P.M.B. Vitányi, *Distributed match-making*, Algorithmica, 3 (1988), pp. 367–391.
[28] S. Muroga, *Threshold Logic and Its Applications*, Wiley-Interscience, New York, 1971.
[29] M. Naor and A. Wool, *The load, capacity and availability of quorum systems*, SIAM J. Comput., 27 (1998), pp. 423–447.
[30] M. Naor and A. Wool, *Access control and signatures via quorum secret sharing*, IEEE Trans. Parallel Distrib. Systems, 9 (1998), pp. 909–922.
[31] U.N. Peled and B. Simeone, *Polynomial-time algorithm for regular set-covering and threshold synthesis*, Discrete Appl. Math., 12 (1985), pp. 57–69.
[32] U.N. Peled and B. Simeone, *An $O(nm)$-time algorithm for computing the dual of a regular Boolean function*, Discrete Appl. Math., 49 (1994), pp. 309–323.
[33] D. Peleg and A. Wool, *The availability of quorum systems*, Inform. and Comput., 123 (1995), pp. 210–223.
[34] J.S. Provan and M.O. Ball, *Efficient recognition of matroids and 2-monotonic systems*, in Applications of Discrete Mathematics, R. Ringeisen and F. Roberts, eds., SIAM, Philadelphia, 1988, pp. 122–134.
[35] K.G. Ramamurthy, *Coherent Structures and Simple Games*, Kluwer Academic Publishers, Dordrecht, the Netherlands, 1990.
[36] M. Spasojevic and P. Berman, *Voting as the optimal static pessimistic scheme for managing replicated data*, IEEE Trans. Parallel Distrib. Systems, 5 (1994), pp. 64–73.
[37] R.H. Thomas, *A majority consensus approach to concurrency control*, ACM Trans. Database Systems, 4 (1979), pp. 180–209.
[38] Z. Tong and R.Y. Kain, *Vote assignments in weighted voting mechanisms*, in Proceedings of the 7th Symposium on Reliable Distributed Systems, 1988, pp. 138–143.
[39] T.W. Yan and H. Garcia-Molina, *Distributed selective dissemination of information*, in Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems, 1994, pp. 89–98.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 408–422

ON APPROXIMATING THE ACHROMATIC NUMBER*

GUY KORTSARZ† AND ROBERT KRAUTHGAMER‡

Abstract. The achromatic number problem is to legally color the vertices of an input graph with the maximum number of colors, denoted ψ^*, so that every two color classes share at least one edge. This problem is known to be NP-hard.

For general graphs we give an algorithm that approximates the achromatic number within a ratio of $O(n \cdot \log\log n/\log n)$. This improves over the previously known approximation ratio of $O(n/\sqrt{\log n})$, due to Chaudhary and Vishwanathan [*Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, LA, 1997, pp. 558–563].

For graphs of girth at least 5 we give an algorithm with an approximation ratio $O(\min\{n^{1/3}, \sqrt{\psi^*}\})$. This improves over an approximation ratio $O(\sqrt{\psi^*}) = O(n^{3/8})$ for the more restricted case of graphs with girth at least 6, due to Krysta and Loryś [*Proceedings of the Seventh Annual European Symposium on Algorithms*, Lecture Notes in Comput. Sci. 1643, Springer-Verlag, Berlin, 1999, pp. 402–413].

We also give the first hardness result for approximating the achromatic number. We show that for every fixed $\epsilon > 0$ there is no $2 - \epsilon$ approximation algorithm, unless $P = NP$.

Key words. achromatic number, graph coloring, approximation algorithms

AMS subject classifications. 68Q25, 68W25, 05C15

PII. S0895480100379579

1. Introduction.

A *coloring* (a.k.a. *legal coloring*) of a graph $G(V, E)$ is an assignment of colors to the graph vertices such that whenever two vertices are adjacent, they are colored differently. A k-coloring is one that uses k colors. A coloring is called *complete* (or *achromatic*) if for every pair of distinct colors, there exist two adjacent vertices which are assigned these two colors. The *achromatic number* $\psi^*(G)$ of a graph G is the largest number k such that G has a complete k-coloring. (For known results on $\psi^*(G)$ see the surveys of Edwards [5] and Hughes and MacGillivray [13]).

Yannakakis and Gavril [17] proved that the problem of computing the achromatic number of a graph, called *the achromatic number problem*, is NP-hard. We are therefore interested in algorithms that find approximate solutions. An *approximation algorithm* with a ratio $\alpha \geq 1$ (called in short an approximation ratio α) for the achromatic number problem is an algorithm that runs in polynomial time and finds for an input graph G a complete coloring with at least $\psi^*(G)/\alpha$ colors. Throughout, let n denote the number of vertices in the input graph G, let m denote the number of edges in it, and let $\psi^* = \psi^*(G)$ be its achromatic number.

1.1. Previous work.

Yannakakis and Gavril [17] proved that the achromatic number problem is NP-hard. In [6], Farber et al. show that the problem is NP-hard on bipartite graphs. In [1], Bodlaender shows that the problem is NP-hard on graphs

*Received by the editors October 16, 2000; accepted for publication (in revised form) June 4, 2001; published electronically August 29, 2001. A preliminary version of this paper appeared in *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms*, Washington, D.C., 2001, pp. 309–318.

http://www.siam.org/journals/sidma/14-3/37957.html

†Department of Computer Science, Open University of Israel, 16 Klauzner St., Ramat-Aviv, Israel (guyk@oumail.openu.ac.il).

‡Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel (robi@wisdom.weizmann.ac.il). The research of this author was supported in part by a Dora Ostre memorial scholarship.

that are simultaneously cographs and interval graphs. In [3], Cairnie and Edwards show that the problem is NP-hard on trees.

Chaudhary and Vishwanathan [4] give the first sublinear approximation algorithm for the achromatic problem with an approximation ratio $O(n/\sqrt{\log n})$. They conjecture that it can be approximated within a much better ratio of $O(\sqrt{\psi^*})$ and give an algorithm with this approximation ratio of $O(\sqrt{\psi^*}) = O(n^{7/20})$ for graphs with *girth* (length of the shortest simple cycle) at least 7. For trees they give an algorithm with approximation ratio 7.

Krysta and Loryś [14] give an algorithm with an approximation ratio $O(\sqrt{\psi^*}) = O(n^{3/8})$ for graphs with girth at least 6, proving for these graphs the conjecture of [4]. For trees, they improve the approximation ratio to 2.

1.2. Related problems. An *independent set* in a graph is a subset of the vertices, every two of which are nonadjacent. The size of the largest independent set in G is denoted by $\alpha(G)$.

A coloring of a graph is a partition of the vertex set into color classes, each of which is an independent set. A possible "greedy" approach for obtaining a complete coloring is to iteratively remove from the graph maximal independent sets of small size. (Maximality here is with respect to containment.) However, the problem of finding a *minimum maximal independent set* cannot be approximated within a ratio $n^{1-\epsilon}$ for any $\epsilon > 0$, unless P=NP [8].

A related notion is the *chromatic number* $\chi(G)$ of a graph G, which is the smallest number k such that G has a (complete) k-coloring. Clearly, $\psi^*(G) \geq \chi(G)$. The chromatic number problem, i.e., computing $\chi(G)$, is also NP-hard.

In many respects, the achromatic number problem differs from the chromatic number problem. For example, when $\psi^*(G) = O(1)$, a complete coloring with $\psi^*(G)$ colors can be found in polynomial time, e.g., by guessing $\binom{\psi^*(G)}{2}$ "critical" edges (see [6] for a more efficient algorithm). In contrast, even when $\chi(G) = 3$, it is NP-hard to find a 3-coloring.

From the aspect of approximation, the chromatic number $\chi(G)$ appears to be understood relatively well: an algorithm with an approximation ratio of $O(n\frac{(\log\log n)^2}{(\log n)^3})$ is given in [9], and an $n^{1-\epsilon}$ hardness of approximation (for every fixed $\epsilon > 0$, assuming NP$\not\subseteq$ZPP) is shown in [7]. It is conjectured [4] that the achromatic number $\psi^*(G)$ can be approximated better than that of the chromatic number, namely, within ratio $O(\sqrt{\psi}^*)$.

1.3. Our results. Our results narrow, in three different ways, the (currently huge) gap in the approximation ratio of the achromatic number problem. We improve the known approximation ratio for general graphs (upper bound); we give the first hardness of approximation result (lower bound); and we extend the family of graphs for which the $O(\sqrt{\psi^*})$ approximation ratio (that is conjectured in [4]) is known to hold.

In section 2 we give an algorithm with an approximation ratio $O(n \log\log n/\log n)$ for general graphs, improving over the previous $O(n/\sqrt{\log n})$ ratio of [4]. The algorithm actually produces a complete $\Omega(\log \psi^*/\log\log \psi^*)$-coloring. We remark that a similar result is not known for some related problems such as the maximum independent set and the chromatic number. Namely, it is not known whether an independent set of size roughly $\log \alpha(G)$ can be found in polynomial time or whether G can be colored in polynomial time with $2^{O(\chi(G))}$ colors.

Our approximation algorithm for general graphs uses a result of Máté [16] for

finding a complete $\Omega(\log \psi^* / \log\log \psi^*)$-coloring in a restricted family of graphs called irreducible graphs. (See section 2 for the definition of irreducibility.) For certain bipartite graphs (those that can be significantly reduced in a sense that is described in section 2) we devise another algorithm whose approximation ratio is significantly better.

In section 3 we give an approximation algorithm with a ratio $O(\min\{\sqrt{\psi^*}, n^{1/3}\})$ for graphs of girth at least 5. In terms of ψ^*, our ratio of $O(\sqrt{\psi^*})$ proves the conjecture of [4] for the special case of graphs with girth at least 5, extending the previously known result of [14] for (the more restricted case of) graphs with girth at least 6. In terms of n, our ratio of $O(n^{1/3})$ for graphs with girth at least 5 improves over the previously known ratios for (the more restricted cases of) graphs with girth at least 6 and 7, namely, the $O(n^{3/8})$ ratio of [14] and the $O(n^{7/20})$ ratio of [4], respectively.

Our proof also gives a lower bound $\psi^* \geq m/n$ in graphs of girth at least 5 (recall that m is the number of edges in the graph), which may be of independent interest in the context of graph theory. This lower bound does not hold for graphs of girth 4, as demonstrated by a complete bipartite graph whose achromatic number is 2.

In section 4 we show that for any fixed $\epsilon > 0$, the achromatic number problem cannot be approximated within a ratio $2 - \epsilon$, unless $P = NP$. No hardness of approximation was previously known for this problem. In particular, our reduction shows that it is NP-hard to decide whether a graph has a complete coloring with all color classes of size exactly 2. In contrast, for a complete coloring with all color classes of size exactly 1, the graph has to be a complete graph (i.e., a clique), and so the corresponding decision problem is in P.

1.4. Preliminaries. For the algorithms, we may assume that $\psi^*(G)$ is known, e.g., by exhaustively searching over the n possible values or by using binary search. Throughout, an algorithm is considered efficient if it runs in polynomial time.

For a subset U of the vertices, let $G[U]$ be the subgraph of G induced on U. We say that a vertex v is *adjacent* to U if v is adjacent to at least one vertex in U. Otherwise, we say that v is *independent* of U. Two subsets of vertices U, W are said to be *adjacent* if they contain a pair of vertices $u \in U, w \in W$ which are adjacent in G. We also say that U, W *share* an edge in G. In a complete coloring, every two color classes are adjacent to each other. The *girth* of a graph is the length of its shortest simple cycle.

A *partial complete coloring* of G is a complete coloring of an induced subgraph $G[U]$, namely, a coloring of a subset of the vertices such that each color class is adjacent to every other color class. The next straightforward lemma is well known [16, 5, 4, 14].

LEMMA 1.1. *A partial complete coloring can be extended greedily to a complete coloring of the entire graph.*

Proof. Consider a yet uncolored vertex. It can be added to one of the independent sets unless it is adjacent to all color classes, in which case this vertex can form a new color class. □

Let $G \setminus v$ denote the graph resulting from removing a vertex v (and its incident edges) from G. The next two lemmas follow from the definition of the achromatic number.

LEMMA 1.2. $\psi^*(G) - 1 \leq \psi^*(G \setminus v) \leq \psi^*(G)$ *for any vertex* v *in the graph* G.

LEMMA 1.3. $\binom{\psi^*}{2} \leq m$ *and hence* $\psi^* \leq O(\sqrt{m})$.

Semi-independent matchings. A collection M of edges of a graph G is called a *matching* if no two edges in M have a common endpoint. A matching $M =$

$\{(x_1, y_1), \ldots, (x_k, y_k)\}$ is called *independent* if no edge of $G \setminus M$ connects two matched vertices, i.e., both $X = \{x_1, \ldots, x_k\}$ and $\{y_1, \ldots, y_k\}$ are independent sets, and for all $i \neq j$, it holds that x_i is not adjacent to y_j.

A matching $M = \{(x_1, y_1), \ldots, (x_k, y_k)\}$ is called *semi-independent* if both $X = \{x_1, \ldots, x_k\}$ and $\{y_1, \ldots, y_k\}$ are independent sets, and for all $j > i$, it holds that x_i is not adjacent to y_j. (Here we assume that the edges of M are ordered from 1 to k and that the endpoints of each edge are also ordered. Note that x_i may be adjacent to y_j for $j < i$, which makes the difference between an independent and a semi-independent matching.)

A semi-independent matching can be used to obtain a partial complete coloring, as demonstrated in the next lemma. This lemma is also used in [16], and a weaker version, based on an independent matching, is used in [4].

LEMMA 1.4. *Given a semi-independent matching of size* $\binom{t}{2}$ *in a graph, a partial complete* t*-coloring of the graph can be computed efficiently.*

Proof. First color $x_1, \ldots, x_{t-1}$ with colors $2, \ldots, t$, respectively, and color all their matched vertices $y_1, \ldots, y_{t-1}$ with color 1. Next color $x_t, \ldots, x_{2t-3}$ with colors $3, 4, \ldots, t$, respectively, and color all their matched vertices $y_t, \ldots, y_{2t-3}$ with color 2. Proceed similarly until for some i, x_i is colored with color t and its matched vertex y_i is colored with color $t-1$. This happens for $i = \binom{t}{2}$, as each pair of distinct colors appears in exactly one edge (x_j, y_j). The lemma follows. □

2. Algorithm for general graphs. Our main result in this section is an algorithm with an approximation ratio $O(n \cdot \log\log n / \log n)$ for the achromatic number problem on general graphs. For bipartite graphs we give another algorithm whose approximation ratio is better in certain cases.

2.1. The equivalence graph. Hell and Miller [11] define the following equivalence relation (called the reducing congruence) on the vertex set of a graph G (see also [5, 13]). Two vertices in G are *equivalent* if they have the same set of neighbors in the graph.

Let $r_1, \ldots, r_q$ denote the different equivalence classes of G, where q is the number of equivalence classes. Let $S(r_i)$ denote the vertices that belong to the class r_i. Each vertex in $S(r_i)$ is called a *copy* of r_i. Note that two equivalent vertices cannot be adjacent to each other in G, so $S(r_i)$ forms an independent set in G.

The *equivalence graph* (also called the *reduced graph*) of G, denoted Q, is a graph whose vertices are the equivalence classes r_i and whose edges connect two vertices r_i, r_j whenever a vertex of $S(r_i)$ is adjacent in G to a vertex in $S(r_j)$. Note that if r_i, r_j are adjacent in Q, the subgraph induced by G on $S(r_i) \cup S(r_j)$ is a complete bipartite graph. The equivalence graph can be used to obtain a complete coloring of G, as shown in the following lemma.

LEMMA 2.1. *A partial complete* t*-coloring of the equivalence graph* Q *implies a partial complete* t*-coloring of* G *(which can be computed efficiently). Hence,* $\psi^*(G) \geq \psi^*(Q)$.

Proof. The proof follows easily by replacing each vertex of Q with an arbitrary copy of it from G. □

2.2. Irreducible graphs. A graph is called *irreducible* if all its equivalence classes are singletons, i.e., consist of a single vertex.

Máté [16] shows that the achromatic number of an irreducible graph on n vertices is lower bounded by $\Omega(\log n / \log\log n)$. The proof of [16] essentially gives an efficient algorithm for finding such a coloring, as stated in the next theorem. For completeness,

we review the algorithm in the appendix. We remark that Erdős constructed a family of graphs that nearly match this $\Omega(\log n/\log\log n)$ lower bound, as their achromatic number is $O(\log n)$, see [16, section 2] for more details.

THEOREM 2.2 (Máté [16]). *There exists a polynomial time algorithm that computes for an irreducible graph on n vertices a partial complete $\Omega(\log n/\log\log n)$-coloring.*

COROLLARY 2.3. *There exists a polynomial time algorithm that computes for a graph with q equivalence classes a partial complete $\Omega(\log q/\log\log q)$-coloring.*

Proof. Note that the equivalence graph Q is always irreducible, and hence Theorem 2.2 yields an algorithm that finds a partial complete $\Omega(\log q/\log\log q)$-coloring of Q. By Lemma 2.1 this coloring implies a partial complete $\Omega(\log q/\log\log q)$-coloring of G. □

For $q \geq n^{\Omega(1)}$, this corollary clearly yields an approximation ratio $O(n \cdot \log\log n/\log n)$. To obtain such an approximation ratio for general graphs we next deal with the case that q is small. Note that in general, q might be as small as $\log_2 \psi^*$.

2.3. Algorithm for general graphs. The next theorem is the main result of this section.

THEOREM 2.4. *A complete $\Omega(\log\psi^*/\log\log\psi^*)$-coloring can be computed efficiently.*

To prove the theorem, it suffices to assume that $q < (\psi^*)^\alpha$ for a constant $0 < \alpha < 1$, as otherwise the theorem follows from Corollary 2.3 since $\log q/\log\log q = \Omega(\log\psi^*/\log\log\psi^*)$. (This assumption on q will be used only at the final step of the algorithm.)

The first step of the algorithm is to remove from G the vertices whose equivalence class is of size smaller than $\psi^*/2q$. Let $\hat{V}$ denote these vertices and then $|\hat{V}| \leq q \cdot (\psi^*/2q) = \psi^*/2$. Let $G' = G \setminus \hat{V}$ denote the resulting graph, let Q' be the equivalence graph of G', and let q' be the number of vertices in Q'.

It suffices to find a partial complete $\Omega(\log\psi^*/\log\log\psi^*)$-coloring of G', since G' is an induced subgraph of G. By considering G' instead of G, we lose only a constant factor in the achromatic number because by Lemma 1.2, $\psi^*(G') \geq \psi^*(G) - |\hat{V}| \geq \psi^*/2$. The advantage of G' is that all its equivalence classes are relatively large. Indeed, when $\hat{V}$ is removed from G, some equivalence classes of G are removed, and the rest either remain intact or merge with each other. Hence any equivalence class of G' is of size at least $\psi^*/2q$.

The next goal of the algorithm will be to find as many as possible independent sets of Q' (not necessarily disjoint; i.e., a vertex in Q' may belong to more than one independent set) such that there is an edge (in the graph Q') between every two of these sets. The reason is that we can later replace the vertices from Q' with their copies from G'. We can actually replace each of these vertices with a distinct copy from G' (and obtain a coloring of G'), since for each vertex of Q' there are sufficiently many copies in G'.

For a set A of vertices of Q', let $N_{Q'}(A)$ denote the neighbors of A in the equivalence graph Q', i.e., all the vertices of Q' that are adjacent to A in the graph Q'. Call two (not necessarily disjoint) subsets A, B of vertices in Q' *equivalent* if $N_{Q'}(A) = N_{Q'}(B)$.

The algorithm iteratively constructs a collection $\mathcal{S}_i$ of (not necessarily disjoint) independent sets in Q' with the property that no two sets in $\mathcal{S}_i$ are equivalent, i.e., $N_{Q'}(A) \neq N_{Q'}(B)$ for all $A \neq B \in \mathcal{S}_i$. The initial collection $\mathcal{S}_1$ consists of all the vertex subsets of size one in Q'; i.e., $\mathcal{S}_1$ consists of all the sets $\{u\}$ where u is a vertex

in Q'. Clearly, each set in $\mathcal{S}_1$ is an independent set and no two sets are equivalent. The next collection $\mathcal{S}_{i+1}$ is constructed from $\mathcal{S}_i$ as follows. Start with $\mathcal{S}_{i+1} = \mathcal{S}_i$ and go over all $A \in \mathcal{S}_i$ and all vertices u of Q' which are independent of A. For every combination of A and u, insert the set $A \cup \{u\}$ into $\mathcal{S}_{i+1}$, unless $\mathcal{S}_{i+1}$ already contains a set that is equivalent to it. It follows that no two sets in $\mathcal{S}_{i+1}$ are equivalent; i.e., $N_{Q'}(A) \neq N_{Q'}(B)$ for all $A \neq B \in S_{i+1}$.

We next claim that $\mathcal{S}_i$ contains an equivalent for every independent set in Q' whose size is at most i.

LEMMA 2.5. *For any independent set A in Q' with $|A| \leq i$, there exists a set $B \in \mathcal{S}_i$ such that $N_{Q'}(A) = N_{Q'}(B)$.*

Proof. The proof proceeds by induction on i. For $i = 1$ the lemma follows from the fact that $\mathcal{S}_1$ contains all the subsets of one vertex of Q'.

Assuming that the lemma holds for $i \geq 1$, we show that it holds for $i + 1$. Let A be an independent set of size $i + 1$ in Q'. Then $A = \hat{A} \cup \{u\}$ for some $\hat{A}$ and u with $|\hat{A}| = i$. By the induction hypothesis, there exists a set $B' \in \mathcal{S}_i$ which is equivalent to $\hat{A}$, i.e., $N_{Q'}(\hat{A}) = N_{Q'}(B')$. The set $B' \cup \{u\}$ is then equivalent to A because $N_{Q'}(B' \cup \{u\}) = N_{Q'}(\hat{A} \cup \{u\}) = N_{Q'}(A)$. Since $B' \in \mathcal{S}_i$ we conclude that $\mathcal{S}_{i+1}$ contains a set that is equivalent to $B' \cup \{u\}$ and thus to A, as desired. □

The algorithm iteratively computes the collection $\mathcal{S}_i$ for $i = 1, 2, \ldots$, until the first time that $|\mathcal{S}_i| \geq \psi^*/2$. The following lemma guarantees that this happens within q' iterations, where clearly $q' \leq n$.

LEMMA 2.6. $|\mathcal{S}_{q'}| \geq \psi^*(G') \geq \psi^*/2$.

Proof. An optimal complete coloring of G' uses at least $\psi^*/2$ colors. Each color class in this coloring is an independent set of G', and hence defines an independent set in Q'. No two of these independent sets of Q' are equivalent because their corresponding color classes in G' share at least one edge. Each of these independent sets is of size at most q' because there are q' vertices in the graph Q', and there is no need to take a vertex of Q' more than once. So by Lemma 2.5, $\mathcal{S}_{q'}$ contains an equivalent for each of these independent sets. It follows that $|\mathcal{S}_{q'}| \geq \psi^*(G') \geq \psi^*/2$. □

Given the collection $\mathcal{S}_i$ with $|\mathcal{S}_i| \geq \psi^*/2$, define the following graph $\mathcal{G}_i$ whose vertex set is $\mathcal{S}_i$ (i.e., each vertex in $\mathcal{G}_i$ is an independent set in Q'). Two vertices are joined by an edge in the graph $\mathcal{G}_i$ if the two corresponding independent sets share an edge in Q'. It is not difficult to see that the graph $\mathcal{G}_i$ is irreducible. Indeed, every two of its vertices correspond to two sets $A, B \in \mathcal{S}_i$ that are not equivalent in Q'. Thus, there exists a vertex u_j of Q' which is independent of A but adjacent to B or vice versa. Since the set $\{u_j\}$ (or a set equivalent to it) belongs to $\mathcal{S}_i$, it corresponds to a vertex of $\mathcal{G}_i$ which is adjacent to exactly one of A and B.

Finally, apply Theorem 2.2 on $\mathcal{G}_i$ and compute for it a partial complete $\Omega(\log|\mathcal{S}_i|/\log\log|\mathcal{S}_i|)$-coloring. Since $|\mathcal{S}_i| \geq \psi^*/2$, the number of colors is $\Omega(\log\psi^*/\log\log\psi^*)$. Each color class is an independent set in $\mathcal{G}_i$ and thus corresponds to an independent set of Q'. Every two color classes in $\mathcal{G}_i$ share an edge (in $\mathcal{G}_i$), and thus their corresponding independent sets in Q' share an edge (in Q'). Now replace each vertex of these $\Omega(\log\psi^*/\log\log\psi^*)$ independent sets of Q' with a distinct copy of it from G'. Recall that each vertex of Q' has at least $\psi^*/2q > (\psi^*)^{1-\alpha}/2 \gg \log\psi^*/\log\log\psi^*$ copies in G'. By replacing each vertex of Q' with a distinct copy of it from G' we therefore obtain a partial complete $\Omega(\log\psi^*/\log\log\psi^*)$-coloring of G', which concludes the proof of Theorem 2.4.

THEOREM 2.7. *The achromatic number can be approximated within a ratio of $O(n \cdot \log\log n/\log n)$.*

Proof. The proof follows from the $O(\psi^* \cdot \log\log\psi^* / \log\psi^*)$ approximation ratio of Theorem 2.4 since $\psi^* \le n$. □

2.4. Algorithm for bipartite graphs. We give an algorithm for bipartite graphs that is based on the algorithm for general graphs but does not use the result of Máté. This algorithm obtains an improved approximation ratio compared to Corollary 2.3 when q, the number of equivalence classes of G, is not too large.

THEOREM 2.8. *The achromatic number of a bipartite graph can be approximated within a ratio of* $O(\max\{q, \sqrt{\psi^*}\})$.

Proof. As in the algorithm of Theorem 2.4 for general graphs, we first find a collection $\mathcal{S}_i$ with $|\mathcal{S}_i| \ge \psi^*/2$. (Note that no restriction on q is needed for this part.) Instead of the final step, which constructs the graph $\mathcal{G}_i$ and applies on it Theorem 2.2, we exploit the bipartiteness of G as follows.

G is bipartite, and hence G' is bipartite. Q' is also bipartite, since each equivalence class of G' is a subset of vertices all of which are from the same side of G'.

Let W^1, W^2 denote the two sides of Q', and "project" $\mathcal{S}_i$ on each side W^j, as follows. Start with $\mathcal{T}^j = \emptyset$ and go over all $A \in \mathcal{S}_i$. For every such A, insert $A \cap W^j$ into $\mathcal{T}^j$, unless $\mathcal{T}^j$ already contains a set that is equivalent to it. It follows that no two sets in $\mathcal{T}^j$ are equivalent.

By definition, $\mathcal{S}_i$ is closed, up to equivalence, under (the operation of) taking nonempty subsets; i.e., if $\mathcal{S}_i$ contains a set A, then for every nonempty $B \subset A$, there is some set equivalent to B in $\mathcal{S}_i$. Therefore, every set in $\mathcal{S}_i$ is equivalent to either a set in $\mathcal{T}^1$, or a set in $\mathcal{T}^2$, or the union of a set from $\mathcal{T}^1$ and a set from $\mathcal{T}^2$. Since no two sets in $\mathcal{S}_i$ are equivalent, we conclude that $|\mathcal{S}_i| \le |\mathcal{T}^1| + |\mathcal{T}^2| + |\mathcal{T}^1| \cdot |\mathcal{T}^2|$, and hence either $|\mathcal{T}^1|$ or $|\mathcal{T}^2|$ is at least $\Omega(\sqrt{|\mathcal{S}_i|})$.

Let $\mathcal{A}$ be the larger of the two sets $\mathcal{T}^1, \mathcal{T}^2$, and thus $|\mathcal{A}| = \Omega(\sqrt{|\mathcal{S}_i|}) = \Omega(\sqrt{\psi^*})$. Without loss of generality, we assume $\mathcal{A} = \mathcal{T}^1$. Observe that every set in $\mathcal{A}$ is an independent set in Q'. We will now apply on $\mathcal{A}$ a divide-and-conquer technique so that every two sets in it will share an edge.

Similarly to the well-known Quicksort algorithm, choose a vertex $p \in W^2$ to be a *pivot* element and separate the collection $\mathcal{A}$ into nonempty $\mathcal{A}^+$ and $\mathcal{A}^-$, where $\mathcal{A}^+$ consists of the sets that are adjacent to p (in Q'), and $\mathcal{A}^-$ consists of the sets that are independent of p (in Q'). (We show below that there always exists such a pivot $p \in W^2$, which can be found using exhaustive search). Add p to every set $B \in \mathcal{A}^-$, i.e., those sets that are not adjacent to p. Possibly, a set B already contains p in which case the operation has no effect. Observe that now every set $B \in \mathcal{A}^-$ shares an edge with every set $C \in \mathcal{A}^+$ because $p \in B$ and C is adjacent to p. For each of $\mathcal{A}^+, \mathcal{A}^-$ apply the same procedure recursively, unless every two subsets in it already share an edge.

The resulting sets are independent sets, since a vertex p is added to a set B only if B is independent of p, and initially every set in $\mathcal{T}^1 \subseteq \mathcal{S}_i$ is an independent set. In addition, every two of the resulting sets share an edge, since either (i) the two sets are separated at some point in the recursion, and then the pivot guarantees that they share an edge; or (ii) the two sets are never separated in the recursion, and then the stopping condition for the recursion guarantees that they share an edge.

We now show that if a collection $\hat{\mathcal{A}}$ contains two sets $\hat{B}$ and $\hat{C}$ which share no edge, then there exists a pivot $p \in W^2$ that separates $\hat{\mathcal{A}}$ into two nonempty parts. In the beginning of the recursive process the two sets $\hat{B}$ and $\hat{C}$ were $B \in \mathcal{T}^1$ and $C \in \mathcal{T}^1$, respectively. The recursive process may add only vertices to a set, and so $B \subseteq \hat{B}$ and $C \subseteq \hat{C}$. Since B and C are two different sets in $\mathcal{T}^1$, they are not

equivalent (in Q'), and hence there is a vertex p (in Q') that is adjacent to exactly one of B, C. However, Q' is bipartite and both $B, C \in \mathcal{T}^1$ contain only vertices from W^1, so $p \in W^2$. We claim that p is also adjacent to exactly one of $\mathcal{B}, \mathcal{C}$ and is hence a pivot that separates $\hat{\mathcal{A}}$ into two nonempty parts. Indeed, the vertices of $\hat{B} \setminus B$ and of $\hat{C} \setminus C$ are added as pivots at some point in the recursion. They must be from W^2 and are thus independent of $p \in W^2$. We conclude p is adjacent to exactly one of $\hat{B}, \hat{C}$ as claimed.

We thus find $\Omega(\sqrt{\psi^*})$ independent sets in Q', every two of which share an edge. Now replace each vertex of these independent sets with a distinct copy of it from G'. Recall that each vertex of Q' has at least $\psi^*/2q$ copies in G', so we can obtain a partial complete $\min\{\psi^*/2q, \Omega(\sqrt{\psi^*})\}$-coloring of G', which concludes the proof of Theorem 2.8. ∎

3. Algorithms for graphs with girth at least 5. In this section we give an algorithm with an approximation ratio $O(\min\{n^{1/3}, \sqrt{\psi^*}\})$ for the achromatic number problem on graphs with girth at least 5. We note in passing that for every such graph, $\psi^* \geq m/n$.

A *star* is a graph in which one of its vertices, called the *head* of the star, is connected to all other vertices, called the leaves of the star. Usually, it is also required that the leaves of the star are not connected to each other, but for graphs of girth larger than 3, this follows immediately.

3.1. Partial complete m/n-coloring.

THEOREM 3.1. *For any graph of girth at least* 5*, a partial complete* m/n*-coloring can be computed in polynomial time, and, in particular,* $\psi^*(G) \geq m/n$.

Proof. Iteratively remove from the graph any vertex whose degree is smaller than m/n. The graph does not turn empty as less than m edges are removed. In the remaining graph G', all degrees are at least m/n.

Consider in G' an arbitrary vertex v, its neighbors $N(v)$, and the set of vertices at distance 2 from v, denoted $N_2(v)$. Let $N(v) = \{x_1, \ldots, x_k\}$ with $k \geq m/n$. Let R_i denote the star connecting x_i to its neighbors in $N_2(v)$. Note that the stars R_i are vertex disjoint, or otherwise a cycle of length 4 (or less) is formed. In particular, for any $i \neq j$ there is no edge between the head of R_i and a leaf of R_j. Since all degrees in G' are at least m/n, each R_i has at least $m/n - 1$ leaves.

We now use the stars $R_1, \ldots, R_{\lceil m/n \rceil}$ to obtain a partial complete $\lceil m/n \rceil$-coloring. (We ignore vertices outside these stars.) Iteratively, for $i = 1, \ldots, \lceil m/n \rceil$, color with color i the head of the star R_i and one leaf from each of the stars R_j with $i < j \leq \lceil m/n \rceil$ so that a total of $\lceil m/n \rceil - i + 1$ vertices are colored with color i. The one leaf from each star R_j is chosen by going iteratively over $j = i+1, i+2, \ldots, \lceil m/n \rceil$, each time choosing (and coloring with color i) an arbitrary leaf of R_j which was not yet colored and is independent of all the vertices that were previously colored with color i. To see that such a leaf of R_j always exists, observe that (i) R_j contains at least $(\lceil m/n \rceil - 1) - (i-1)$ uncolored leaves; (ii) the head of R_i is independent of all the leaves of R_j; and (iii) a leaf of $R_{j'}$ for $i < j' < j$ is adjacent to at most one leaf of R_j, or otherwise a cycle of length 4 is formed. Hence, at least $(\lceil m/n \rceil - 1) - (i-1) - (j-i-1) = \lceil m/n \rceil - j + 1 \geq 1$ leaves of R_j can be chosen.

It follows that the vertices that are colored with each color i form an independent set. Moreover, each color class i contains a leaf from R_j for $i < j \leq \lceil m/n \rceil$. This leaf is adjacent to the head of R_j, which is colored by j, and thus color class i is adjacent to every color class $j > i$. Hence, this coloring is a partial complete $\lceil m/n \rceil$-coloring.

By Lemma 1.1 this coloring can be extended to a complete coloring of the entire graph G. □

Theorem 3.1 implies a relatively simple $O(\sqrt{n})$ approximation ratio. We further improve this ratio in the next subsections.

THEOREM 3.2. *There is an approximation algorithm with a ratio $O(\sqrt{n})$ for the achromatic number problem on graphs with girth at least 5.*

Proof. $\psi^* \le O(\sqrt{m})$ by Lemma 1.3. If $m < n$, then $\psi^* = O(\sqrt{n})$, and any greedy complete coloring will have at least one color, and hence a ratio $O(\sqrt{n})$. If $m \ge n$, the algorithm of Theorem 3.1 finds a complete coloring with at least m/n colors, and its ratio is thus $O(n/\sqrt{m}) = O(\sqrt{n})$. □

3.2. An $O(\sqrt{\psi^*})$ approximation algorithm.

THEOREM 3.3. *For every graph of girth at least 5, a partial complete $\Omega(\sqrt{\psi^*})$-coloring can be computed in polynomial time, and hence the achromatic number can be approximated within a ratio of $O(\sqrt{\psi^*})$.*

Proof. In what follows, we describe the algorithm together with its analysis. The algorithm is also depicted more schematically in Figure 3.1.

Algorithm Girth.

(1) Let $\rho \leftarrow \lfloor \sqrt{\psi^*} \rfloor$.
(2) $V_h \leftarrow \{v \in V : deg(v) \ge \rho\}$.
(3) If $|V_h| \ge \rho$,
    then return partial complete ρ-coloring of $V_h \cup N(V_h)$ using Lemma 3.4.
(4) $G'' \leftarrow G \setminus V_h, \mathcal{R} \leftarrow \emptyset, r \leftarrow 0$.
(5) While G'' contains edges,
    (a) pick a nonisolated vertex u in G'';
    (b) remove from G'' the vertex u and all its neighbors, and insert them into $\mathcal{R}$.
    (c) $r \leftarrow r + 1$.
(6) If $|\mathcal{R}| \ge \rho^3/16$,
    then return a partial complete $\Omega(\sqrt{r})$-coloring of $G[\mathcal{R}]$ using Lemma 3.5.
(7) Return a partial complete $m_{\mathcal{R}}/n_{\mathcal{R}}$-coloring of $G[\mathcal{R}]$ using Theorem 3.1, where $m_{\mathcal{R}}$ and $n_{\mathcal{R}}$ are the number of edges and vertices in $G[\mathcal{R}]$.

FIG. 3.1. *Algorithm for graphs of girth at least 5.*

Let $\rho = \lfloor \sqrt{\psi^*} \rfloor$. (Recall we assumed that ψ^* is known.) Let V_h be the set of vertices whose degree in G is at least ρ, and let $N(V_h)$ be the set of the neighbors of V_h in G. If V_h contains at least ρ vertices, then we use Lemma 3.4 to compute a partial complete ρ-coloring of (the subgraph induced on) $V_h \cup N(V_h)$, and the desired $\Omega(\sqrt{\psi^*})$-coloring follows.

So from now on we assume that $|V_h| \le \rho - 1$ and let $G' \leftarrow G \setminus V_h$. By Lemma 1.2 we know that $\psi^*(G') \ge \psi^* - \rho + 1 \ge \psi^*/2 + 1$. G' is an induced subgraph of G, so it suffices to compute a partial complete $\Omega(\sqrt{\psi^*})$-coloring of G'.

We partition the vertices of G' into two disjoint sets $\mathcal{R}$ and $\mathcal{I}$ as follows. Initially, let $G'' \leftarrow G', \mathcal{R} \leftarrow \emptyset$, and while G'' contains edges, iteratively perform the following 2 operations: (a) pick a nonisolated vertex u in G''; (b) remove from G'' the vertex u and all its neighbors and insert them into $\mathcal{R}$. Note that when removing this star, some vertices of G'' which are not removed may become isolated vertices and remain in G'' in all the iterations. Let $\mathcal{I}$ be the (possibly empty) set of vertices that remain in

G'' at the end of this process (i.e., when G'' has no edges). If $\mathcal{I}$ is not empty, then it is an independent set in G. Indeed, G'' is obtained from G by operations of removing vertices, so G and G'' have exactly the same edges inside $\mathcal{I}$.

Let r be the number of iterations in the above process. Note that in each iteration, we insert into $\mathcal{R}$ a star (consisting of a vertex u and its neighbors in the corresponding iteration), and hence $\mathcal{R}$ can be partitioned to r disjoint subsets, each corresponding to a star in G.

Consider the case that $|\mathcal{R}| \geq \rho^3/16$. Since all vertices in G' have degree less than ρ (in G and hence also in G'), each of the r stars (of $\mathcal{R}$) is of size at most ρ, and hence $r \geq |\mathcal{R}|/\rho \geq \rho^2/16$. Each of these r stars contains at least one leaf. We can thus use Lemma 3.5 to compute a partial complete $\Omega(\sqrt{r})$-coloring of $G[\mathcal{R}]$, the subgraph of G that is induced on these r stars. Since $\Omega(\sqrt{r}) = \Omega(\rho) = \Omega(\sqrt{\psi^*})$, this coloring is as desired.

In the case that $|\mathcal{R}| < \rho^3/16$, we apply the algorithm of Theorem 3.1 on the graph $G[\mathcal{R}] = G' \setminus \mathcal{I}$. Let $n_{\mathcal{R}}$ and $m_{\mathcal{R}}$ be the number of vertices and edges, respectively, in $G[\mathcal{R}]$. Clearly, $n_{\mathcal{R}} = |\mathcal{R}| < \rho^3/16 = O((\psi^*)^{3/2})$, so let us give a lower bound on $m_{\mathcal{R}}$. The number of edges in G' is at least $\binom{\psi^*/2+1}{2} \geq (\psi^*)^2/8$ by Lemma 1.3. Since $\mathcal{I}$ is an independent set also in G', all the edges in G' which are adjacent to $\mathcal{I}$ are also adjacent to $\mathcal{R}$. By the upper bound ρ on the degrees in G', we get that the number of these edges is at most $|\mathcal{R}| \cdot \rho \leq \rho^4/16 \leq (\psi^*)^2/16$. It follows that the number of edges in $G' \setminus \mathcal{I}$ is $m_{\mathcal{R}} \geq {\psi^*}^2/8 - {\psi^*}^2/16 = \Omega({\psi^*}^2)$. Hence, $m_{\mathcal{R}}/n_{\mathcal{R}} = \Omega(\sqrt{\psi^*})$, and Theorem 3.1 produces a partial complete $\Omega(\sqrt{\psi^*})$-coloring. □

To finish the proof of Theorem 3.3, we need to prove Lemmas 3.4 and 3.5.

LEMMA 3.4. *If $|V_h| \geq \rho$, a partial complete ρ-coloring of (the subgraph of G induced on) $V_h \cup N(V_h)$ can be computed in polynomial time.*

Proof. Consider the stars that each vertex of V_h defines (together with its neighbors) in G, denoted $Q_1, \ldots, Q_{|V_h|}$. Each star Q_i contains at least ρ leaves, since the degree of each vertex of V_h is at least ρ. Note that the stars $Q_1, \ldots, Q_k$ need not be disjoint. However, the heads of the stars are distinct, since they are distinct vertices in V_h.

We now proceed with coloring the stars $Q_1, \ldots, Q_\rho$ similarly to the coloring of the stars $R_1, \ldots, R_{m/n}$ in the proof of Theorem 3.1. Iteratively, for $i = 1, \ldots, \rho$, color with the ith color the head of the star Q_i and one leaf from each of the stars Q_j with $i < j \leq \rho$, so a total of $\rho - i + 1$ vertices are colored with color i. The $\rho - i$ leaves of the stars Q_j for $i < j \leq \rho$ are chosen iteratively for $j = i+1, i+2, \ldots, \rho$, each iteration coloring with color i a leaf of Q_j which was not colored yet and is independent of all the vertices that were previously colored with color i. To see that such a leaf of Q_j always exists, observe that (i) Q_j contains at least $\rho - (i-1)$ uncolored leaves; (ii) each vertex which was previously colored with color i is adjacent to at most one leaf of Q_j, or otherwise a cycle of length 4 is formed. Since the number of vertices which were previously colored with color i is $j - i$, at least $\rho - (i-1) - (j-i) = \rho - j + 1 \geq 1$ leaves of Q_j can be chosen.

It follows that the vertices that are colored with each color i form an independent set. Moreover, each color class i contains a leaf from Q_j for $i < j \leq \rho$. This leaf is adjacent to the head of Q_j, which is colored by j, and thus the color class i is adjacent to every color class $j > i$. Hence, this coloring is a partial complete ρ-coloring. □

LEMMA 3.5. *If $r \geq \rho^2/16$, then a partial complete $\Omega(\sqrt{r})$-coloring of the subgraph of G induced on the r stars of $\mathcal{R}$ can be computed in polynomial time.*

Proof. Let $x_1, \ldots, x_r$ be the heads of the r stars and choose arbitrarily one edge

(x_i, y_i) from each star. This gives a matching with r edges. (Observe that each star contains at least one leaf, since the vertex u chosen is not an isolated vertex.)

We now proceed with a partial $\sqrt{r}/8$-coloring for the vertices of the matching $(x_1, y_1), \ldots, (x_r, y_r)$ as follows. (In a sense, this extends the coloring from Lemma 1.4.) Iteratively, for $j = 1, 2, \ldots, \sqrt{r}/8$, perform the following four operations: (a) find *candidates* to be colored by color j, which are the vertices y_i that are yet uncolored and are independent of the vertices that were already colored by j; (b) find in the induced subgraph (of G) on these candidates y_i an independent set I_j of size $\sqrt{r}/8 - j$; (c) color all the vertices of I_j with color j; (d) color each of the vertices x_i which are matched to I_j with a distinct color from the set $\{j+1, j+2, \ldots, \sqrt{r}/8\}$.

To see that operation (b) is always feasible, consider an iteration j. Less than $r/64$ of the y_i vertices are colored because each previous iteration colors less than $\sqrt{r}/8$ of them. In addition, $j - 1 \leq \sqrt{r}/8$ vertices were already colored with j (these are x_i vertices), and each of them has less than $\rho \leq 4\sqrt{r}$ neighbors in G, so the number of y_i candidates to be colored by color j is at least $r - r/64 - 4\sqrt{r} \cdot \sqrt{r}/8 > r/3$. By Lemma 3.6 we know that one can efficiently find an independent set of size $\sqrt{r/3}/3 \geq \sqrt{r}/8$ in the subgraph induced on these candidates. □

LEMMA 3.6. *Let G be a graph of girth 5 on n vertices. Then an independent set of G of size $\sqrt{n}/3$ can be found in polynomial time.*

Proof. The average degree $\bar{d}$ is at most $2\sqrt{n}$ as a graph with girth 5 has at most $n\sqrt{n}$ edges; cf. [2]. Turan's theorem implies that G has an independent set of size $n/(\bar{d}+1) \geq \sqrt{n}/3$. Furthermore, a simple greedy algorithm finds such an independent set; see, for example, [10]. □

3.3. An $O(n^{1/3})$ approximation algorithm. In terms of n we have the following ratio.

THEOREM 3.7. *The achromatic number problem can be approximated within a ratio of $O(n^{1/3})$ in graphs with girth at least 5.*

Proof. If $m > n^{4/3}$, then Theorem 3.1 gives a ratio of $O(n/\sqrt{m}) = O(n^{1/3})$ as desired. If $m \leq n^{4/3}$, then $\psi^* \leq O(\sqrt{m}) = O(n^{2/3})$ and the required ratio follows from Theorem 3.3. □

4. Hardness of approximation. In this section we prove NP-hardness for the problem of approximating the achromatic number within a ratio of $2 - \epsilon$. The same reduction shows that it is NP-hard to decide whether a graph has a complete coloring with all color classes of size exactly 2 (or, alternatively, at most 3).

Our reduction uses a graphs operation called the (disjoint) union of graphs. We remark that Hell and Miller [12] give a tight analysis of the effect of this operation on the achromatic number from an extremal point of view.

THEOREM 4.1. *For every fixed $\epsilon > 0$, it is NP-hard to approximate the achromatic number within a ratio of $2 - \epsilon$.*

Proof. Our starting point is a reduction that creates a gap in the chromatic number $\chi(G)$ in the following way. Given an input x for an NP-complete language L, one can efficiently produce a graph $G(V, E)$ which satisfies (let $n = |V|$).

(1) If $x \in L$, then for a certain $\hat{\chi}$, the graph G can be (legally) colored with $\hat{\chi}$ colors, i.e., $\chi(G) \leq \hat{\chi}$, so that all color classes will be of equal size $n/\hat{\chi}$.
(2) If $x \notin L$, then every independent set in G is of size at most $\hat{\alpha}$, i.e., $\alpha(G) \leq \hat{\alpha}$, for a certain $\hat{\alpha} < n/\hat{\chi}$. (Hence $\chi(G) \geq n/\hat{\alpha}$.)

This reduction creates a gap of $n/\hat{\alpha}\hat{\chi} > 1$ in the chromatic number of G. Lund and Yannakakis [15] have shown such reductions (see also [7]). The gap that is created

in these reductions is an arbitrarily large constant. (We remark that for every fixed $\delta > 0$, a larger gap of $n^{1-\delta}$ can be obtained using randomized reductions.)

LEMMA 4.2. *For every fixed $\epsilon > 0$, there exists a reduction as above with gap $n/\hat{\alpha}\hat{\chi} \geq 1/\epsilon$.*

To produce a gap for the achromatic number problem, start with the graph G from the reduction of Lemma 4.2 and construct from it a graph H on $n(\hat{\chi}+1)$ vertices as follows. H is the union of $\bar{G}$, the edge complement of G, and a complete $\hat{\chi}$-partite graph on the vertex set W which consists of $\hat{\chi}$ parts $W_1, \ldots, W_{\hat{\chi}}$, each of size $n/\hat{\chi}$. In other words, the vertex set of H is $V \cup W_1 \cup \cdots \cup W_{\hat{\chi}}$; its edges inside V form the graph $\bar{G}$; a vertex from a set W_i is connected by an edge to a vertex of a set W_j if and only if $i \neq j$; and there are no other edges (e.g., between V and W).

We first show that if $x \in L$, then $\psi^*(H) \geq n$. If $x \in L$, then there is a coloring of G with $\hat{\chi}$ colors so that all color classes are of equal size $n/\hat{\chi}$. Let V_i denote the ith color class in this coloring for $i = 1, \ldots, \hat{\chi}$. Each color class V_i is an independent set in G and thus a clique in $\bar{G}$ and in H. Let us pair each vertex of V_i with a distinct vertex of W_i (observe that $|V_i| = n/\hat{\chi} = |W_i|$), so there would be exactly n pairs, one for each vertex of V.

We claim that considering each of the n pairs as a distinct color class gives a complete n-coloring of H, and hence $\psi^*(H) \geq n$. Indeed, each pair is an independent set because it contains one vertex from each of V, W and there are no edges between these two sets. To see that there is an edge between every two pairs, consider two arbitrary pairs. Suppose that one pair has a vertex from V_i and a vertex from W_i, and the other pair has a vertex from V_j and a vertex from W_j. If $i = j$, then the two vertices from V_i are connected because V_i forms a clique in $\bar{G}$ and thus in H. If $i \neq j$, then the vertex from W_i and the vertex from W_j are connected. Each of the $2n$ vertices of H belongs to some pair, and hence the n pairs form a complete n-coloring of H as claimed.

Now consider the case that $x \notin L$. Let $l = \psi^*(H)$, and let $C_1, \ldots, C_l$ be the corresponding color classes. Observe that a color class C_k is an independent set and thus contains vertices from at most one set W_j. So every class can be identified by a particular W_j from which it contains vertices, or it may contain no vertices from $W = \cup_j W_j$.

First consider the classes C_k which contain no vertices from V, and let l_0 denote the number of such classes. By the above observation, each of these l_0 classes is entirely contained in one set W_j. Furthermore, each of these l_0 color classes is contained in a different W_j, since each W_j is an independent set, and every two color classes share an edge. Therefore, $l_0 \leq \hat{\chi}$.

Next consider the color classes C_k which contain exactly one vertex from V and possibly some vertices from W, and let l_1 denote the number of these classes. We group these classes according to the observation above as follows. Let I_j denote the classes which contain exactly one vertex from V and one or more vertices from W_j, and let I_0 denote the classes which consist of a single vertex of V and no vertices of W. Then $l_1 = |I_0 \cup I_1 \cup \cdots I_{\hat{\chi}}|$.

The classes in I_j, for $j \geq 1$, cannot be connected to each other using their W_j vertices because W_j is an independent set. Since these color classes share an edge, they must be connected using their vertices from V. However, each of these classes has exactly one vertex from V, so these vertices from V form a clique in $\bar{G}$. A clique in $\bar{G}$ is of size at most $\hat{\alpha}$ (since $x \notin L$), and hence $|I_j| \leq \hat{\alpha}$ for every $j \geq 1$. A similar argument applies for I_0 and also for $I_0 \cup I_1$ (i.e., their corresponding vertices in V

must form a clique), and hence also $|I_0 \cup I_1| \le \hat{\alpha}$. We conclude that

$$l_1 = |I_0 \cup I_1 \cup \cdots \cup I_{\hat{\chi}}| \le \hat{\chi} \cdot \hat{\alpha}.$$

Finally, consider the remaining color classes C_k, i.e., those that contain two or more vertices from V and possibly some vertices from W, and let l_2 denote the number of such classes. These l_2 classes contain together the remaining $n - l_1$ vertices of V, and hence the number of such classes is $l_2 \le (n - l_1)/2$.

The total number of classes C_k in a complete coloring is thus

$$l = l_0 + l_1 + l_2 \le l_0 + l_1 + \frac{n - l_1}{2} \le \hat{\chi} + \frac{n + \hat{\chi} \cdot \hat{\alpha}}{2}.$$

Since $\hat{\alpha} \ge 1$, we can have in Lemma 4.2 that $\hat{\chi} \le \hat{\chi} \cdot \hat{\alpha} \le \epsilon n$, and hence

$$\psi^*(H) = l \le \frac{n(1 + 3\epsilon)}{2}.$$

We conclude that for any fixed $\epsilon > 0$ there is a gap of $2/(1 + 3\epsilon)$ between the achromatic number of H in the cases $x \in L$ and $x \notin L$. Since $\epsilon > 0$ is arbitrarily small, we can obtain any gap of $2 - \epsilon$ as desired. □

THEOREM 4.3. *It is NP-hard to decide whether an input graph has a complete coloring with all color classes of size exactly* 2 *or whether every complete coloring of the graph has a color class of size at least* 4.

Proof. Consider the reduction from the proof of Theorem 4.1. If $x \in L$, then there is a complete coloring of H with every color class of size exactly 2. If $x \notin L$, then a complete coloring of H can use at most $n(1 + 3\epsilon)/2$ colors; since H has $2n$ vertices, there must be a color class of size at least $\frac{2n}{n(1+3\epsilon)/2} > 3$. □

Appendix. An algorithm for irreducible graphs.

We describe below an efficient algorithm for finding a partial complete $\Omega(\log n / \log\log n)$-coloring of an irreducible graph G. The algorithm follows from the results of Máté [16].

1. Find greedily a (legal) coloring of G by iteratively removing from the graph a maximal independent set. Let $I_1, \ldots, I_p$ be the color classes, i.e., the independent sets that are removed in the iterations.
2. If the greedy coloring uses more than $\log n$ colors, then *return* this coloring. (Note that the greedy coloring is complete.)
3. Assume without loss of generality that $I = I_1$ is the largest of the independent sets. (Note that $|I| \ge n/\log n$.)
4. Fix an arbitrary total order $\prec$ on the vertices of I.
5. For each unordered pair $x, y \in I$ fix a *distinguisher* $d_{x,y}$, which is a vertex that is adjacent to exactly one of x, y. Let $f(\{x, y\})$ be the color of $d_{x,y}$ in the greedy coloring of step 1. (Note that $d_{x,y} \notin I$ and thus $f(x, y) \in \{2, \ldots, k\}$.)
6. For each unordered triple $x, y, z \in I$ let $g(\{x, y, z\})$ be the following boolean value. Assuming that $x \prec y \prec z$, let $g(\{x, y, z\})$ be 0 if $d_{x,y}$ and z are adjacent in G, and 1 otherwise.
7. Iteratively construct $I = S_0 \supset S_1 \supset \cdots \supset S_l$, and a set $Z = \{z_0, \ldots, z_{2l-1}\}$, as follows. Start with $S_0 \leftarrow I$ and let $M \leftarrow \log^{-2} |I|$. While $|S_i| \ge 50M^{-3}$, construct S_{i+1}:
   (a) let $z_{2i} \leftarrow \min_{\prec} S_i$ and $z_{2i+1} \leftarrow \min_{\prec} S_i \setminus \{z_{2i}\}$;
   (b) partition $S_i \setminus \{z_{2i}, z_{2i+1}\}$ into S^0_{i+1} and S^1_{i+1}, where S^r_{i+1} consists of the vertices $x \in S_i \setminus \{z_{2i}, z_{2i+1}\}$ with $g(\{z_{2i}, z_{2i+1}, x\}) = r$;

(c) let S_{i+1} be one of S^0_{i+1} and S^1_{i+1}, as follows:
if $|S^0_{i+1}| > Me^{-M/2}|S_i|$, then $S_{i+1} \leftarrow S^0_{i+1}$;
otherwise (it must hold that $|S^1_{i+1}| > e^{-M}|S_i|$) let $S_i \leftarrow S^1_{i+1}$.
(Note that $z_0 \prec z_1 \prec \ldots \prec z_{2l-1}$.)

8. For $r = 0, 1$,
   (a) let Q_r be the set of iterations i in which $S_{i+1} \leftarrow S^r_{i+1}$ is taken in (7c);
   (b) let $U_r \leftarrow \{(z_{2i}, z_{2i+1}) : i \in Q_r\}$.
9. If $|Q_0| \geq \Omega(\log |I| / \log\log |I|)$,
   return the following partial complete $|Q_0|$-coloring. For each pair of vertices $(z_{2i}, z_{2i+1}) \in U_0$ color with a fresh color its distinguisher $d_{z_{2i},z_{2i+1}}$ and one of the pair z_{2i}, z_{2i+1} which is not adjacent to the distinguisher $d_{z_{2i},z_{2i+1}}$.
10. Otherwise (it must hold that $|Q_1| \geq \Omega(\log^3 |I|)$).
    Find the f value that is most frequent among the pairs $z_{2i}, z_{2i+1} \in U_1$, and let U'_1 consist of the pairs with this f value. (Since f accepts at most $\log n$ values, $|U'_1| \geq |U_1| / \log n = \Omega(\log^2 n)$.)
11. Select from each pair of vertices $(z_{2i}, z_{2i+1}) \in U'_1$ the one which is adjacent to its distinguisher $d_{z_{2i},z_{2i+1}}$. These selected vertices form a matching to their corresponding distinguishers $d_{z_{2i},z_{2i+1}}$. This matching of size $\Omega(\log^2 n)$ can be shown to be semi-independent. We can thus use Lemma 1.4 to *return* a partial complete $\Omega(\log n)$-coloring of G.

Acknowledgment. The second author thanks Michael Langberg for helpful discussions.

REFERENCES

[1] H. L. Bodlaender, *Achromatic number is NP-complete for cographs and interval graphs*, Inform. Process. Lett., 31 (1989), pp. 135–138.
[2] B. Bollobás, *Extremal Graph Theory*, Academic Press, London, 1978.
[3] N. Cairnie and K. Edwards, *Some results on the achromatic number*, J. Graph Theory, 26 (1997), pp. 129–136.
[4] A. Chaudhary and S. Vishwanathan, *Approximation algorithms for the achromatic number*, in Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, 1997, pp. 558–563.
[5] K. Edwards, *The harmonious chromatic number and the achromatic number*, in Surveys in Combinatorics, Cambridge University Press, Cambridge, UK, 1997, pp. 13–47.
[6] M. Farber, G. Hahn, P. Hell, and D. Miller, *Concerning the achromatic number of graphs*, J. Combin. Theory Ser. B, 40 (1986), pp. 21–39.
[7] U. Feige and J. Kilian, *Zero knowledge and the chromatic number*, J. Comput. System Sci., 57 (1998), pp. 187–199.
[8] M. M. Halldórsson, *Approximating the minimum maximal independence number*, Inform. Process. Lett., 46 (1993), pp. 169–172.
[9] M. M. Halldórsson, *A still better performance guarantee for approximate graph coloring*, Inform. Process. Lett., 45 (1993), pp. 19–23.
[10] M. M. Halldórsson and J. Radhakrishnan, *Greed is good: Approximating independent sets in sparse and bounded-degree graphs*, Algorithmica, 18 (1997), pp. 145–163.
[11] P. Hell and D. J. Miller, *On forbidden quotients and the achromatic number*, Congr. Numer., 15 (1976), pp. 283–292.
[12] P. Hell and D. J. Miller, *Achromatic numbers and graph operations*, Discrete Math., 108 (1992), pp. 297–305.
[13] F. Hughes and G. MacGillivray, *The achromatic number of graphs: A survey and some new results*, Bull. Inst. Combin. Appl., 19 (1997), pp. 27–56.
[14] P. Krysta and K. Loryś, *Efficient approximation algorithms for the achromatic number*, in Proceedings of the Seventh Annual European Symposium on Algorithms, Lecture Notes in Comput. Sci. 1643, Springer-Verlag, Berlin, 1999, pp. 402–413.

[15] C. Lund and M. Yannakakis, *On the hardness of approximating minimization problems*, J. Assoc. Comput. Mach., 41 (1994), pp. 960–981.
[16] A. Máté, *A lower estimate for the achromatic number of irreducible graphs*, Discrete Math., 33 (1981), pp. 171–183.
[17] M. Yannakakis and F. Gavril, *Edge dominating sets in graphs*, SIAM J. Appl. Math., 38 (1980), pp. 364–372.

SIAM J. DISCRETE MATH.
Vol. 14, No. 3, pp. 423–431

A HYPERGRAPH APPROACH TO THE IDENTIFYING PARENT PROPERTY: THE CASE OF MULTIPLE PARENTS*

ALEXANDER BARG†, GÉRARD COHEN‡, SYLVIA ENCHEVA§, GREGORY KABATIANSKY¶, AND GILLES ZÉMOR‖

Abstract. Let C be a code of length n over an alphabet of q letters. An n-word y is called a descendant of a set of t codewords $x^1, \dots, x^t$ if $y_i \in \{x_i^1, \dots, x_i^t\}$ for all $i = 1, \dots, n$. A code is said to have the t-identifying parent property if for any n-word that is a descendant of at most t parents it is possible to identify at least one of them. We prove that for any $t \le q - 1$ there exist sequences of such codes with asymptotically nonvanishing rate.

Key words. Helly property, error-correcting codes, identifying parent property

AMS subject classifications. 94A60, 05C65

PII. S0895480100376848

1. Introduction.

Let Q be an alphabet of size q, and let us call any subset C of Q^n an (n, M)-*code* when $|C| = M$. Elements $x = (x_1, \dots, x_n)$ of C will be called *codewords*.

Let C be an (n, M)-code. Suppose $X \subseteq C$. For any coordinate i define the *projection*

$$P_i(X) = \bigcup_{x \in X} x_i.$$

Define the *envelope* $e(X)$ of X by

$$e(X) = \{x \in Q^n : \forall i, x_i \in P_i(X)\}.$$

Elements of the envelope $e(X)$ will be called *descendants* of X. Observe that $X \subseteq e(X)$ for all X, and $e(X) = X$ if $|X| = 1$.

Given a word $s \in Q^n$ (a son) which is a descendant of X, we would like to identify without ambiguity at least one member of X (a parent). When this is always possible for any descendant s of an X of size two, the code C is said to have the *identifiable parent property* [9]. More generally, we have the following definition.

DEFINITION 1.1. *For any $s \in Q^n$ let $\mathcal{H}_t(s)$ be the set of subsets $X \subset C$ of size at most t such that $s \in e(X)$. We shall say that C has the* identifiable parent property

*Received by the editors August 11, 2000; accepted for publication June 4, 2001; published electronically August 29, 2001. These results were presented at the International Workshop on Coding and Cryptography, Paris, France, 2001.
http://www.siam.org/journals/sidma/14-3/37684.html

†Bell Labs, Lucent Technologies, 600 Mountain Ave., Rm. 2C-375, Murray Hill, NJ 07974 and IPPI RAN, Moscow, Russia (abarg@research.bell-labs.com). The research of this author was supported in part by Binational Science Foundation (USA-Israel) under grant 1999099.

‡ENST, 46 rue Barrault, 75013 Paris, France (cohen@infres.enst.fr).

§HSH, Bjørnsonsg. 45, 5528 Haugesund, Norway (sbe@hsh.no).

¶IPPI RAN, Bol'shoj Karetnyj 19, Moscow 101447, Russia (kaba@iitp.ru). This research was done while the author was visiting DIMACS Center, Rutgers University, Piscataway, NJ, in March, 2000. The research of this author was supported in part by the Russian Foundation for Fundamental Research grant 99-01-00828.

‖ENST, 46 rue Barrault, 75013 Paris, France (zemor@infres.enst.fr).

of order t *(or is a t-identifying code, or is t i.p.p. for short) if for any $s \in Q^n$, either* $\mathcal{H}_t(s) = \emptyset$ *or*

$$\bigcap_{X \in \mathcal{H}_t(s)} X \neq \emptyset.$$

It will be convenient to view $\mathcal{H}_t(s)$ as the set of edges of a hypergraph. Its vertices are codewords of C.

Example. Let $C \subset \{0,1,2,3\}^4$ be the code defined by $C = \{u, v, w, x, y, z\}$ where

$$\begin{aligned} u &= \begin{bmatrix} 0 & 1 & 2 & 3 \end{bmatrix}, \\ v &= \begin{bmatrix} 1 & 2 & 3 & 0 \end{bmatrix}, \\ w &= \begin{bmatrix} 2 & 3 & 0 & 1 \end{bmatrix}, \\ x &= \begin{bmatrix} 3 & 0 & 1 & 2 \end{bmatrix}, \\ y &= \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}, \\ z &= \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}. \end{aligned}$$

The triple $\{w, y, z\}$ can produce the son $s = (2010)$. The hypergraph $\mathcal{H}_3(s)$ contains three edges, namely, $X = \{v, w, x\}$, $X' = \{w, x, y\}$, and $X'' = \{w, y, z\}$. Their intersection is $X \cap X' \cap X'' = \{w\}$, and w is therefore identified as a parent of s. The code C, however, is not 3-identifying; it is not even 2-identifying since $\mathcal{H}_2(0101) = \{u, w\} \cup \{y, z\}$ and $\{u, w\} \cap \{y, z\} = \emptyset$.

The concept of t-identification originates with the work of Chor, Fiat, and Naor [5] on broadcast encryption. It is also related to the problem of fingerprinting numerical data [4].

It is not difficult to prove that if the minimum Hamming distance of C is big enough, then C must be t-identifying. We have [5] the following proposition.

PROPOSITION 1.2. *If C has minimum Hamming distance d satisfying*

$$d > (1 - 1/t^2)n,$$

then C is a t-identifying code.

Actually, this condition implies a stronger property, namely, t-traceability; see [15].

As usual, let $R = R(C) = \log_q M/n$ denote the rate of the (n, M)-code C. Let $R_q(t) = \liminf_{n\to\infty} \max R(C_n)$, where the maximum is computed over all t-identifying codes C_n of length n.

Note that for alphabet sizes $q \leq t^2$, Proposition 1.2 does not prove that $R_q(t) > 0$ (because, for example, (n, M)-codes that satisfy the distance condition must have $M \leq qd$; see Plotkin's bound, e.g., in [12]).

In fact, nontrivial t-identifying codes do not always exist if the alphabet size q is not big enough. Hollman et al. [9] give constructions of 2-identifying codes and existence bounds on $R_q(2)$ for any alphabet size $q \geq 3$. They prove

$$R_q(2) \geq \log_q(q/(4q^2 - 6q + 3)^{1/3}). \tag{1.1}$$

The case of arbitrary t was discussed in a recent paper [15], where it is shown that nontrivial t-identifying codes do not exist when $t > q - 1$ and do exist when $q \geq \lfloor (t+2)^2/4 \rfloor$. Consequently, it was asked in [15] whether $R_q(t) > 0$ for any $t \leq q - 1$. In this paper we shall answer this question and prove the following theorem.

THEOREM 1.3. *$R_q(t) > 0$ if and only if $t \le q - 1$.*

We shall also give a lower bound on $R_q(t)$. We shall give particular attention to the case $t = 3$ and in the case $t = 2$, $q = 3$ strengthen (1.1) by showing that it can be achieved by a sequence of linear ternary i.p.p. codes.

2. Decomposing the t-identifying property with the Berge–Duchet theorem. Let us call a subset of edges of a hypergraph a *star* if it has a nonempty intersection. A code C is t-identifying if all the nonempty hypergraphs $\mathcal{H}_t(s)$ are stars. In this section we give necessary and sufficient conditions for $\mathcal{H}_t(s)$ to be a star.

Let us say that a family of sets, any t (or less) of which have nonempty intersection, is *t-wise* intersecting.

Recall that a family of sets has the *t-Helly property* if every t-wise intersecting finite subfamily is a star.

Let us quote a Helly-type result due to Berge and Duchet [3]; see also [6, p. 393].

THEOREM 2.1. *A hypergraph has the t-Helly property if and only if, for every set A of $t + 1$ vertices, all the edges E such that $|E \cap A| \ge t$ share a common vertex.*

Let us reword this result for our purposes. Recall that a hypergraph on $t + 1$ vertices whose edges are all the t-subsets is called a t-simplex, denoted $K_t(t + 1)$.

COROLLARY 2.2. *The hypergraph $\mathcal{H}_t(s)$ has the t-Helly property if and only if it does not contain $K_t(t + 1)$ as a subhypergraph.*

Proof. Consider any set A of size $t + 1$ vertices of $\mathcal{H}_t(s)$. Let $E_1, E_2, \dots, E_m$ be all the edges of $\mathcal{H}_t(s)$ that have at least t vertices in A. Since the edges of $\mathcal{H}_t(s)$ have at most t vertices we have $|E_i| = t$ for all i and

$$|E_1 \cap E_2 \cap \cdots \cap E_m| = t + 1 - m.$$

Therefore this intersection is nonempty if and only if $m < t + 1$; i.e., $E_1, E_2, \dots, E_m$ do not make up the t-simplex with vertex set A. □

Reworded again, we get the following corollary.

COROLLARY 2.3. *Suppose the hypergraph $\mathcal{H}_t(s)$ has at least $t + 1$ vertices. Then it is a star if and only if*

1. *any t (or less) of its edges have a nonempty intersection;*
2. *it does not contain $K_t(t + 1)$.*

3. Ensuring t-identification for any $t \le q + 1$.

3.1. Hashing families. A subset C of Q^n is said to be *t-hashing* (or t-separating; see, e.g., [10]) if any t of its members have t distinct entries in some common coordinate $i \in \{1, \dots, n\}$.

LEMMA 3.1. *$C \subset Q^n$ is $(t + 1)$-hashing if and only if $\mathcal{H}_t(s)$ has the t-Helly property for every $s \in Q^n$.*

Proof. Suppose C is $(t + 1)$-hashing. Let A be any set of $t + 1$ codewords, and let $s \in Q^n$. Since there is a coordinate where the codewords of A are all different, there exists at least one subset $X \subset A$, $|X| = t$, such that $s \notin e(X)$. Therefore $\mathcal{H}_t(s)$ cannot contain a t-simplex.

Conversely, suppose C is not $(t + 1)$-hashing, so that there exists a subset A of $t + 1$ codewords such that for every coordinate i, there exist at least two distinct codewords a, b of A such that $a_i = b_i$. Then define $s \in Q^n$ by choosing, for every coordinate $i \in \{1, \dots, n\}$, a value that occurs at least twice among the a_i, $a \in A$. Then we have $s \in e(X)$ for every subset X of size t of A, which means that $\mathcal{H}_t(s)$ contains the t-simplex with vertex set A. □

Remark. A consequence of Lemma 3.1 is that when $q \leq t$, there are no q-ary t-identifying codes C of size $|C| \geq t+1$ (see Lemma 1.6 of [15]).

We now have a condition on C that ensures that all the hypergraphs $\mathcal{H}_t(s)$ do not contain t-simplexes. To apply Corollary 2.3 we now need a condition to ensure that any t edges of any $\mathcal{H}_t(s)$ have a nonempty intersection.

3.2. Partially hashing families.

DEFINITION 3.2. *Let us say that a subset $C \subset Q^n$ is (t,u)* partially hashing *if for any two subsets T, U of C such that $T \subset U \subset C$, $|T| = t$, $|U| = u$, there is some coordinate $i \in \{1, \ldots, n\}$ such that for any $x \in T$ and any $y \in U, y \neq x$, we have $x_i \neq y_i$.*

Remark. If $u = t+1$, then (t,u) partial hashing is the same as $(t+1)$-hashing.

The motivation for this last definition is the following lemma.

LEMMA 3.3. *Let $\mathcal{X}$ be a subset of edges of $\mathcal{H}_t(s)$, and let u be an upper bound on the number of vertices spanned by the edges of $\mathcal{X}$. If C is (t,u) partially hashing, then $\mathcal{X}$ is a star, i.e., $\cap_{X \in \mathcal{X}} X \neq \emptyset$.*

Proof. Let $U = \cup_{X \in \mathcal{X}} X$, so that $|U| \leq u$ by the hypothesis. Let T be some edge of $\mathcal{X}$. Because C is (t,u) partially hashing, there is some coordinate i satisfying the condition of Definition 3.2 for T and U. Then $s_i = x_i$ for some $x \in T$ because $s \in e(T)$. However, then the definition implies that for all $y \neq x$, $y \in U$, we have $y_i \neq s_i$. Since all edges X of $\mathcal{X}$ are in $\mathcal{H}_t(s)$ we conclude that they must all contain x. □

Lemma 3.1 means that to enforce t-identification it is sufficient to have $(t+1)$-hashing and any property which forces any t edges of $\mathcal{H}_t(s)$ to intersect. Since any t edges of $\mathcal{H}_t(s)$ span at most t^2 vertices, Lemma 3.3, together with the remark after Definition 3.2, now implies the following corollary.

COROLLARY 3.4. *If C is (t,t^2) partially hashing, then C is a t-identifying code.*

The (t,u) hashing property is easier to handle than t-identification; in particular, it will give us a lower bound on $R_q(t)$ through the probabilistic method.

3.3. A lower bound on the size of (t,u) partially hashing codes. Fix $t \leq q-1$ and let $u \geq t+1$. We apply the probabilistic method with expurgation (see, e.g., [2]) to (t,u) partially hashing codes. This means that we take a random (n,M)-code C and compute the expectation $\mathbf{E}$ of the number of pairs of subsets T, U, $T \subset U \subset C$, $|T| = t$, $|U| = u$, that contradict the (t,u) partially hashing property. Whenever $\mathbf{E} \leq M/2$, then $(n, M/2)$-codes with the (t,u) partially hashing property exist.

The probability that a given T and U violate the partially hashing property is

$$P_{t,u,n} = \left(1 - \frac{q(q-1)\cdots(q-t+1)(q-t)^{u-t}}{q^u}\right)^n = \left(1 - \frac{q!(q-t)^{u-t}}{(q-t)!q^u}\right)^n.$$

The expectation of the number of pairs T, U that violate the partially hashing property is

$$\mathbf{E} = \binom{M}{u}\binom{u}{t} P_{t,u,n}.$$

Writing $M = q^{Rn}$ and letting n go to infinity we get that infinite sequences of (t,u) partially hashing codes exist for all rates R such that $\log_q \mathbf{E} < Rn$, i.e., such that

$$uR + \frac{1}{n}\log_q P_{t,u,n} < R.$$

Hence we get the following lemma.

LEMMA 3.5. *Let $u \geq t+1$: infinite sequences of (t,u) partially hashing codes exist for all rates R such that*

$$R < \frac{1}{u-1} \log_q \frac{(q-t)!q^u}{(q-t)!q^u - q!(q-t)^{u-t}}.$$

As a consequence, applying Corollary 3.4, we obtain $R_q(t) > 0$ for any $q \geq t+1$ which proves Theorem 1.3.

3.4. Improvements: Forbidding minimal configurations. We shall now show that the quantity t^2 in Corollary 3.4 can be lowered; namely, we shall obtain the following lemma.

LEMMA 3.6. *Let $u = \lfloor (t/2+1)^2) \rfloor$. If C is (t,u) partially hashing, then C is a t-identifying code.*

Before proving Lemma 3.6 it will be convenient to decompose all subsets of edges with empty intersection into "minimal forbidden configurations."

Let $\mathcal{X} = (X_1, \ldots, X_m)$ be a collection of subsets of codewords with $|X_i| \leq t$, $i = 1, \ldots, m$. We shall call $\mathcal{X}$ a *configuration* if it has an empty intersection, $\cap_{i=1}^m X_i = \emptyset$, and we shall say that $\mathcal{X}$ is a *minimal configuration* if it is minimal under inclusion, i.e., if $\cap_{i \neq j} X_i \neq \emptyset$ for any $j = 1, \ldots, m$.

Let $\mathcal{X}$ be a minimal configuration of size m. A set $B(\mathcal{X}) = (b^1, \ldots, b^m)$ will be called a *frame* of $\mathcal{X}$ if

$$b^j \in \bigcap_{i \neq j} X_i.$$

By minimality, frames of minimal configurations always exist. One useful property of frames gives rise to the following lemma, which follows somewhat along the lines of [15].

LEMMA 3.7. *Let $\mathcal{X}$ be a minimal configuration. Then*

$$\left| \bigcup_{i=1}^m X_i \right| \leq \sum |X_i| - m(m-2).$$

Proof. All the points b^j of a frame $B(\mathcal{X})$ are different since otherwise $\cap_i X_i \neq \emptyset$. Furthermore, by definition of $B(\mathcal{X})$ we have $(B(\mathcal{X}) \setminus b^j) \subset X_j$ for any $j = 1, \ldots, m$, and hence $m - 1 \leq t$. Then

$$\begin{aligned} \left| \bigcup_{i=1}^m X_i \right| &\leq \sum (|X_i \setminus (B(\mathcal{X}) \setminus b^i)|) + |B(\mathcal{X})| \\ &= \sum_{i=1}^m (|X_i| - (m-1)) + m. \qquad \square \end{aligned}$$

Since $|X_i| \leq t$, we obtain $|\cup X_i| \leq m(t-m+2)$. The maximum on m of this expression for $m = 1, \ldots, t$ is $u = \lfloor (t/2+1)^2) \rfloor$, which is also an upper bound on the cardinality of a minimal configuration (see [15]).

Note that the maximum value of m which gives a positive upper bound is $t+1$. Also note that the only minimal configurations with $m = t+1$ are t-simplexes. In particular this gives an alternative proof of Corollary 2.3.

We observe that any configuration must contain some minimal configuration. Now apply Lemma 3.3 as before to obtain Lemma 3.6. Lemmas 3.5 and 3.6 imply the following improved lower bound on $R_q(t)$.

THEOREM 3.8. *Let* $u = \lfloor (t/2+1)^2) \rfloor$. *We have*

$$R_q(t) \geq \frac{1}{u-1} \log_q \frac{(q-t)!q^u}{(q-t)!q^u - q!(q-t)^{u-t}}.$$

4. Small t. The (t,u) partially hashing property is only a sufficient condition for a code to be t-identifying. In the case of small t we can obtain precise necessary and sufficient conditions. A code C is t-identifying if and only if, for every s such that $\mathcal{H}_t(s) \neq \emptyset$, the hypergraph $\mathcal{H}_t(s)$ has the t-Helly property and any m edges of $\mathcal{H}_t(s)$ have a nonempty intersection for any m, $2 \leq m \leq t$. Lemma 3.1, together with Corollary 2.3, tells us therefore that C is t-identifying if and only if it is $(t+1)$-hashing and any m edges of $\mathcal{H}_t(s)$ have a nonempty intersection for any m, $2 \leq m \leq t$, and for any $s \in Q^n$. For $t = 2$, the latter property means that $\mathcal{H}_2(s)$ does not contain disjoint edges. Equivalently, for any $X = \{a,b\}$, $Y = \{c,d\}$, with a,b,c,d distinct codewords, $e(X) \cap e(Y) = \emptyset$. Equivalently again, this means that there exists a coordinate i such that

$$\{a_i, b_i\} \cap \{c_i, d_i\} = \emptyset. \tag{4.1}$$

This property of C was named IPP2 by Hollman et al. in [9]. In other contexts it has often been called $(2,2)$-separation and has been investigated by a number of authors [7, 8, 11, 13, 14]. The 3-hashing property was called IPP1 in [9].

Let us now characterize the 3 i.p.p. property.

4.1. The case $t = 3$. This time Lemma 3.1 and Corollary 2.3 tell us that C is 3-identifying if and only if

(i) it is 4-hashing;

(ii) for any $s \in Q^n$, any two edges of $\mathcal{H}_3(s)$ have a nonempty intersection and any three edges of $\mathcal{H}_3(s)$ have a nonempty intersection.

Condition (ii) is equivalent to saying that $\mathcal{H}_3(s)$ does not contain minimal configurations of size $m = 2$ and of size $m = 3$.

That $\mathcal{H}_3(s)$ does not contain minimal configurations of size two is equivalent to saying that for any $X = \{a,b,c\}$, $\{d,e,f\}$, with a,b,c,d,e,f distinct codewords, $e(X) \cap e(Y) = \emptyset$, which means that there exists a coordinate i such that

$$\{a_i, b_i, c_i\} \cap \{d_i, e_i, f_i\} = \emptyset. \tag{4.2}$$

Property (4.2) is usually called $(3,3)$-separation [7, 8, 11, 14].

Remark. As proved in [15] and follows from Corollary 2.3, the $(t+1)$-hashing and (t,t) separation properties are necessary for a code to have the t i.p.p. property. For $t = 2$, they are also sufficient [9]. For $t = 3$, we show how to complement them to form a set of sufficient conditions.

There remains to characterize the condition that $\mathcal{H}_3(s)$ does not contain a minimal configuration of three edges, i.e., $\mathcal{X} = (X,Y,Z)$ such that $X \cap Y \cap Z = \emptyset$, but any two edges of $\mathcal{X}$ intersect. Clearly, the only cases that we need to consider are when $|X| = |Y| = |Z| = 3$. We have two situations to forbid:

(a) for any $X,Y,Z \subset \binom{C}{3}$ such that $X \cap Y \cap Z = \emptyset$ and $|X \cap Y| = |Y \cap Z| = |Z \cap X| = 1$;

(b) for any $X, Y, Z \subset \binom{C}{3}$ such that $X \cap Y \cap Z = \emptyset$ and $|X \cap Y| = 2$ and $|Y \cap Z| = |Z \cap X| = 1$.

Forbidding these configurations means that we must have, in each case,

$$e(X) \cap e(Y) \cap e(Z) = \emptyset.$$

Those two cases involve, respectively, six and five codewords. After a straightforward examination we finally obtain the following proposition.

PROPOSITION 4.1. *C is 3-identifying if and only if the four following conditions hold:*

1. *C is 4-hashing.*
2. *C is $(3,3)$-separating.*
3. *For any six distinct codewords a, b, c, d, e, f there exists a coordinate i such that*
   - *a_i, b_i, c_i are all different,*
   - *$d_i \neq a_i$, $e_i \neq b_i$, $f_i \neq c_i$, and*
   - *d_i, e_i, f_i are not all equal.*
4. *For any five distinct codewords a, b, c, d, e there exists a coordinate i such that*
   - *$c_i \neq e_i$ and $\{a_i, b_i\} \cap \{c_i, e_i\} = \emptyset$, and*
   - *$d_i \neq a_i$ and $d_i \neq b_i$.*

Example. q=4. By repeatedly applying the probabilistic expurgation method, we get lower bounds on the rate of codes satisfying the previous four conditions. Namely,

$$\begin{aligned} R_1 &\geq \frac{1}{3}\log_4(32/29), \\ R_2 &\geq \frac{1}{5}\log_4(2^{10}/919), \\ R_3 &\geq \frac{1}{5}\log_4(256/217), \\ R_4 &\geq \frac{1}{4}\log_4(256/226). \end{aligned}$$

Taking the smallest of the R_i's gives a 3-identifying quaternary code with rate R_2, showing that

$$R_4(3) \geq \frac{1}{5}\log_4(1024/919) \approx 0.0156.$$

The lower bound of Theorem 3.8 gives only

$$R_4(3) \geq \frac{1}{5}\log_4(1024/1018) \approx 0.000848.$$

4.2. Linear i.p.p. codes. Bound (1.1) for $q = 3$ implies that

$$R_3(2) \geq (1/3)\log_3(9/7).$$

We strengthen this result by proving that the same bound holds for linear codes as well. The result in [9] implies only the existence of unrestricted codes with the same rate.

THEOREM 4.2. *There exists a sequence of linear ternary 2-identifying codes C_n with $R(C_n) \geq (1/3)\log_3(9/7)$.*

Proof. We again apply the probabilistic method and prove Theorem 4.2 by averaging this time over the ensemble of linear ternary codes. Let C be a linear subspace of F_3^n. The code C is i.p.p. if any triple of vectors in it is 3-hashing and any quadruple satisfies the separation condition (4.1).

Linear $(2,2)$-separating codes were first studied in [13], where most of the calculations below were essentially carried out. We include them here for completeness, showing that there exist such codes of rate $R \geq (1/3)\log_3(9/7)$.

Consider condition (4.1). Suppose that $\dim C = k$ and let G be a generator matrix of C, i.e., a $k \times n$ matrix whose rows form a basis of C as an F_3-linear space. Let $g^1, g^2, \ldots, g^n$ be the columns of G. Any vector $c \in C$ has the form aG for some $a \in F_3^k$. Let $c^1, \ldots, c^4$ be some vectors in C. Since the i.p.p. property is translation invariant, suppose that $c^4 = 0$. Suppose that $c^i = a^i G$ for $i = 1, 2, 3$.

Case (a). a^1, a^2, a^3 are linearly independent. Choose a basis $f_1, \ldots, f_k$ in F_3^k such that $a^i \cdot f_j = \delta_{ij}$ for $i = 1, 2, 3$; $j = 1, \ldots, k$. (Complement a^1, a^2, a^3 to a basis and take the dual basis.) Observe $\{c_m^1, c_m^2\} \cap \{c_m^3, c_m^4\} = \emptyset$ (for any given $m = 1, 2, \ldots, n$) if and only if the first three coordinates of the column g^m in the basis (f) have one of the following forms:

$$\begin{array}{ccc} \pm 1 & \pm 1 & 0 \\ -1 & -1 & 1 \\ 1 & 1 & -1 \end{array}.$$

Hence the total number of favorable choices is 6 out of 27. This implies that the probability for a matrix G to be bad for a given linearly independent triple is $(21/27)^n = (7/9)^n$. The number of triples is less than 3^{3k}, so the probability that a given matrix spans a quadruple of vectors that violate condition (4.1) is bounded above by $3^{3k}(7/9)^n$. Hence if $R = (1/3)\log_3(9/7) - \epsilon$, for any $\epsilon > 0$, there exists a favorable choice.

Case (b). Some of the vectors a^1, a^2, a^3 are linearly dependent. For instance, suppose that a^3 is spanned by a^1, a^2, and these two are not collinear. Let $a^3 = a^1 + a^2$. Take the basis dual to a basis that includes a^1, a^2. As above, we count the number of unfavorable choices for the column g^m. Good choices for (g_1^m, g_2^m) are $(\pm 1, \pm 1)$. Hence the fraction of bad choices of G is at most $3^{2k}(5/9)^n$, and this is less than $3^{3k}(7/9)^n$. Other cases of dependence are dealt with analogously; none accounts for a fraction of bad matrices larger than in Case (a).

Now let us give a lower bound on the rate of linear 3-hashing codes. (This is a special case of a result announced in [1, Thm. 2].) Again let c^1, c^2, c^3 be some vectors in C. Since the 3-hash property is translation invariant we can assume that $c^3 = 0$. Suppose that $c^i = a^i G$ for $i = 1, 2$. There are two cases.

Case (a). a^1 and a^2 are linearly independent. Choose a basis $f_1, \ldots, f_k$ in F_3^k such that $a^i \cdot f_j = \delta_{ij}$ for $i = 1, 2$; $j = 1, \ldots, k$. Observe that $c_m^1 \neq c_m^2 \neq 0$ if and only if the first two coordinates of the column g^m in the basis (f) equal either $(1, -1)$ or $(-1, 1)$. Hence the total number of favorable choices is 2 out of 9. This implies that the probability for a matrix G to be bad for a given linearly independent pair is $(7/9)^n$. The number of pairs is less than 3^{2k}, so the probability that a given matrix spans a triple of vectors such that a^1 and a^2 are linearly independent, and such that they violate the 3-hash condition, is bounded above by $3^{2k}(7/9)^n$. Hence if $R = (1/2)\log_3(9/7) - \epsilon$, for any $\epsilon > 0$, there exists a favorable choice.

Case (b). a^1 and a^2 are collinear, i.e., $a^1 = \lambda a^2$. As above, take the basis dual to a basis that includes a^1. Good choices for g_1^m are ± 1. Hence the number of bad choices of G is at most $3^{2k}(1/3)^n$, and this is less than $3^{2k}(7/9)^n$.

It remains to find the minimum of the achievable rates for $(2, 2)$-separating linear codes and for linear 3-hashing codes. This minimum is $(1/3)\log_3(9/7)$ as was to be proved. □

The argument in this section is generalized directly to prove existence of linear 2 i.p.p. codes that reach bound (1.1) over any finite field alphabet.

REFERENCES

[1] L. A. Bassalygo, M. Burmester, A. Dyachkov, and G. Kabatianski, *Hash codes*, in Proceedings of the IEEE International Symposium on Information Theory, Ulm, Germany, 1997, p. 174.

[2] L. A. Bassalygo, S. I. Gelfand, and M. S. Pinsker, *Simple methods for obtaining lower bounds in coding theory*, Problems Inform. Transmission, 27 (1991), pp. 277–281.

[3] C. Berge and P. Duchet, *A generalisation of Gilmore's theorem*, in Recent Advances in Graph Theory, M. Fiedler, ed., Academia, Prague, 1975, pp. 49-55.

[4] D. Boneh and J. Shaw, *Collusion-secure fingerprinting for digital data*, IEEE Trans. Inform. Theory, 44 (1998), pp. 480–491.

[5] B. Chor, A. Fiat, and M. Naor, *Tracing traitors*, in Advances in Cryptology–CRYPTO'94, Lecture Notes in Comput. Sci. 839, Springer-Verlag, Berlin, 1994, pp. 257–270.

[6] P. Duchet, *Hypergraphs*, in Handbook of Combinatorics, Vol. 1, R. L. Graham, M. Grötschel and L. Lovász, eds., North-Holland, Amsterdam, 1995, pp. 381–432.

[7] M. L. Fredman and J. Komlós, *On the size of separating systems and families of perfect hash functions*, SIAM J. Algebraic Discrete Methods, 5 (1984), pp. 61–68.

[8] A. D. Friedman, R. L. Graham, and J. D. Ullman, *Universal single transition time asynchronous state assignments*, IEEE Trans. Comput., 18 (1969), pp. 541–547.

[9] H. D. L. Hollmann, J. H. van Lint, J.-P. Linnartz, and L. M. G. M. Tolhuizen, *On codes with the identifiable parent property*, J. Combin. Theory Ser. A, 82 (1998), pp. 121–133.

[10] J. Körner and A. Orlitski, *Zero-error information theory,* IEEE Trans. Inform. Theory, 44 (1998), pp. 2207–2229.

[11] J. Körner and G. Simonyi, *Separating partition systems and locally different sequences*, SIAM J. Discrete Math., 1 (1988), pp. 355–359.

[12] J. H. van Lint, *Introduction to Coding Theory*, Springer-Verlag, Berlin, 1982.

[13] M. S. Pinsker and Yu. L. Sagalovich, *Lower bound for the power of an automaton state code*, Problems Inform. Transmission, 8 (1972), pp. 224–230.

[14] Yu. L. Sagalovich, *Separating systems*, Problems Inform. Transmission, 30 (1994), pp. 105–123.

[15] J. N. Staddon, D. R. Stinson, and R. Wei, *Combinatorial properties of frameproof and traceability codes*, IEEE Trans. Inform. Theory, 47 (2001), pp. 1042–1049.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 433–445

FORCING STRUCTURES AND CLIQUES IN UNIQUELY VERTEX COLORABLE GRAPHS*

AMIR DANESHGAR†

Abstract. Let G be a simple undirected uniquely vertex k-colorable graph, or a k-UCG for short. M. Truszczyński [*Some results on uniquely colorable graphs*, in Finite and Infinite Sets, North-Holland, Amsterdam, 1984, pp. 733–748] introduced $e^*(G) = |V(G)|(k-1) - \binom{k}{2}$ as the minimum number of edges for a k-UCG and S. J. Xu [*J. Combin. Theory Ser. B*, 50 (1990), pp. 319–320] conjectured that any minimal k-UCG contains a K_k as a subgraph. In this paper, first we introduce a technique called *forcing*. Then by applying this technique in conjunction with a feedback structure we construct a k-UCG with clique number $k-t$, for each $t \geq 1$ and each k, when k is large enough. This also improves some known results for the case $t = 1$.

Second, we analyze the parameter $\Lambda(G) = |E(G)| - e^*(G)$ for our constructions, and we obtain some bounds for the functions

$$\lambda_t(k) = \min\{\Lambda(G) \ : \ G \text{ is a } k\text{-UCG and } cl(G) = k - t\},$$

$$\nu_t(k) = \min\{|V(G)| \ : \ G \text{ is a } k\text{-UCG and } cl(G) = k - t\}.$$

Also, we introduce some possible applications of the technique in cryptography and data compression.

Key words. uniquely vertex colorable graphs

AMS subject classification. 05C15

PII. S0895480196304994

1. Introduction and preliminaries.

In this paper, we consider *finite, simple*, and *undirected* graphs. For such a graph G, $V(G)$ and $E(G)$ are *vertex set* and *edge set* of G, while $|V(G)|$ and $|E(G)|$ are *order* and *size* of G, respectively. A *k-vertex-coloring* of G is a partition of $V(G)$ into k color classes such that vertices in the same class are not adjacent. Moreover, G is called *uniquely vertex k-colorable* (or a k-UCG for short) if every k-coloring of it induces the same partition on $V(G)$. Also, for a k-UCG, G, $cl(G)$ is the *maximum clique number* of G, and $ccl(G) = k - cl(G)$ is defined to be the *coclique* of G.

k-UCG's have been studied by Aksionov [1], Bollobás [2, 3], Borwiecki and Burchardt [5], Chartrand and Geller [7], Harary [18], Harary, Hedetniemi, and Robinson [19], Xu [26], and Truszczyński [25]. In [25] Truszczyński introduced $e^*(G) = |V(G)|(k-1) - \binom{k}{2}$ as the minimum number of edges for a k-UCG and S. J. Xu [26] proposed the following conjecture.

CONJECTURE 1.1 ([26]). *If G is a k-UCG and $|E(G)| = e^*(G)$, then G contains a K_k as its subgraph.*

This conjecture will be our major motivation throughout this paper, where we try to study the clique structure of k-UCG's. Of course, it should be noted that the existence of k-UCG's with large girth has already been established by Bollobás and

*Received by the editors June 7, 1996; accepted for publication (in revised form) June 11, 2001; published electronically October 4, 2001. This research was partially supported by the Institute for Studies in Theoretical Physics and Mathematics (IPM) and the Research Council of the Sharif University of Technology.

http://www.siam.org/journals/sidma/14-4/30499.html

†Department of Mathematical Sciences, Sharif University of Technology, P.O. Box 11365–9415, Tehran, Iran (daneshgar@sharif.edu).

Sauer [4] using probabilistic methods based on the work of Erdös and Spencer [17] and by Müller [22] based on a construction of Lovász [20] for k-chromatic graphs with large girth. However, it seems that it is quite difficult to determine exact values of the function

$$\nu_t(k) = \min\{|V(G)| \ : \ G \text{ is a } k\text{-UCG and } cl(G) = k - t\}.$$

The only known result about this function is the following fact about $\nu_1(k)$ [25]:

$$\nu_1(k) = \begin{cases} 12 & k = 3 \\ 10 & k = 4 \\ k + 5 & k \geq 5. \end{cases}$$

Our basic goals are twofold. First, we introduce a technique which will be called *forcing*. By using this technique in conjunction with a feedback structure, we construct families of k-UCG's with maximum clique number $k-t$ for each $t \geq 1$. It will be shown that the technique is very suitable to reduce the parameter $\Lambda(G) = |E(G)| - e^*(G)$, and in this regard, we also enhance some known results for the case $t = 1$.

Second, we address ourselves to Xu's conjecture, and we analyze $\Lambda(G)$ for our graphs. Although we will not prove or disprove Xu's conjecture in this paper, this analysis reveals a relationship among Λ, the number of vertices, and the maximum clique number in k-UCG's. In this direction, we introduce the following counterpart to $\nu_t(k)$:

$$\lambda_t(k) = \min\{\Lambda(G) \ : \ G \text{ is a } k\text{-UCG and } cl(G) = k - t\},$$

and we prove theorems which show that

$$\lambda_1(k) \leq 1, \qquad \lambda_2(k) \leq 6 \quad (k \geq 7),$$

$$\lambda_t(k) \leq \frac{(t+1)!}{2}\left(\sum_{j=1}^{t-1} \frac{1 + \delta_{j+1}}{j!} + 1\right) \qquad \left(t > 1,\ k \geq \binom{t+2}{2} + 1\right), \text{ and}$$

$$\nu_t(k) \leq k - k_{min} + (t+1)!\left(\sum_{j=1}^{t-1} \frac{j^2 + j + 6}{(j+2)!} + 5\right) \quad \left(t > 1,\ k \geq k_{min} = \frac{t^2 + t + 6}{2}\right).$$

(For the definition of δ_j see Theorem 4.2.) Also, we introduce some possible applications of our results in data security and data compression, and we propose some basic questions.

Throughout this paper, colors are denoted by numbers $0, 1, 2, \ldots$, and $\mathbf{Z}_d$ is the additive cyclic group of order d. If $x \in V(G)$, then $[x]$ is the color class which contains x, and the following notation is used for the sake of simplicity:

$$\forall\ m, n \in \mathbf{Z} \quad [m, n] \ = \ \begin{cases} \phi, & n < m, \\ \{m, m+1, \ldots, n\}, & m \leq n. \end{cases}$$

Moreover, we introduce a simple lemma which will be used frequently throughout this paper. (For more on this type of invariant and related topics see [11, 15, 16].)

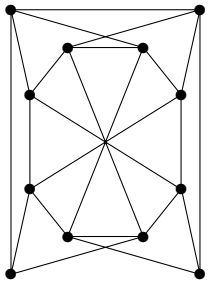


FIG. 1. *A 3-UCG without any triangle and* $\Lambda = 2$.

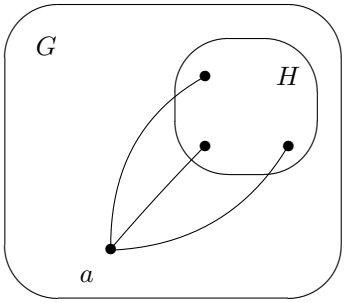


FIG. 2. *Forcing on vertex a.*

LEMMA 1.2. *Let G be a k*-UCG. *Build a* $(k+1)$-UCG, H, *by adding a new vertex v which is connected to all vertices of G. Then,* $\Lambda(G) = \Lambda(H)$ *and* $ccl(G) = ccl(H)$.

Proof.

$$\Lambda(H) = |E(H)| - k|V(H)| + \binom{k+1}{2} = |E(G)| - |V(G)|(k-1) + \binom{k}{2} = \Lambda(G). \qquad \square$$

2. A k-UCG which contains no K_k $(t = 1)$. Before we proceed, it should be noted that the existence of a 3-UCG without any triangle on 12 vertices has already been verified [6, 23, 25] (Figure 1). This, in conjunction with Lemma 1.2 gives rise to a family of k-UCG's with $k + 9$ vertices, maximal clique number $k - 1$, and $\Lambda = 2$ for each $k \geq 3$. However, in this section we are going to reduce this bound on the number of vertices to $k + 6$ (with $\Lambda = 1$) by means of a technique which will be called *forcing* [10, 11].

This will be our major construction method throughout this paper. We illustrate the basic idea in a clique reduction to $k - 1$ in this section.

Consider Figure 2 and the graph G with $\chi(G) = k$ such that the vertex a is connected to all color classes of the $(k-1)$-UCG, H. Also, assume that vertices of H cannot take the color $k - 1$. Then since H admits all colors from 0 to $k - 2$, a is forced to take the color $k - 1$, regardless of the exact coloring of H. Now one can use a feedback structure of such subgraphs to indirectly fix the colors of some vertices in G. After that, direct connections can be used to fix the rest of the colors without increasing the clique number.

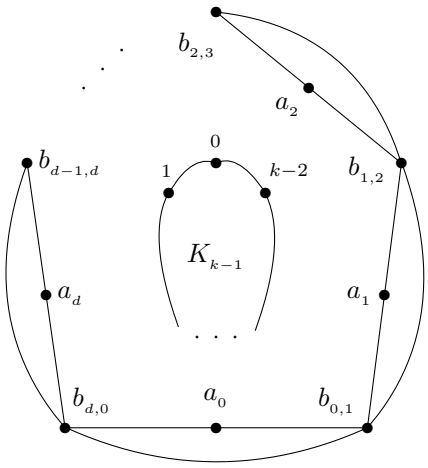


FIG. 3. *The graph* $\mathsf{H}_1(d)$.

To illustrate our technique, for each $k > 3$, we construct a k-UCG which contains no K_k as a subgraph.

Let's consider the graph $\mathsf{H}_1(d)$ as it is depicted in Figure 3 in which we have $1 < d < k-1$. In this graph, we assume that the central clique has taken colors 0 to $k-2$, and a_i is connected to all vertices in the central clique except the vertex with color i. Also, $b_{i-1,i}$ is connected to all vertices of the central clique except vertices with colors $i-1$ and i. We claim that in every k-coloring of this graph, a_i should take the color $k-1$ for all $i \in \mathbf{Z}_{d+1}$.

To see this, let a_0 take the color 0. Then $b_{0,1}$ and a_1 form a 2-clique whose vertices can take only colors 1 and $k-1$. Hence, $b_{1,2}$ is forced to take the color 2. This forces $b_{2,3}$ to take the color 3, and so forth. Therefore, $b_{d,0}$ is forced to take the color 0 by the feedback structure, which is a contradiction, since it is connected to a_0. The rest of the proof follows by symmetry.

This shows that a_i should take the color $k-1$ in any k-coloring of $\mathsf{H}_1(d)$. Now, it is quite easy to build a k-UCG from $\mathsf{H}_1(d)$, since we have a number of vertices with fixed colors. Introduce a new vertex u_d and join it to $b_{d,0}$, a_1 and all vertices in the central clique except the vertex with color d. This forces $b_{d,0}$ to take the color 0 which again forces $b_{i-1,i}$ to take the color i for all $i \in \mathbf{Z}_{d+1}$.

We refer to this new graph by $\mathsf{U}_{k,1}(d)$, and we summarize our results in the following proposition.

PROPOSITION 2.1. *For the graph* $\mathsf{U}_{k,1}(d)$ $(1 < d < k-1)$ *we have the following:*

(1) $cl(\mathsf{U}_{k,1}(d)) = k-1$.
(2) $|V(\mathsf{U}_{k,1}(d))| = k + 2(d+1)$.
(3) $\Lambda(\mathsf{U}_{k,1}(d)) = 1$.

Note that in our construction of graph $\mathsf{U}_{k,1}(d)$ any vertex in the central $(k-1)$-clique whose color is in $\{d+1, \ldots, k-2\}$ is connected to any other vertex as in Lemma 1.2. Hence, this shows that the most important case of the construction is when d is equal to $k-2$. In this regard we define $\mathsf{U}_{k,1}$ as $\mathsf{U}_{k,1}(k-2)$. (This is actually the *core* of our construction [15], and it is also interesting to note that the graphs $\mathsf{U}_{k,1}$

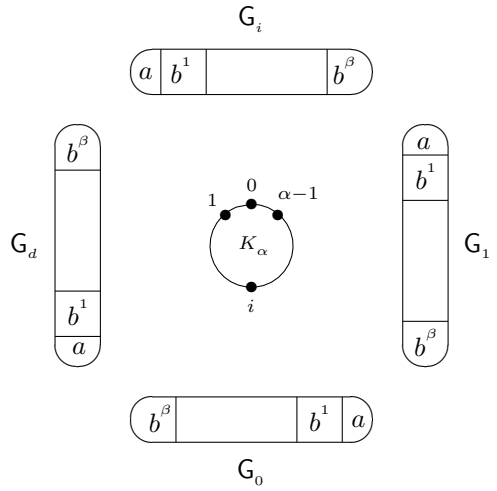


FIG. 4. *The graph* $\mathsf{H}_t(d)$.

for $k \geq 6$ are the only known graphs for which the fractional Hall number is strictly greater than the fractional Hall condition number [12, 14].) Therefore, we have the following proposition.

PROPOSITION 2.2. *For each* $k \geq 4$*, there exists a family of* k-UCG*'s,* $\mathsf{U}_{k,1}$*, such that* $|V(\mathsf{U}_{k,1})| = 3k-2$*,* $ccl(\mathsf{U}_{k,1}) = \Lambda(\mathsf{U}_{k,1}) = 1$*,* $\delta(\mathsf{U}_{k,1}) > k-1$*, and* $\mathsf{U}_{k,1}$ *does not have any color-class of size one.*

For the importance of the above proposition we refer the interested reader to [15]. Also, Proposition 2.1, for the case $d = 2$, can be used to reduce the bound on the number of vertices as follows.

THEOREM 2.3. *For each* $k \geq 4$*, there exists a family of* k-UCG*'s,* $\mathsf{U}_{k,1}(2)$*, such that* $|V(\mathsf{U}_{k,1}(2))| = k+6$ *and* $ccl(\mathsf{U}_{k,1}(2)) = \Lambda(\mathsf{U}_{k,1}(2)) = 1$.

The graph $\mathsf{U}_{4,1}$ is different from the graph H_4 which is introduced in [25], since $\Lambda(H_4) = 2$ but $\Lambda(\mathsf{U}_{4,1}) = 1$. On the other hand, the graph $\mathsf{U}_{4,1}$ has a close relationship to Δ–color-critical graphs [13].

Also, Theorem 2.3 shows that the bound in Xu's conjecture is the best possible. Moreover, it shows that it is quite difficult to prove the correctness of this conjecture (if it is true), since one cannot use any kind of approximation technique on the number of edges.

This simple example shows a relationship among maximal clique number, number of vertices and Λ. We pursue this point of view in the last section.

3. The generalized construction ($t > 1$). In this section we are going to generalize our cyclic construction of the previous section for cocliques $t > 1$. To do this, for a fixed $t > 1$, let G_i $(i \in \mathbf{Z}_{d+1})$ be a $(\beta+1)$-UCG with $cl(\mathsf{G}_i) = q > 2$ such that $\beta = t+q-2$. Then assuming $\alpha \stackrel{\text{def}}{=} k-t \geq 3$, $1 < d < k-\beta$, and using G_i's, we construct a k-colorable graph $\mathsf{H}_t(d)$ with coclique t whose general pattern is depicted in Figure 4 as a cyclic structure round a central clique K_α. In what follows, we address ourselves to the description of details of connections and colorings in this graph as well as some necessary conditions on graphs G_i's. In the next section we show how these conditions can be satisfied.

Without loss of generality we may assume that the central clique K_α has taken

colors 0 to $\alpha-1$, and we refer to the vertices by the corresponding colors. Also, we assume that there exists $B_i \subset V(\mathsf{G}_i)$ such that for each $i \in \mathbf{Z}_{d+1}$, we have the following:

($\mathcal{B}_1$) There exist two adjacent vertices a_i and $b^{\beta}_{i-1,i}$ such that

$$a_i \in V(\mathsf{G}_i), \quad a_i \notin B_i, \quad b^{\beta}_{i-1,i} \in B_i.$$

(Note that this implies $[a_i] \neq [b^{\beta}_{i-1,i}]$ in the unique coloring of G_i.)

($\mathcal{B}_2$) The induced subgraph on $B_i \cup \{a_i\}$ contains no q-clique.

($\mathcal{B}_3$) The induced subgraph on $V(\mathsf{G}_i) - B_i$ contains no q-clique.

Choose B_i minimal with respect to these conditions. Then choose $b^{j}_{i-1,i}$ for $1 \leq j < \beta$ from $V(\mathsf{G}_i)$ as representatives of the color classes in $\mathsf{G}_i - [b^{\beta}_{i-1,i}] - [a_i]$. These vertices should be chosen from B_i or if it is not possible they should be chosen such that

($\mathcal{B}_4$) $(\{b^{j}_{i-1,i} \mid 1 \leq j < \beta\} - B_i) \cup \{a_i\}$ contains no q-clique.

Hereafter, we assume that G_i's are identical copies of a $(\beta+1)$-UCG, G, whose maximal clique number is $q > 2$. In the next section we show that such a graph, G, exists and the above conditions can be satisfied.

In what follows we describe the interconnections among the G_i's and the central clique. Note that henceforth, $i \in \mathbf{Z}_{d+1}$ and

$$\Gamma \stackrel{\text{def}}{=} [d+1, d+q-2] \cup [\alpha, k-1].$$

- a_i is connected to all of the vertices in the central clique except the vertex with color i.
- Each vertex in B_i is connected to all vertices in the central clique except those with colors in $\Gamma \cup \{i-1, i\}$, and each vertex in $\mathsf{G}_i - B_i - \{a_i\}$ is connected to all vertices in the central clique except those with colors in $\Gamma \cup \{i\}$.
- For each i, both vertices a_i and $b^{\beta}_{i-1,i}$ are connected to all of the vertices in B_{i+1}.
- For each i, the vertex $b^{\beta}_{i,i+1}$ is connected to $b^{j}_{i-1,i}$ for all $1 \leq j \leq \beta$.

LEMMA 3.1. *$ccl(\mathsf{H}_t(d)) = t$.*

Proof. First, note that all vertices in the central clique with colors in $\{d+q-1, \ldots, \alpha-1\}$ are connected to any other vertex in $\mathsf{H}_t(d)$ as in Lemma 1.2. Therefore, it is sufficient to consider the case $\alpha = d+q-1$. Moreover, as the worst cases, by considering ($\mathcal{B}_1$)–($\mathcal{B}_4$), for each $i \in \mathbf{Z}_{d+1}$ we have the following.

- The maximum clique number of the graph induced on $V(\mathsf{G}_i) - B_i$ along with all vertices in the central clique whose colors are in $\{0, \ldots, d\} - \{i\}$ forms, at most, a clique of size $(q-1)+d = \alpha$.
- The maximum clique number of the graph induced on $B_{i+1} \cup \{a_i, b^{\beta}_{i-1,i}\}$ is, at most, $q+1$, which along with all vertices in the central clique whose colors are in $\{0, \ldots, d\} - \{i-1, i, i+1\}$ forms, at most, a clique of size $(d-2)+(q+1) = \alpha$.
- The maximum clique number of the graph induced on

$$(\{b^{j}_{i-1,i} \mid 1 \leq j < \beta\} - B_i) \cup \{a_i, b^{\beta}_{i,i+1}\}$$

is, at most, q, which along with all vertices in the central clique whose colors are in $\{0, \ldots, d\} - \{i, i+1\}$ forms, at most, a clique of size $(d-1)+q = \alpha$.

- The maximum clique number of the graph induced on

$$\{b^{j}_{i-1,i} \mid 1 \le j \le \beta\} \cup \{a_i, b^{\beta}_{i,i+1}\}$$

is, at most, q, which along with all vertices in the central clique whose colors are in $\{0,\dots,d\} - \{i-1, i, i+1\}$ forms, at most, a clique of size $(d-2)+q = \alpha - 1$.

Hence, $ccl(\mathsf{H}_t(d)) = k - \alpha = t$. □

LEMMA 3.2. *In any k-coloring of $\mathsf{H}_t(d)$, for each i, the vertex a_i cannot take the color i.*

Proof. By symmetry it suffices to prove the lemma for $i = 0$. Let a_0 take the color 0. Then since a_0 is connected to all vertices in B_1, each vertex of G_1 can take only colors from the set $\{1\} \cup \Gamma$ which has $t + q - 1 = \beta + 1$ elements. Therefore, G_1 is uniquely colored by these colors. Now, $b^{\beta}_{1,2}$ in G_2 is connected to all classes of G_1 and consequently takes the color 2. This eliminates color 2 from G_3, and so on. This cyclic forcing implies either that $b^{\beta}_{d-1,d}$ takes the color d or the graph G_d is uniquely colored by colors $\{d\} \cup \Gamma$. In the first case (d even), $b^{\beta}_{d-1,d}$ eliminates color d from G_0, and it is uniquely colored by colors $\{0\} \cup \Gamma$. This forces $b^{\beta}_{0,1}$ to take the color 1, and again by a cyclic forcing we may deduce that for any d, G_d is uniquely colored by colors $\{d\} \cup \Gamma$, which implies that $b^{\beta}_{d,0}$ is connected to all colors in $\{d\} \cup \Gamma$ and it is also connected to a_0 (by $\mathcal{B}_1$). However, this is a contradiction since this vertex can take only one of these colors.

In the second case (d odd), $b^{\beta}_{d,0}$ is connected to all colors in $\{d\} \cup \Gamma$ and it is also connected to a_0. However, this is also a contradiction since it can take only one of these colors. □

Note that in $\mathsf{H}_t(d)$ we now have $d+1$ vertices which can take only their colors from the set $[\alpha, k-1]$ (namely a_i's). These may be used in a fixing procedure to turn this graph into a k–UCG as described in what follows.

Consider $\mathsf{H}_t(t)$ (i.e., set $d = t$) and form a $(t-1)$-clique from a_i's for $i \neq t, 1$, and connect a_t and a_1 to all vertices of this clique. Therefore, without loss of generality, we may assume that a_t takes the color $\alpha+1$ and a_i takes the color $\alpha+i$ for $0 \le i < t$. Now for the process of fixing we consider two cases.

- *t even.*

  Connect $b^{\beta}_{t,0}$ to a_i for $1 < i < t$. Then $b^{\beta}_{t,0}$ should take one of the colors in $\{0\} \cup [t, \beta]$. Introduce a new vertex u_t and connect it to a_i for $0 \le i < t$ and to all vertices in the central clique except the vertex with color t. This forces u_t to take the color t. Also, introduce new vertices u_m for $m \in [t+1, \beta]$ and connect each u_m to a_i for $0 \le i < t$ and to all vertices in the central clique except the vertex with color m. This forces each u_m to take the color m.

  Connect $b^{\beta}_{t,0}$ to u_m for $m \in [t, \beta]$. Note that this forces this vertex to take the color 0 and by the feedback forcing each $b^{\beta}_{i-1,i}$ is forced to take the color i for $i \in \mathbf{Z}_{t+1}$.

  In order to fix the rest of the colors again, introduce new vertices u_m for $m \in [\alpha, k-1]$ such that each new u_m is connected to u_j for $j \in [t, \beta]$ and to a_i for $i \neq m, t$ and to all vertices in the central clique except those with colors in $[t, \beta]$. Note that with this construction each u_m is forced to take the color m for $m \in [t, \beta] \cup [\alpha, k-1]$.

  Now, note that in each G_i there are $\beta - 1$ color classes whose colors are not fixed. In order to fix the color of these classes it is sufficient to fix the color

of one representative in each class, which needs $\binom{\beta-1}{2}$ edges for each G_i using forcing by u_m's.

- *t odd.*
  We use the same technique in this case too; however, since t is odd, $b^{\beta}_{i-1,i}$ is forced to take the color i when i is even. Hence, since $b^{\beta}_{t-1,t}$ is connected to a_t and G_t is uniquely colored by colors in $\Gamma \cup \{t\}$, we have to use $\beta - 1$ extra edges, using u_m's, to fix the color of $b^{\beta}_{t-1,t}$ to color t. This guarantees that $b^{\beta}_{i-1,i}$ takes the color i when i is odd. The rest of the fixing procedure is the same as the previous case.

This construction gives rise to a k-UCG which will be called $\mathsf{U}_{k,t}(\mathsf{G})$, and as a consequence of the above constructions we have the following theorem.

THEOREM 3.3. *Let t, k, and q be natural numbers such that $t > 1$, $k - t \geq 3$, $k \geq 2t + q - 1$, and let G_i $(i \in \mathbf{Z}_{d+1})$ be identical copies of a $(t+q-1)$-UCG, G, with $cl(\mathsf{G}) = q > 2$ and a subset $B \subset V(\mathsf{G})$ of size b which satisfies conditions $(\mathcal{B}_1)$–$(\mathcal{B}_4)$. Then if $|V(\mathsf{G})| = \nu$, for the graph $\mathsf{U}_{k,t}(\mathsf{G})$ we have the following:*

(1) $ccl(\mathsf{U}_{k,t}(\mathsf{G})) = t$.
(2) $|V(\mathsf{U}_{k,t}(\mathsf{G}))| = \nu(t+1) + k + q - 1$.
(3) $\Lambda(\mathsf{U}_{k,t}(\mathsf{G})) = (t+1)(\Lambda(\mathsf{G}) + b) + R_t$, *where*

$$R_t = \begin{cases} q-4, & t \text{ even}, \\ 2q+t-7, & t \text{ odd}. \end{cases}$$

Proof. For (1) again note that all the vertices in the central clique with colors in $\{t+q-1, \ldots, \alpha-1\}$ are connected to any other vertex as in Lemma 1.2. Therefore, it is sufficient to consider the case $k - t = t + q - 1$, and we may focus only on the fixing process by Lemma 3.1. Therefore, we note that, as worst cases, connecting a_i's for $i \in [0, t-1]$ along with the vertex in the central clique whose color is t form a $(t+1)$-clique, while $\{a_i, a_{i+1}\} \cup B_{i+1}$ along with vertices in the central clique whose colors are in $[0,t] - \{i, i+1\}$ produce, at most, a clique of size $q + (t-1) = \alpha$. However, since $k - t = t + q - 1$ and $q > 2$ we have $k - t > t + 1$ which implies that $ccl(\mathsf{U}_{k,t}(\mathsf{G})) = t$.

(2) and (3) follow by direct computation. □

4. Clique reduction. In this section we use our previous constructions recursively to reduce the clique number in k-UCG's. In order to reduce the clique number to $k-2$ (i.e., $t = 2$), it is necessary to have a $(q+1)$-UCG with clique number q to be used as the graph G in our general construction (Theorem 3.3), and the first case is the case of a 4-UCG with maximum clique number $q = 3$. For this, we consider the graph $\mathsf{U}_{4,1}(2)$ constructed in section 2 (Figure 5), and in what follows we show that this graph can be used as the graph G in Figure 4 for the case $t = 2$ where $\beta = 3$.

Let $b_{2,0}$ serve as the vertex b^3 in G. Consider a_2 as a, $b_{0,1}$ as b^2 and the vertex with color 2 in the central clique as b^1. Also, consider B as the set of four vertices which are circled in Figure 5 and note that this set satisfies all conditions mentioned in the previous section (conditions $(\mathcal{B}_1)$–$(\mathcal{B}_4)$).

Now, using this graph as the graph G in $\mathsf{U}_{k,2}(\mathsf{G})$ gives rise to a k-UCG with clique number $k-2$. Therefore, for this graph we have

$$t = 2,\ q = \beta = 3,\ k \geq 2t + q - 1 = 6,\ b = 4,\ \nu = 10,\ |V| = k + 32,\ \Lambda = 14.$$

However, we can still reduce Λ by using some more vertices. Consider $\mathsf{U}_{5,1}(2)$ as the graph G. Let the vertex with color 3 in the central clique serve as $b^{\beta} = b^4$ and note

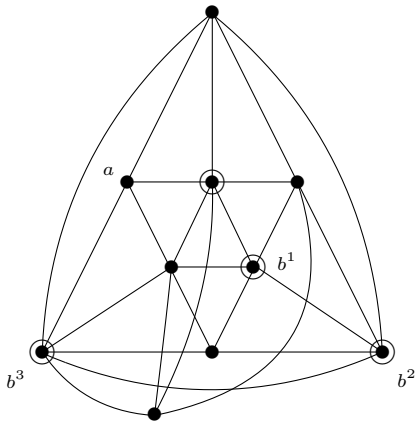


FIG. 5. *The graph* $\mathsf{U}_{4,1}$ *and the subset* $B \cup \{a\}$. *(Vertices of* B *are circled.)*

that this vertex is in all maximal cliques of this graph. Therefore, we may define $B = \{b^{\beta}\}$ and we define the rest of representatives as in the previous case for $\mathsf{U}_{4,1}(2)$. Hence, for $\mathsf{U}_{k,2}(\mathsf{U}_{5,1}(2))$ we have

$$t = 2,\ q = \beta = 4,\ k \geq 2t + q - 1 = 7,\ b = 1,\ \nu = 11,\ |V| = k + 36,\ \Lambda = 6.$$

Now applying Lemma 1.2 we have the following proposition.

PROPOSITION 4.1. *For* $k \geq 7$, *there exists a family of* k-UCG*'s,* $\mathsf{U}_{k,2}$, *such that* $|V(\mathsf{U}_{k,2})| = k + 36$, $ccl(\mathsf{U}_{k,2}) = 2$, *and* $\Lambda(\mathsf{U}_{k,2}) = 6$.

Consider such a graph for $k = 8$. In this way we can again set $b = 1$, and we may construct $\mathsf{U}_{k,3}$ $(k \geq 11)$ with $|V(\mathsf{U}_{k,3})| = k + 181$, $ccl(\mathsf{U}_{k,3}) = 3$, and $\Lambda(\mathsf{U}_{k,3}) = 36$.

This shows that we can always set $b = 1$ by sacrificing the minimality of the number of vertices in our construction. Therefore, by a recursive construction we get the following generalization of Theorem 2.3.

THEOREM 4.2. *If* $t > 1$ *is fixed, then for each* $k \geq k_{min}^{t} = \binom{t+2}{2} + 1$ *there exists a* k-UCG, $\mathsf{U}_{k,t}$, *such that*

(1) $ccl(\mathsf{U}_{k,t}) = t$;

(2) $|V(\mathsf{U}_{k,t})| = k + (V_{min}^{t} - k_{min}^{t})$, *where* $V_{min}^{t} = (t+1)! \left(\sum_{j=1}^{t-1} \frac{j^2 + 4j + 8}{(j+2)!} + 5 \right)$;

(3) $\Lambda(\mathsf{U}_{k,t}) = \frac{(t+1)!}{2} \left(\sum_{j=1}^{t-1} \frac{1 + \delta_{j+1}}{j!} + 1 \right)$, *where* $\delta_j = \begin{cases} 0, & j \text{ even}, \\ 1, & j \text{ odd}. \end{cases}$

Proof. Assume that superscript t denotes our level of recursion (for the clique number $k - t$). Let k_{min}^{t} be the minimum coloring number for which the above construction is valid, where we consider the vertex with color $q^{t-1} + t - 2$ in $\mathsf{U}_{k,t-1}$ as b^{β} which is connected to all the rest of vertices. This implies that $b = 1$, $k_{min}^{t} = q^{t} + 2t - 1$, and consequently $k_{min}^{t-1} = q^{t-1} + 2t - 3$. On the other hand, we have

$$q^{t} = (k_{min}^{t-1} + 1) - (t - 1) = k_{min}^{t-1} - t + 2$$

which gives rise to the following difference equations:

$$q^{t} = q^{t-1} + t - 1, \quad q^{2} = 4,$$

$$k_{min}^{t} - t = k_{min}^{t-1} + 1, \quad k_{min}^{2} = 7,$$

with solutions $q^{t} = \binom{t}{2} + 3$, $k_{min}^{t} = \binom{t+2}{2} + 1$. Also, by Theorem 3.3 we have

$$\Lambda^{t} = (t+1)\Lambda^{t-1} + (1+\delta_t)(q^{t} + t - 3) = (t+1)\Lambda^{t-1} + (1+\delta_t)\binom{t+1}{2}, \quad \Lambda^{2} = 6,$$

and

$$V_{min}^{t} = (V_{min}^{t-1} + 1)(t+1) + k_{min}^{t} + q^{t} - 1 = V_{min}^{t-1}(t+1) + t^2 + 2t + 5, \quad V_{min}^{2} = 43,$$

whose solutions are

$$V_{min}^{t} = (t+1)! \left(\sum_{j=1}^{t-1} \frac{j^2 + 4j + 8}{(j+2)!} + 5 \right)$$

and

$$\Lambda(\mathsf{U}_{k,t}) = \frac{(t+1)!}{2} \left(\sum_{j=1}^{t-1} \frac{1+\delta_{j+1}}{j!} + 1 \right),$$

where $\delta_j = \begin{cases} 0, & j \text{ even}, \\ 1, & j \text{ odd}. \end{cases}$ □

COROLLARY 4.3. *If t is large enough, then*

(a) $V_{min}^{t} \simeq 8.3(t+1)!$;

(b) $\Lambda^{t} \simeq 1.6(t+1)!$.

Moreover, as a special asymptotic result we have the following theorem.

THEOREM 4.4. *If $k = \binom{t+2}{2} + 1$ and $t > 1$ is large enough, then there exists a family of k-UCG's, U_t, such that $ccl(\mathsf{U}_t) = t$ and $V(\mathsf{U}_t) \simeq 5\Lambda(\mathsf{U}_t)$.*

Theorem 4.2 gives an upper bound for $\lambda_t(k)$. Also, note that using the same kind of computations, we may obtain a bound for $\nu_t(k)$ (without the extra vertex). Hence, we summarize these results as follows.

THEOREM 4.5.

(a) $\lambda_t(k) \le \frac{(t+1)!}{2} \left(\sum_{j=1}^{t-1} \frac{1+\delta_{j+1}}{j!} + 1 \right) \qquad (t > 1,\ k \ge \binom{t+2}{2} + 1).$

(b) $\nu_t(k) \le k - k_{min} + (t+1)! \left(\sum_{j=1}^{t-1} \frac{j^2 + j + 6}{(j+2)!} + 5 \right) \quad (t > 1,\ k \ge k_{min} = \frac{t^2+t+6}{2}).$

5. Applications. Critical sets of Latin squares have been studied by Cooper, Donovan, Seberry, and others (see, e.g., [8]). Also, the concept has been generalized as *defining sets* for other combinatorial structures such as designs [24] and vertex colorings [21]. In [9] it has been shown that the concept of a critical set for latin squares can be used in the construction of secret sharing schemes, where it is quite important to investigate the class of critical sets (or defining sets) of minimal size.

However, in this case, it is not quite easy to introduce an effective method of design for secret sharing schemes which are based on Latin squares, since the class of easily constructible Latin squares is not as large as needed. This approach can be generalized as follows.

Let $\mathbf{C}$ be the class of all combinatorial structures of a specified type (such as Latin squares, designs, etc.), and let $C \in \mathbf{C}$ be of size n with a minimal defining set of size s, which means that a subset of size s in C uniquely specifies C in $\mathbf{C}$. Now, if elements of size n form a relatively large subset of $\mathbf{C}$ and $n - s$ is large enough, then we may construct a secret sharing scheme by distributing the defining set among shareholders and making n public. Note that in order to have a good secret sharing scheme we must also assume that each subset with less than s elements extends to a relatively large number of elements of $\mathbf{C}$.

It is clear that forcing structures can be used to construct graphs with restricted types of colorings, which may be used to design different types of secret sharing schemes. To illustrate the technique we propose a method to construct such graphs.

Let G be a k-UCG with a forcing structure. Then it is clear that the smallest defining set for its vertex coloring is of size $k-1$. Choose a vertex v from the forcing structure and delete it along with all of its edges. The new graph has a relatively large number of k-colorings because of the nature of k-UCG's, even if the vertex has the smallest degree. Now note that if one knows the colors of all vertices adjacent to v in the unique coloring of G, then these colors extend to a unique coloring for $G-v$. Therefore, all vertices adjacent to v, $N(v)$, along with their specific colors form a defining set for this special coloring. Also, note that it is hard to find this special k-coloring if one does not know about the forcing structure. Moreover, in order to enhance the scheme, one can build a large k-UCG and delete more than one vertex. This will give rise to a larger number of colorings for the new graph, which leads to a more secure secret sharing scheme.

This procedure can be generalized by applying forcing structures to construct graphs with special types of colorings. Then if the graph is large enough, our knowledge about the forcing structure helps to find the coloring quite easily; however, for someone who does not know about this structure it is quite hard to find that special coloring. This may be used in different cryptographic applications by keeping the coloring secret while making the graph public. Then, since the number of colored graphs on n vertices grows exponentially with n, this procedure introduces a one-way bottleneck which has many cryptographic applications.

On the other hand, if one looks for some algorithmic approach to design UCG's by means of forcing structures, the subject is probably applicable in data compression if one is able to code the data in terms of the coloring. This, of course, is still far from real applications since we do not know enough about different algorithms which may classify different UCG's, but in [11] some primary steps toward this idea are taken.

6. Concluding remarks. In this paper cliques in k-UCG's are investigated. Considering the parameter $\Lambda(G) = |E(G)| - e^*(G)$, we introduced the function

$$\lambda_t(k) = \min\{\Lambda(G) \ : \ G \text{ is a } k\text{-UCG and } cl(G) = k - t\},$$

and we obtained some upper bounds for this function using a technique called *forcing*. In this regard, for each $t \geq 1$ and for k large enough, we constructed a family of k-UCG's with maximum clique number $k - t$.

The first interesting aspect of this study is the case of small k's. As it is clear from our construction, q is not a constant, and consequently there is a big difference

between k and $2t$ when t gets large. It seems that it is quite difficult to build a k-UCG with $cl(G) \leq \lfloor \frac{k+1}{2} \rfloor$ and small Λ, and to construct such graphs seems to require a large jump in the number of vertices. For the small cases we explicitly ask for the following.

- Does there exist a 5-UCG, G, with $|V(G)| = 10$, $cl(G) = 4$ and $\Lambda(G) = 1$? (Note that the graph introduced in [25] has $\Lambda = 4$.)
- Find a 3-UCG without any triangle that can be used in a generalized version of our construction.
- Construct a 4-UCG with maximum clique number 2 and small Λ.
- Construct a 5-UCG with maximum clique number 3 and small Λ.

Also, we would like to note that the case of 3-UCG's is quite different from the case of k-UCG's for $k > 3$, which is mainly due to the fact that a 2-clique is nothing but an edge. Therefore, one is restricted to use only pure symmetric structures to reduce the clique. This method of construction is quite interesting, since one is actually using all the power of each vertex.

Again, by considering our construction in section 3 it is clear that the construction is more or less local, which means that we tried to build k-UCG's by using such graphs for smaller k's and a cyclic symmetry. This suggests that it might be possible to build k-UCG's by applying pure symmetric structures with a relatively small clique number and small Λ. Of course, if this is possible, then the graph will have a large number of vertices. Also, note that the structure of such graphs is very much related to the parameters k and t (e.g., note the parity of t in our construction). For instance, in this regard one may ask the following:

- Is it true that both $\lambda_t(k)$ and $\nu_t(k)$ grow exponentially with t?

Therefore, as far as Xu's conjecture is concerned, counterexamples may exist for large values of these parameters. In other words, it seems that there is a tradeoff between the number of vertices on one hand and Λ and the clique number on the other.

We note that in [15] it is proved that verification of Xu's conjecture may be reduced to the verification of the conjecture for minimal cores or to the verification of the conjecture for minimal k-UCG's on $2k$ vertices. Also, it is proved that for any minimal k-UCG on $2k$-vertices either the conjecture is true or there exists a vertex which is connected to all other vertices of the graph. On the other hand, it is proved that the conjecture is true for any k-UCG whose core is 4-chromatic and has 9 vertices.

All the same, the author believes it is quite probable that Xu's conjecture is false as it is stated; however, it seems that there exists a bound for the number of vertices, such as $V(k) > 2k$, depending on the chromatic number $k > 2$, such that the conjecture is true for all minimal k-cores with less than $V(k)$ vertices. (For definitions see [15].)

To sum up, it is clear that the study of k-UCG's is tightly related to the study of symmetric constructions on vertices and their colorings, which seems to need a combination of combinatorial and algebraic methods.

Acknowledgments. It is a pleasure for the author to thank the anonymous referees for their invaluable comments and to thank professor E. S. Mahmoodian who introduced to him the subject of defining sets for combinatorial structures and uniquely colorable graphs. Also, he is much obliged for the financial support of the Institute for Studies in Theoretical Physics and Mathematics (IPM) and the Research Council of the Sharif University of Technology.

REFERENCES

[1] V. A. Aksionov, *On uniquely* 3*-colorable planar graphs*, Discrete Math., 20 (1977), pp. 209–216.

[2] B. Bollobás, *Extremal Graph Theory*, Academic Press, New York, 1978.

[3] B. Bollobás, *Uniquely colorable graphs*, J. Combin. Theory Ser. B, 25 (1978), pp. 54–61.

[4] B. Bollobás and N. Sauer, *Uniquely colorable graphs with large girth*, Canad. J. Math., 28 (1976), pp. 1340–1344.

[5] M. Borowiecki and E. D. Burchardt, *Classes of chromatically unique graphs*, Discrete Math., 111 (1993), pp. 71–75.

[6] C. Y. Chao and Z. Chen, *On uniquely* 3*-colorable graphs*, Discrete Math., 112 (1993), pp. 21–27.

[7] G. Chartrand and D. P. Geller, *On uniquely colorable planar graphs*, J. Combinatorial Theory, 6 (1969), pp. 271–278.

[8] J. Cooper, D. Donovan, and J. Seberry, *Latin squares and critical sets of minimal size*, Australas. J. Combin., 4 (1991), pp. 113–120.

[9] J. Cooper, D. Donovan, and J. Seberry, *Secret sharing schemes arising from Latin squares*, Bull. Inst. Combin. Appl., 12 (1994), pp. 33–43.

[10] A. Daneshgar, *Forcing Structures and Cliques in Uniquely Vertex Colorable Graphs*, Tech. Rep. 97–209, Institute for Studies in Theoretical Physics and Mathematics (IPM), Tehran, Iran, 1997.

[11] A. Daneshgar, *Forcing and Graph Colorings*, Tech. Rep. 98–292, Institute for Studies in Theoretical Physics and Mathematics (IPM), Tehran, Iran, 1998.

[12] A. Daneshgar, *Private communication*, 1998.

[13] A. Daneshgar, *On* r*–type constructions and* Δ*–color–critical graphs*, J. Combin. Math. Combin. Comput., 29 (1999), pp. 183–206.

[14] A. Daneshgar, A. J. W. Hilton, and P. D. Johnson, *Relations among the fractional chromatic, choice, Hall, and Hall–condition numbers of simple graphs*, Discrete Math., to appear.

[15] A. Daneshgar and R. Naserasr, *On small uniquely–vertex–colorable graphs and Xu's conjecture*, Discrete Math., 223 (2000), pp. 93–108.

[16] A. Daneshgar and R. Naserasr, *On some parameters related to uniquely vertex–colorable graphs and defining sets*, submitted.

[17] P. Erdös and J. Spencer, *Probabilistic Methods in Combinatorics*, Probab. Math. Statist. 17, Academic Press, New York, Akadémiai Kiadó, London, 1974.

[18] F. Harary, *Graph Theory*, Addison-Wesley, Reading, MA, 1969.

[19] F. Harary, S. T. Hedetniemi, and R. W. Robinson, *Uniquely colorable graphs*, J. Combinatorial Theory, 6 (1969), pp. 264–270.

[20] L. Lovász, *On chromatic number of finite set-systems*, Acta Math. Acad. Scient. Hungar., 19 (1968), pp. 59–67.

[21] E. S. Mahmoodian, R. Naserasr, and M. Zaker, *Defining sets in vertex colorings of graphs and latin rectangles*, Discrete Math., 167/168 (1997), pp. 451–460.

[22] V. Müller, *On colorings of graphs without short cycles*, Discrete Math., 26 (1979), pp. 165–176.

[23] L. J. Osterweil, *Some classes of uniquely* 3*-colorable graphs*, Discrete Math., 8 (1974), pp. 59–69.

[24] A. P. Street, *Defining sets for block designs: An update*, in Combinatorics Advances, C. J. Colbourn and E. S. Mahmoodian, eds., Math. Appl. 329, Kluwer Academic Publishers, Dordrecht, 1995, pp. 307–320.

[25] M. Truszczyński, *Some results on uniquely colorable graphs*, in Finite and Infinite Sets, Colloq. Math. Soc. Janos Bolyai 37, North-Holland, Amsterdam, 1984, pp. 733–748.

[26] S. J. Xu, *The size of uniquely colorable graphs*, J. Combin. Theory Ser. B, 50 (1990), pp. 319–320.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 446–457

BLOCKING SEMIOVALS OF TYPE $(1, M+1, N+1)^*$

LYNN M. BATTEN† AND JEREMY M. DOVER‡

Abstract. We consider the existence of blocking semiovals in finite projective planes which have intersection sizes $1, m+1$ or $n+1$ with the lines of the plane for $1 \leq m < n$. For those prime powers $q \leq 1024$, in almost all cases, we are able to show that, apart from a trivial example, no such blocking semioval exists in a projective plane of order q. We are also able to prove, for general q, that if q^2+q+1 is a prime or three times a prime, then only the same trivial example can exist in a projective plane of order q.

Key words. projective planes, blocking sets, semiovals

AMS subject classifications. 51E20, 51E21

PII. S0895480100338002

1. Motivation. Blocking sets in projective planes have been much studied; the "classical" results due to Bruen [7], [8] state that in a projective plane of order q, a blocking set has between $q+\sqrt{q}+1$ and $q^2-\sqrt{q}$ points. Many additional references, as well as descriptions of applications in game theory and cryptography, can be found in Chapter 8 of Batten [2].

A *semioval* in a projective plane is a set of points S such that for each point P of S, there exists a unique line which meets S in exactly the point P. In [15], Hubaut proved that in a projective plane of order q, a semioval S has between $q+1$ and $q\sqrt{q}+1$ points. These two extremes occur in the case when S is an oval (see [2]) or a unital (see [11]), respectively. In the case of *regular* semiovals, that is, when S has constant line size a, considered as a design in its own right, Blokhuis and Szönyi [5] prove that either S is an oval or $a|q-1$.

Blocking sets and semiovals coincide in the case when each is a unital. In fact, for *minimal* blocking sets (where each point is on at least one tangent), it is known that the upper bound on the number of points is $q\sqrt{q}+1$, which is precisely the unital case (see [9]).

This leads to the more general question: in what other cases is a blocking set also a semioval? There is a trivial example on $3(q-1)$ points in every finite projective plane of order $q > 2$. Take three nonconcurrent lines (a "triangle") and delete the three points where these lines intersect (the "vertices"). It is not difficult to check that this set is a blocking semioval.

As well as being interesting objects in their own right, our main motivation for their study comes from Batten [3], where blocking semiovals are studied in relation to a cryptographic protocol designed by the author.

We say that a set X of points in a plane Π is of *type* $(m_1, m_2, \ldots, m_k)$ if each line of Π meets X in m_i points for some i, $1 \leq i \leq k$, and if for each m_i, $1 \leq i \leq k$, some line of Π meets X in m_i points. A unital thus has type $(1, \sqrt{q}+1)$ and the triangle

*Received by the editors July 24, 2000; accepted for publication (in revised form) June 4, 2001; published electronically October 4, 2001.

http://www.siam.org/journals/sidma/14-4/33800.html

†School of Computing and Mathematics, Deakin University, Clayton, Vic 3168, Australia (lmbatten@deakin.edu.au). The research of this author was partially supported by NSERC grant OGP0045831.

‡Department of Mathematics, North Dakota State University, Fargo, ND 58105-5075 (ajdover@aol.com). The research of this author was partially supported by NSF grant OSR-9452892.

with deleted vertices has type $(1, 3, q-1)$. In case $q-1=3$, it is easy to see that these two coincide.

It is not difficult, using the methods of Proposition 2.1 of the next section, to show that a blocking semioval of type $(1, n)$, $n \geq 2$, must be a unital. This result also follows from the deeper work of Tallini-Scafati [20] on the classification of sets of type $(1, n)$. The purpose of this paper is to explore the situation of type $(1, m+1, n+1)$ blocking semiovals. (We use $m+1$ and $n+1$ to facilitate simpler computations.)

In section 2, a number of arithmetic conditions on blocking semiovals of type $(1, m+1, n+1)$ are given and families of possible parameters exhibited. The principal result in this direction is the following theorem.

THEOREM 2.3. *Let $q > 4$ be a square prime power, and let Π be a projective plane of order q. Then a blocking semioval of type $(1, \sqrt{q}-(1+\lambda), \sqrt{q}+1)$ and size $(q+\sqrt{q}+1)(\sqrt{q}-(1+\lambda))$ is arithmetically feasible in Π if and only if $\frac{\lambda}{\lambda+2}(q+\sqrt{q})$ is an integer with $0 \leq \lambda \leq \sqrt{q}-3$.*

Using MAGMA [10], we were able to show that for q a prime power less than or equal to 1024, there is only a small number of possible blocking semiovals of our type whose existence remains undecided; this is the content of section 3. Other than the triangles with deleted vertices, we know of only one type of blocking semioval with just three intersection numbers. For any Singer cycle σ of $PG(2, 7)$, the three point orbits under σ^3 are each a blocking semioval of type $(1, 3, 4)$, each containing 19 points. (This set was originally considered by Brouwer [6] in another context.)

The main results of nonexistence are presented in section 4 with the following two theorems.

THEOREM 4.1. *Let Π be a projective plane of order $q \geq 2$ such that q^2+q+1 is prime. Then the only blocking semioval of type $(1, m+1, n+1)$, $1 \leq m < n$, in Π is a triangle with vertices deleted.*

THEOREM 4.2. *Let Π be a projective plane of order $q \geq 2$, $q \neq 7$, such that $q^2+q+1 = 3p$, p prime. Then the only blocking semioval of type $(1, m+1, n+1)$, $1 \leq m < n$, in Π is a triangle with vertices deleted.*

The "unique tangent" condition ascribed to semiovals has been generalized to the concept of a "strong representative system." Blokhuis and Metsch [4], for instance, use this setting to show that any semioval or minimal blocking set on $q\sqrt{q}$ points for q square and $q \geq 49$ must be part of a unital. Hence no minimal blocking set of this size exists. We discuss this, as well as some of their other results, further in section 4.

In section 5, we summarize our results and pose several conjectures.

2. Arithmetic conditions. In this section, we begin with a lemma which describes the various arithmetic conditions which constrain the parameters of our semiovals for which we need some notation. Let Π be a projective plane of order q, and let S be a blocking semioval with three intersection numbers in Π. Let v denote the number of points in S, and let $m+1$ and $n+1$ denote the nontangent line intersection sizes of S, where we may assume without loss of generality that $m < n$. As every point of S lies on exactly one tangent, it is a simple computation to show that there exist constants a and b such that every point of S lies on exactly b $(m+1)$-secants and a $(n+1)$-secants. The numbers $(v, m+1, n+1, a, b)$ are called the *parameters* of the blocking semioval S.

We can now prove the following proposition.

PROPOSITION 2.1. *Let Π be a projective plane of order $q > 2$, and let S be a blocking semioval in Π with parameters $(v, m+1, n+1, a, b)$. Then the following*

conditions hold:

$$v = 1 + an + bm, \tag{1}$$

$$q^2 + q + 1 = v\left(1 + \frac{b}{m+1} + \frac{a}{n+1}\right), \tag{2}$$

$$a + b = q, \tag{3}$$

$$m < \sqrt{q}, \tag{4}$$

$$v \geq (m+1)(n+1), \tag{5}$$

$$m + n \leq q. \tag{6}$$

Further, equality holds in inequalities (5) *and* (6) *if and only if* S *is a triangle with vertices removed.*

Proof. Equation (1) can be obtained by counting the number of points in S in two different ways, and (3) arises from counting the number of lines through a point of S.

To obtain (2), notice that every line must be either a tangent, $(m+1)$-secant, or $(n+1)$-secant to S. One can easily count that there are v tangents, $\frac{vb}{m+1}$ $(m+1)$-secants, and $\frac{va}{n+1}$ $(n+1)$-secants to S. The sum of these three numbers must equal the number of lines in the plane $q^2 + q + 1$, which establishes the equality.

Inequality (4) can be proven by contradiction. Suppose $m \geq \sqrt{q}$. As $n > m$, we know $n > \sqrt{q}$ as well. Using our second condition, we have $v = 1 + an + bm > 1 + a\sqrt{q} + b\sqrt{q}$. This latter expression equals $q\sqrt{q} + 1$ using our first condition. However, no semioval may contain more than $q\sqrt{q}+1$ points (see Hubaut [15]), which is our contradiction.

To establish inequality (5), we proceed by assuming $v \leq (m+1)(n+1)$. We compute

$$\begin{aligned} v &\leq (m+1)(n+1), \\ an + (q-a)m &\leq mn + m + n \text{ (using (1) and (3))} \\ (a-1)(n-m) + (q-2)m &\leq mn. \end{aligned}$$

As $a \geq 1$ and $n > m$, we know the term $(a-1)(n-m)$ is nonnegative, which implies $(q-2)m \leq mn$, with equality if and only if $a = 1$. This forces $n \geq q-2$, again with equality if and only if $a = 1$.

If $a > 1$, then $n > q-2$, which forces some line to meet S in at least q points. From Dover [12], this only can happen in $PG(2,3)$, and in that one case, $v = (m+1)(n+1)$. However, if $a = 1$, this quickly forces $b = q-1$ and $n = q-2$. Using (1) and (2), one can solve for m to find $m = 3$, which forces $v = 3q-3$. Again from Dover [12], any semioval with $3q-3$ points such that some line meets it in $q-1$ points must be a triangle with deleted vertices.

To prove inequality (6), begin with (2), clear denominators, and subtract $v(m+1)(n+1)$ from both sides to get

$$(q^2 + q + 1 - v)(m+1)(n+1) = v(bn + am + a + b). \tag{7}$$

Using (3) directly and in conjunction with (1) in

$$\begin{aligned} bn + am &= (a+b)n + (a+b)m - (an + bm) \\ &= q(n+m) - (v-1), \end{aligned}$$

we can substitute into (7) to obtain

$$(q^2+q+1-v)(m+1)(n+1) = (q(n+m+1)+1-v)v. \tag{8}$$

By inequality (5), we have $v \geq (m+1)(n+1)$ with equality if and only if S is a triangle. Using this fact on the right-hand side of (8) and cancelling we find

$$q^2+q+1-v \leq q(n+m+1)+1-v,$$

which implies $m+n \geq q$, with equality if and only if S is a triangle, as claimed. □

We call any set of parameters $(v, m+1, n+1, a, b)$ which satisfy the conditions of Proposition 2.1 *arithmetically feasible*. We now wish to give some examples of arithmetically feasible parameter sets.

PROPOSITION 2.2. *For any prime power $q \geq 5$, the parameter set of the triangle with deleted vertices, $(3q-3, 3, q-1, 1, q-1)$, is always arithmetically feasible. Further, for all planes of order $q \geq 5$ such a blocking semioval exists, and any blocking semioval with these parameters must be a triangle with deleted vertices.*

Proof. The proof follows directly from the existence of vertexless triangles and Proposition 2.1. □

We note here that the triangle forms a blocking semioval in all planes of order $q > 2$, yet we did not include the cases $q = 3, 4$ in Proposition 2.2. The reason is that if $q = 3$, this would force m to be greater than n, contrary to our assumption that $m < n$. In the case $q = 4$, $m = n$ and our blocking semioval has only one nontangent intersection number, not two. As mentioned in the introduction, this forces the vertexless triangle to be a unital when $q = 4$.

We now give a result which describes a family of feasible parameters for every $q > 2$ of square prime power order. Unlike Proposition 2.2, we know of no semioval with these parameters which exists.

THEOREM 2.3. *Let $q > 4$ be a square prime power, and let Π be a projective plane of order q. The parameter set*

$$\begin{aligned} v &= (q+\sqrt{q}+1)(\sqrt{q}-(1+\lambda)),\\ m+1 &= (\sqrt{q}-(1+\lambda)),\\ n+1 &= \sqrt{q}+1,\\ a &= q-(1+\mu),\\ b &= 1+\mu \end{aligned}$$

is arithmetically feasible if and only if $\mu = \frac{\lambda}{\lambda+2}(q+\sqrt{q})$ is an integer with $0 \leq \lambda \leq \sqrt{q}-3$.

Proof. Checking the arithmetic conditions of Proposition 2.1 for these parameters is tedious but straightforward, and thus it is left to the reader. □

COROLLARY 2.4. *The parameter sets $(q\sqrt{q}-1, \sqrt{q}-1, \sqrt{q}+1, q-1, 1)$ and $\left((q+\sqrt{q}+1)(\sqrt{q}-3), \sqrt{q}-3, \sqrt{q}+1, q-\left(1+\frac{1}{2}(q+\sqrt{q})\right), 1+\frac{1}{2}(q+\sqrt{q})\right)$ are arithmetically feasible for all square prime powers $q \geq 25$ with the first parameter set being arithmetically feasible for all $q \geq 9$.*

Proof. The first set corresponds to the trivial case $\lambda = 0$, while the second set corresponds to $\lambda = 2$, which requires $\sqrt{q} \geq 5$, as $\lambda \leq \sqrt{q}-3$. □

As mentioned previously, it is unknown if blocking semiovals with the second parameter set exist. Blokhuis and Metsch [4] have shown that no blocking semioval with the parameter set in Theorem 2.3 with $\lambda = 0$ can exist in $PG(2, q)$ when q is odd, and Ball [1] has proven a similar result for even q.

TABLE 1
Sporadic arithmetically feasible parameter sets.

Order of plane	v	$m+1$	$n+1$	a	b	Exists?
$q=7$	19	3	4	4	3	Yes by [6]
$q=16$	49	3	7	4	12	?
$q=25$	56	2	8	5	20	?
$q=64$	209	3	11	10	54	No by Prop. 3.2
$q=121$	1134	10	54	1	120	No by Prop. 3.1
	518	2	6	99	22	?
	342	2	6	55	66	?
$q=191$	1612	7	12	93	98	?
$q=263$	4251	17	18	42	221	?
$q=343$	3774	10	17	98	245	No by Prop. 3.2
$q=373$	7154	20	22	33	340	?
$q=947$	19390	18	25	470	477	?
$q=1024$	5889	5	13	224	800	?
	11585	5	13	936	88	?
	11585	11	35	56	968	No by Prop. 3.2

Table 1 details all arithmetically feasible parameters of all prime powers up to and including 1024, excepting those parameters shown to be feasible in Proposition 2.2 and Theorem 2.3. For omitted orders, the parameters given by the applicable results above are the only possibilities.

An exhaustive computer search using the package MAGMA [10] was used to obtain these possibilities for those values of q which could not be eliminated using the results of the next section. We note that inequality (4) was used strongly in this search to limit the possibilities for m.

3. Eliminating possible parameter sets. In this section we prove several results which will allow us to rule out a number of the possibilities in Table 1. The first deals with the case where $a = 1$.

PROPOSITION 3.1. *Let Π be a projective plane of order q, and let S be a blocking semioval in Π with parameters $(v, m+1, n+1, 1, q-1)$. Then m must divide $q-n$ and $v-(q+1)$.*

Proof. First suppose that every point P off S lies on an $(n+1)$-secant. Since there are precisely $\frac{v}{n+1}$ $(n+1)$-secants to S, there are at most $\frac{v}{n+1}(q-n)$ points outside of S. Therefore $q^2+q+1-v \leq \frac{v}{n+1}(q-n)$, which we can rearrange to obtain $q^2+q+1 \leq \frac{v(q+1)}{n+1}$.

Solving this inequality for v, we find that $v \geq \frac{(n+1)(q^2+q+1)}{q+1}$, which we can divide through to get $v > q(n+1)$. However, again from (1) we have $v = n+1+(q-1)m$, which implies $(q-1)m > (q-1)(n+1)$, contradicting the fact that $m < n$.

Hence there must exist a point P which lies on no $(n+1)$-secants. Suppose P lies on x tangents and y $(m+1)$-secants. Then simple counting yields that $x+y = q+1$ and $x+(m+1)y = v$. Subtracting these yields $my = v-(q+1)$. Using (1) to substitute in for v yields $my = 1+n+(q-1)m-q-1$. Rearranging gives us $m(y-q+1) = n-q$, which implies that m divides $q-n$. That m divides $v-(q+1)$ now quickly follows from (1). ∎

In particular, this proposition rules out the possibility that a blocking semioval in a plane of order 121 with parameters $(1134, 10, 54, 1, 120)$ could exist (see Table 1), as 9 does not divide $121-53 = 68$.

Up to this point, we have focused on the "internal" structure of a blocking semi-

oval. We would now like to prove a result concerning points outside the semioval; this will give us a strong divisibility condition amongst our parameters.

PROPOSITION 3.2. *Let Π be a projective plane of order q, and let S be a blocking semioval in Π with parameters $(v, m+1, n+1, a, b)$. For every point P outside of S, let $d(P)$ denote the number of tangents to S passing through P, $e(P)$ the number of $(m+1)$-secants, and $f(P)$ the number of $(n+1)$-secants. Then we have*

$$\begin{aligned} nd(P) &\equiv q+n \pmod{n-m},\\ me(P) &\equiv v-(q+1) \pmod{n},\\ nf(P) &\equiv v-(q+1) \pmod{m}. \end{aligned}$$

Proof. Counting lines through P, we have the relation $d(P)+e(P)+f(P) = q+1$. By counting pairs of points (P, Q), where $Q \in S$, we obtain $d(P) + (m+1)e(P) + (n+1)f(P) = v$.

Multiply the first relation through by n to obtain $nd(P) + ne(P) + nf(P) = n(q+1)$, which we can rewrite as $nd(P) + (n-m)e(P) + me(P) + nf(P) = n(q+1)$. Notice from our second relation that $me(P) + nf(P) = v - (d(P) + e(P) + f(P)) = v - (q+1)$. Hence we have $nd(P) + (n-m)e(P) = (n+1)(q+1) - v$. From (1) we know $v = 1 + an + bm$, from which we can get $v = 1 + (a+b)n + b(m-n)$. Putting this all together we obtain $nd(P)+(n-m)e(P) = (n+1)(q+1)-(1+qn-b(n-m))$. Finally reducing modulo $n-m$ we obtain $nd(P) \equiv q+n \pmod{n-m}$, as claimed.

To obtain the latter two relations in the proposition, we need recall only that $me(P) + nf(P) = v - (q+1)$, which we can successively reduce modulo m and n. □

To use this result effectively, we need the following two counts. Let S be a blocking semioval with parameters $(v, m+1, n+1, a, b)$ in a plane of order q. We first count all of the pairs (P, ℓ), where P is a point off of S and ℓ is a line through P which is tangent to S. On the one hand, there are v tangents to S, each of which contains q points off S, yielding qv pairs. On the other hand, for each point P off S, there are $d(P)$ (using the notation of Proposition 3.2) lines which can pair with it. Hence we have the equation $\sum_P d(P) = qv$, where the sum is taken over all points P off S. A similar count of triples (P, ℓ_1, ℓ_2), where ℓ_1 and ℓ_2 are distinct tangents to S meeting in P, yields the equation $\sum_P d(P)(d(P)-1) = v(v-1)$. From this we obtain $\sum_P d(P)^2 = v(q+v-1)$.

Let us look at the possible parameter set $(209, 3, 11, 10, 54)$ in a plane of order 64. By Proposition 3.2, we know that for any point P off S, we have $10d(P) \equiv 74 \pmod 8$, or $d(P) \equiv 1 \pmod 4$. In particular $d(P)$ cannot take on the values 2, 3, or 4.

Consider $\sum_P (d(P)-1)(d(P)-5)$. By the above paragraph, no term of this sum can be negative. Further, this sum must be divisible by 4^2. For this particular case, we have $\sum_P d(P) = 13376$ and $\sum_P d(P)^2 = 56848$. Thus we can evaluate $\sum_P (d(P)-1)(d(P)-5) = 56848 - 6(13376) + 5(64^2 + 64 + 1 - 209) = -19456$. This contradicts the fact that all of our summands are nonnegative, implying that no blocking semioval with these parameters can exist. A similar argument rules out the possible parameter set in a plane of order 343 and the third possibility for a plane of order 1024.

We performed a similar analysis for $e(P)$ and $f(P)$ but were unable to rule out any additional parameter sets. We are left with the unresolved sporadic cases in Table 2.

TABLE 2
Unresolved sporadic parameter sets.

Order of plane	v	$m+1$	$n+1$	a	b
$q=16$	49	3	7	4	12
$q=25$	56	2	8	5	20
$q=121$	518	2	6	99	22
	342	2	6	55	66
$q=191$	1612	7	12	93	98
$q=263$	4251	17	18	42	221
$q=373$	7154	20	22	33	340
$q=947$	19390	18	25	470	477
$q=1024$	5889	5	13	224	800
	11585	5	13	936	88

4. The p and $3p$ cases. The prime factorization of q^2+q+1 can be used in conjunction with (2) of Proposition 2.1 to yield some information about blocking semiovals in the plane. Thus, in this section, we consider several special cases of this factorization. Note that representations of q^2+q+1 have been much studied in number theory (see, for example, Mordell [18]). In particular, the cases where q^2+q+1 is a prime power or divisible by 3 have been given much attention [14], [17], [19].

The simplest case to tackle is q^2+q+1 prime, which occurs quite frequently for $q \leq 1024$.

THEOREM 4.1. *Let Π be a projective plane of order $q \geq 2$ such that q^2+q+1 is prime. Then the only blocking semioval of type $(1, m+1, n+1)$, $1 \leq m < n$, in Π is a triangle with vertices deleted.*

Proof. Let $p = q^2+q+1$ and S be a blocking semioval of type $(1, m+1, n+1)$, $1 \leq m < n$, on v points in Π. By (2) of Proposition 2.1, $p(m+1)(n+1) = v[(m+1)(n+1) + a(m+1) + b(n+1)]$. Since v must be less than p, $v \mid (m+1)(n+1)$. By inequality (5) of Proposition 2.1, S is a triangle with vertices deleted. □

Since approximately half of all values of q^2+q+1, q a prime power, are congruent to 0 modulo 3, it is worthwhile to examine the case when 3 divides the number of points in the plane. In case $q=7$, $q^2+q+1 = 57 = 3 \cdot 19$, and as we saw in the previous section, there is a semioval decomposition of the plane of the type wanted. We next show that for q^2+q+1 equal to 3 times a prime, the case $q=7$ is the only one in which a nontrivial semioval of our type can occur.

THEOREM 4.2. *Let Π be a projective plane of order $q \geq 2$, $q \neq 7$, such that $q^2+q+1 = 3p$, p prime. Then the only blocking semioval of type $(1, m+1, n+1)$, $1 \leq m < n$, in Π is a triangle with vertices deleted.*

Proof. For $q < 7$, the only value of q^2+q+1 of the desired form is 21 with $q=4$. By inequality (6) of Proposition 2.1, $v \leq 9$ in this case. Applying (2) and (3) of Proposition 2.1 and noting that the only possible values for m and n are 1 and 2, respectively, we find that there are no semiovals of type $(1,2,3)$ in this plane.

From now on, we assume $q \geq 8$. Proposition 2.1 yields

$$3p(m+1)(n+1) = v[(m+1)(n+1) + a(m+1) + b(n+1)]. \tag{9}$$

Now $3p > v \geq (m+1)(n+1)$ by inequality (5) of Proposition 2.1 implies $(3p, (m+1)(n+1)) = 1, 3$ or p. We consider several cases.

1. Suppose this is p. Then $p \mid (m+1)(n+1)$ and $p \leq (m+1)(n+1) < 3p$ implies $(m+1)(n+1) = p$ or $2p$.

(a) If $(m+1)(n+1) = p$, (9) implies $v \mid 3p^2$, and the only factors of v are 3 and p. Together with $v < 3p^2$, either $v = 3$, which is too small, or $v = p = (m+1)(n+1)$, which yields the triangle by Proposition 2.1.
(b) If $(m+1)(n+1) = 2p$, (9) implies $v \mid 6p^2$. Together with $(m+1)(n+1) = 2p \leq v < 3p$, we obtain $v = 2p = (m+1)(n+1)$ and thus we have the triangle.

2. Now suppose $(3p, (m+1)(n+1)) = 1$ or 3. Then from (9) we have $v \mid 3(m+1)(n+1)$ or $p \mid v$.
(a) Suppose $p \mid v$. Then $v = p$ or $2p$. In either situation, we get $p \leq q\sqrt{q}+1$, the maximum size of any semioval, while $3p = q^2 + q + 1 > 3q\sqrt{q} + 3$ yields a contradiction if $q \geq 8$.
(b) Suppose $v \mid 3(m+1)(n+1)$. Then $v = x(m+1)(n+1)$ where $x = 1, \frac{3}{2}$, or 3. The first case yields the triangle, so we consider separately the cases $v = \frac{3}{2}(m+1)(n+1)$ and $v = 3(m+1)(n+1)$.

First assume $v = \frac{3}{2}(m+1)(n+1)$. Applying Proposition 2.1 to (9), we obtain $2p = \frac{2}{3}(q^2+q+1) = q(m+n+1) + 1 - \frac{1}{2}(m+1)(n+1)$. This yields

$$(2q-1)^2 - (2q-1)(3m+3n-1) + 3mn + 1 = 0,$$

which has roots

$$2q - 1 = \frac{3m+3n-1 \pm \sqrt{(3m+3n-1)^2 - 4(3mn+1)}}{2}. \tag{10}$$

Now $(3m+3n-1)^2 - 4(3mn+1) > (3n-m+3)^2$ if $m \geq 2$, which gives either $4q - 2 > 3m + 3n - 1 + (3n - m + 3)$ or $4q - 2 < 3m + 3n - 1 - (3n - m + 3)$. The second of these yields, using inequality (4) of Proposition 2.1, $4q < 4m - 2 < 4\sqrt{q} - 2$, which is false for $q \geq 2$. From the first of these, we obtain $4q > 6n + 2m + 4$. On the other hand, using (1) and (3) of Proposition 2.1, and assuming $b \leq q - 2$, we obtain $q \leq 3(n+1)/2 - (n-3)/2m + 2$. Putting inequalities together, we get $6n + 2m + 6 \leq 4q \leq 6n + 6 - 2(n-3)/m + 8$; hence $2(n-3)/m + 2m \leq 8$, which is false for $n \geq 3$ and $m > 4$. It remains, for this value of v, to consider the separate cases $b = q - 1$, $n = 2$, $m = 1$, and $n \geq 3$ while $m = 2, 3$, or 4.

If $m = 1$, substituting in (10) yields $4q - 2 = 2$ or $6n + 2$. Only $q = (3n+2)/2$ is possible; so $n = 2(q-1)/3$, and $v = 2q + 1$. No blocking semioval of this size can exist for $q \geq 7$ by Dover [13]. n being an integer forces $q \equiv 1 \pmod 3$, so the only remaining q for which n is an integer is 4, and this yields the vertexless triangle.

If $n = 2$, (10) yields $4q - 2 = 3m + 5 \pm \sqrt{(3m+1)^2 + 20}$. The discriminant is a square only if $m = 1$ (implying $q = 4$), and in this case, we obtain the triangle with deleted vertices. The cases $m = 2, 3$, and 4 can be eliminated in the same way. For instance, $m = 4$ yields $(3n+3)^2 + 108$ a square, implying that 108 factors as $(x+y)(x-y)$, where $y = 3n + 3$. This is not possible for n an integer larger than 1. Finally, suppose $a = 1$, $b = q - 1$. By Proposition 3.1, $m \mid v - (q+1)$, which implies $2m \mid 3n - 2q + 1$. So $2m \mid 2q(3n - 2q + 1)$. However, from (9), using $3p = q^2 + q + 1$, we get $2(q^2+q+1) = 3(m+1)(n+2) + 3(q-1)(n+1)$, from which $m \mid 2q^2 - q - 3qn - 1$. It follows that $2m \mid 4q^2 - 2q - 6qn - 2 + 2q(3n - 2q + 1) = -2$. The situation $m = 1$ was dealt with above.

In order to eliminate the possibility that $v = 3(m+1)(n+1)$, we first show that it implies $b = q - 2$, or $n = (q-1)/3$ or $(q-1)/4$, or $m \leq 2$.

First suppose that $a = 1$. By Proposition 3.1, we must have $m \mid v - (q+1)$. As $v = 3(m+1)(n+1)$, this implies $m \mid 3n - q + 2$. However, from (9), we obtain

$3p = q^2+q+1 = 3[(m+1)(n+1)+(m+1)+(q-1)(n+1)] = 3m(n+2)+3+3qn+3q$ and so $(q-1)^2 = 3m(n+2)+3+3qn$. It follows that $m \mid (q-1)^2-3qn-3$. However, from above, $m \mid q-1-(3n+1)$ and therefore $m \mid (q-1)^2-(q-1)(3n+1)$. Thus $m \mid 3n-q+4$, finally yielding $m \mid 2$, which forces $m \leq 2$.

Now assuming that $a \geq 2$, we have $b \leq q-2$. If $b = q-2$, we are done. So suppose $b \leq q-3$. Using (1) of Proposition 2.1 yields $qn = b(n-m)+3(m+1)(n+1)-1 \leq (q-3)(n-m)+3(mn+n+m)+2$, implying $mq \leq 3mn+6m+2$, and so $q \leq 3n+6+\frac{2}{m}$. As we may assume $m > 2$, then $q \leq 3n+6$, or $n \geq (q-6)/3$. We proceed to determine precisely the possible values for n.

Using (2) of Proposition 2.1, substituting for v, and applying (3) of the proposition, we obtain $3(m+1)(n+1)+3(bn+am) = (q-1)^2$. Using (1) and (3) then yields

$$(q-1)^2-3(q-1)(n+m)+3(2mn+n+m+1) = 0, \tag{11}$$

a quadratic in $q-1$. Therefore

$$q-1 = \frac{3(n+m) \pm \sqrt{9n^2+9m^2-6mn-12n-12m-12}}{2}.$$

(Note that (11) is independent of assumptions on m or n.) Since $n > m > 2$, and using inequality (4) of Proposition 2.1, it follows that $q-1 > [3(n+m)+\sqrt{9n^2+9m^2-18mn-12n+12m+4}]/2 = (3(n+m)+(3n-3m-2))/2 = 3n-1$, or $q-1 < 3m+1 < 3\sqrt{q}+1$. In this latter case, $q \leq 12$. However, $3 \mid q^2+q+1$ implies $q \neq 8, 9, 11, 12$, and no projective plane of order 10 exists (see Lam, Thiel, and Swiercz [16]). Thus this case is eliminated. Consequently, $n < q/3$. Again, $q \equiv 1 (\mathrm{mod}\, 3)$, since $3 \mid q^2+q+1$, and thus $(q-6)/3 \leq n < q/3$ implies $3n = q-1$ or $q-4$.

We proceed to eliminate each of the above cases.

Suppose $m = 1$. Substituting in (11) gives $(q-1)^2-3(q-1)(n+1)+3(3n+2) = 0$, so $(q-1)^2-3(q-1)(n+1)+9(n+1) = 3$. The fact that $3 \mid q^2+q+1$ again gives $3 \mid q-1$, and so 9 divides the left-hand side but not the right, a contradiction.

Suppose $m = 2$. Using (11) again yields $3n = [(q-6)(q-2)+4]/(q-6) = q-2+\frac{4}{q-6}$. Since this must be an integer, $q \geq 8$ and the fact that no projective plane of order 10 exists give a contradiction.

Suppose $3n = q-1$. Substituting in (11) gives $9n^2-9n(n+m)+6mn+3n+3m+3 = 0$. This forces $n = 1+2/(m-1)$, which is not possible.

Suppose $3n = q-4$. So $q-1 = 3(n+1)$, and (11) yields the impossibility $n(m-4) = 2(2-m)$.

We finally consider the case $b = q-2$. Using (1) and (3) of Proposition 2.1, we get $a = 2$ and $v = 3(m+1)(n+1) = 1+2n+(q-2)m$, and it follows that $q = 3n+5+\frac{n+2}{m}$, whence $m \mid n+2$.

Fix a line of size $n+1$ in S. Each of its points is on a second line of this size since $a = 2$. Thus the number of lines of this size in S is at least $n+2$. However, counting the precise number in two ways produces $\frac{v \cdot 2}{n+1}$ lines of size $n+1$ or $6(m+1)$. Thus $n+2 \leq 6(m+1)$. If $m \geq 6$, then $n+2 \leq 7m$, and we may set $n+2 = xm$, $2 \leq x \leq 7$.

From above, $q = 3xm+x-1$. Substituting for q in (2) in Proposition 2.1 implies

$$(3xm+x-1)^2+3xm+x = 3(m+1)(xm-1)+3(3xm+x-3)(xm-1)+6(m+1).$$

This reduces to $3m(2mx-x^2-4x+1)+12 = (x+1)^2 \leq 64$, which implies $m(2mx-x^2-4x+1) \leq 17$. If $m \geq 6$, $2mx-x^2-4x+1 \leq 2$, or $x(2m-x-4) \leq 1$. It follows that $2m-x-4 < 0$, which contradicts $m \geq 6$.

It remains only to dispose of the cases $m = 3, 4, 5$. We return to the equation $3m(2mx - x^2 - 4x + 1) + 12 = (x + 1)^2$. For $m = 3$, this becomes $5x^2 - 8x - 10 = 0$, which implies $2 \mid x$ and then $4 \mid 10$, a contradiction. For $m = 4$, it becomes $13x^2 - 46x - 23 = 0$, implying $23 \mid x$ and then $(23)^2 \mid 23$, a contradiction. Finally, for $m = 5$, $8x^2 + 44x - 13 = 0$ implies the contradiction $2 \mid 3$. □

In attempting to generalize Theorem 4.2 to q^2+q+1, a product of distinct primes, we have had only partial success. We summarize this in the next result.

PROPOSITION 4.3. *Let* Π *be a projective plane of order* $q \geq 2$ *such that* $q^2+q+1 = p'p$, p', *and* p *both prime, with* $p' < p$. *Let* S *be a blocking semioval in* Π *of type* $(1, m+1, n+1)$. *Then* $p \not| (m+1)(n+1)$; *and if* $p' | (m+1)(n+1)$, *then* p' *divides both* $m+1$ *and* $n+1$, *or* $p' | a$ *or* $p' | b$.

Proof. If $p|(m+1)(n+1)$, then $p|m+1$ or $p|n+1$ while both of $m+1$ and $n+1$ are less than q, and p must be bigger than q.

Suppose $p'|(m+1)(n+1)$. Again, $p'|m+1$ or $p'|n+1$. If $p'|m+1$, set $m+1 = (p')^\alpha x$, $p' \not| x$, $\alpha \geq 1$. So by (2) of Proposition 2.1, $(p')^{\alpha+1}px(n+1) = v[(p')^\alpha x(n+1) + a(p')^\alpha x + b(n+1)]$, which implies $p'|vb(n+1)$. We may assume $p' \not| b$ and $p' \not| n+1$. Then $p'|v$. Set $v = (p')^\beta y$, $p' \not| y$, $\beta \geq 1$, and $y < p$. So $\alpha + 1 \geq \beta$ and $(p')^{\alpha-\beta+1}px(n+1) = y[(p')^\alpha x(n+1) + a(p')^\alpha x + b(n+1)]$. If $\alpha - \beta + 1 > 0$, then $p'|yb(n+1)$, a contradiction. So $\alpha - \beta + 1 = 0$. Then $px(n+1) = y[(m+1)(n+1) + a(m+1) + b(n+1)]$ and $y < p$ gives $x(n+1) > (n+1)(m+1+b) + a(m+1)$, so that $x > m+1+b$, which contradicts $x < m+1$.

If we now suppose that $p'|n+1$, the argument is completely analogous and introduces only the last possibility that $p'|a$. □

Before leaving this section, we make two observations: first, we look at the case where $m = 1$. Second, noting that no blocking semioval of our type can have size $q\sqrt{q} + 1$, we address the next possibility in a square order plane, i.e., $v = q\sqrt{q}$.

PROPOSITION 4.4. *Let* Π *be a projective plane of order* $q \geq 7$ *containing a blocking semioval* S *of type* $(1, 2, n+1)$ *with* $n > 1$. *Then the* $(n+1)$*-secants to* S *form a dual blocking set, and consequently there are between* $q + \sqrt{q} + 1$ *and* $q^2 - \sqrt{q}$ *of them.*

Proof. No point of Π exterior to S is only on tangents to S, as this would force S to have exactly $q + 1$ points, which is too small to be a blocking set by Bruen [8]. Nor is any such point only on 2-secants, as this would imply $v = 2q + 2$, giving by (2) of Proposition 2.1, $2(n+1)(q^2+q+1) = 2(q+1)[2(n+1) + 2a + b(n+1)]$. Since $(q^2+q+1, q+1) = 1$, we obtain $q+1|n+1$ while $n+1 \leq q-1$. Similarly, if an exterior point is only on $(n+1)$-secants, then $v = (n+1)(q+1)$, and this same equation results in $2(q^2+q+1) = (q+1)[2(n+1) + 2a + b(n+1)]$, a contradiction.

Let an exterior point be on x 2-secants and y tangents and assume it is on no $(n+1)$-secants. Then $2x + y = v$ and $x + y = q + 1$. So $x = v - (q+1)$ and $y = 2(q+1) - v \geq 0$. Dover [13] shows that any blocking semioval satisfies $v \geq 2(q+1)$ for $q \geq 7$. This forces us to have $v = 2q + 2$, which was eliminated in the previous paragraph.

Hence every point lies on at least one $(n+1)$-secant, and the first paragraph shows no point is only on $(n+1)$-secants. Therefore the $(n+1)$-secants form a dual blocking set and the result follows from Bruen [8]. □

Blokhuis and Metsch [4, Theorem 1.2] show that for $q \geq 49$ and square, a semioval on $q\sqrt{q}$ points must be part of a unital and hence cannot be a blocking set. Specializing to the case of semiovals of type $(1, m+1, n+1)$, we can generalize this result to the following proposition.

PROPOSITION 4.5. *Let* Π *be a projective plane of square order* $q \geq 2$. *Then* Π

contains no blocking semioval of type $(1, m+1, n+1)$, $1 \le m < n$, *on* $q\sqrt{q}$ *points.*

Proof. Applying (2) of Proposition 2.1, $(m+1)(n+1)(q^2+q+1) = q\sqrt{q}[(m+1)(n+1)+b(n+1)+a(m+1)]$ and $(q\sqrt{q}, q^2+q+1) = 1$ implies $q\sqrt{q}|(m+1)(n+1)$. However, $v = q\sqrt{q} \ge (m+1)(n+1)$ by inequality (5) of Proposition 2.1, and equality results in the triangle $m = 2$, $n = q-2$, and so $3(q-1) = q\sqrt{q}$ that yields $q|3$, which is impossible. □

Blokhuis and Metsch [4, Theorems 1.3 and 1.4] also consider $v = q\sqrt{q}-1$, proving that if the point/tangent incidences of S are the point/tangent incidences of a unital, and if $q \ge 25$, then S is indeed part of a unital or a minimal blocking set. If the plane is Desarguesian, only the former of these can hold.

5. Conclusion. We have given a number of conditions which constrain the possible parameters of a blocking semioval with three intersection numbers; while these conditions have eliminated many possibilities, the remaining cases seem very difficult to work with. Indeed, we conjecture that there are no blocking semiovals with three intersection numbers other than the triangle and the sporadic example when $q = 7$.

On a more optimistic note, we suspect that some of the remaining sporadic cases may be attackable. For instance, the parameters $(56, 2, 8, 5, 20)$ for a blocking semioval of our type in a plane of order 25 could be analyzed. Indeed, a cursory analysis shows that if such a semioval were to exist, it would imply the existence of a blocking set of size 35 and type $(1, 2, 5)$ in that plane. It is not known if such a set can exist.

One pattern which our data indicates is that in a projective plane of order q, where q^2+q+1 is the product of two distinct primes, the parameter set of the triangle is the only arithmetically feasible parameter set for that order. Proposition 4.3, summarizing our results in this direction, inclines us to believe this conjecture is true.

As a final comment, we note that if $q = 2^{2k+1}$ for $1 \le k \le 5$, the only arithmetically feasible parameter sets for a semioval of our type are those of the triangle. We conjecture that this is true for all $k \ge 1$.

Acknowledgment. The authors would like to thank the referee for many useful suggestions and particularly for suggesting the method of proof used in Proposition 3.2.

REFERENCES

[1] S. Ball, *Partial unitals and related structures in Desarguesian planes*, Des. Codes Cryptogr., 15 (1998), pp. 231–236.
[2] L. M. Batten, *Combinatorics of Finite Geometries*, 2nd ed., Cambridge University Press, Cambridge, New York, Melbourne, 1997.
[3] L. M. Batten, *Determining sets*, Australas. J. Combin., 22 (2000), pp. 167–176.
[4] A. Blokhuis and K. Metsch, *Large minimal blocking sets, strong representative systems, and partial unitals*, in Finite Geometry and Combinatorics, Cambridge University Press, Cambridge, New York, Melbourne, 1993, pp. 37–52.
[5] A. Blokhuis and T. Szönyi, *Note on the structure of semiovals in finite projective planes*, Discrete Math., 106/107 (1992), pp. 61–65.
[6] A. E. Brouwer, *A series of separable designs with application to pairwise orthogonal Latin squares*, European J. Combin., 1 (1980), pp. 39–41.
[7] A. A. Bruen, *Baer subplanes and blocking sets*, Bull. Amer. Math. Soc., 76 (1970), pp. 342–344.
[8] A. Bruen, *Blocking sets in finite projective planes*, SIAM J. Appl. Math., 21 (1971), pp. 380–392.
[9] A. A. Bruen and J. A. Thas, *Blocking sets*, Geom. Dedicata., 6 (1977), pp. 193–203.
[10] J. Cannon and C. Playoust, *An Introduction to MAGMA*, University of Sydney, Sydney, Australia, 1993.
[11] P. Dembowski, *Finite Geometries*, Springer-Verlag, New York, 1968.

[12] J. Dover, *Semiovals containing large collinear subsets*, J. Geom., 69 (2000), pp. 58–67.
[13] J. Dover, *A lower bound on blocking semiovals*, European J. Combin., 21 (2000), pp. 571–577.
[14] W. Feit, *Finite projective planes and a question about primes*, Proc. Amer. Math. Soc., 108 (1990), pp. 561–564.
[15] X. Hubaut, *Limitation du nombre de points d'un* (k, n)*-arc régulier d'un plan projectif fini*, Atti Accad. Naz. Lincei Rend. Cl. Sci. Fis. Mat. Natur. (8) 48, (1970), pp. 490–493.
[16] C. W. H. Lam, L. Thiel, and S. Swiercz, *The non-existence of finite projective planes of order* 10, Canad. J. Math., 41 (1991), pp. 1117–1123.
[17] W. Ljunggren, *Einige Bemerkungen über die Darstellung ganzer Zahlen durch binare kubische Formen mit positiver Diskriminante*, Acta Math., 75 (1942), pp. 1–21.
[18] L.J. Mordell, *Diophantine Equations*, Academic Press, London, New York, 1969.
[19] T. Nagell, *Des equations indetermineés* $x^2 + x + 1 = y^n$ *et* $x^2 + x + 1 = 3y^n$, Norsk. Mat. Forenings Skr. Ser. I, 2 (1921), pp. 12–14.
[20] M. Tallini-Scafati, *{k,n}-archi in un piano grafico finito, con particolare riguardo a quelli con due caratteri*, Atti Accad. Naz. Lincei Rend. Cl. Sci. Fis. Mat. Natur. (8), 40 (1966), pp. 1020–1025.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 458–470

AN OPTIMAL ONLINE ALGORITHM FOR BOUNDED SPACE VARIABLE-SIZED BIN PACKING*

STEVEN S. SEIDEN†

Abstract. An online algorithm for variable-sized bin packing, based on the HARMONIC algorithm of Lee and Lee [*J. ACM*, 32 (1985), pp. 562–572], is investigated. This algorithm was proposed by Csirik [*Acta Inform.*, 26 (1989), pp. 697–709], who proved that for all sets of bin sizes, 1.69103 upper bounds its performance ratio. The upper bound is improved in the sense that we give a method of calculating the performance ratio to any accuracy for any set of bin sizes. Further, it is shown that the algorithm is optimal among those which use bounded space. An interesting feature of the analysis is that, although it is shown that our algorithm achieves a performance ratio arbitrarily close to the optimum value, it is not known precisely what that value is. The case where bins of capacity 1 and $\alpha \in (0, 1)$ are used is studied in greater detail. It is shown that among algorithms which are allowed to choose α, the optimal performance ratio lies in $[1.37530, 1.37532]$.

Key words. bin packing, online algorithms, lower bounds

AMS subject classification. 68W40

PII. S0895480100369948

1. Introduction.

Bin packing is one of the oldest and most well studied problems in computer science [5, 3]. Ideas which originated in the study of the bin packing problem have helped shape computer science as we know it today. The idea of finding the best approximation algorithm for a problem has its origins in bin packing. The study of online algorithms also has its roots in the study of bin packing. In this paper, we investigate a natural generalization of the classical bin packing problem known as variable-sized bin packing.

In the *bin packing* problem, we receive a sequence σ of *pieces* $p_1, p_2, \ldots, p_N$. Each piece has a fixed *size* in $(0, 1]$. In a slight abuse of notation, we use p_ℓ to indicate both the ℓth piece and its size. We have an infinite number of *bins*, each with *capacity* 1. Each piece must be assigned to a bin. Further, the sum of the sizes of the pieces assigned to any bin may not exceed its capacity. A bin is *empty* if no piece is assigned to it; otherwise it is *used*. The goal is to minimize the number of bins used.

The *variable-sized bin packing* problem differs from the classical problem in that bins do not all have the same capacity. We are given real numbers $\alpha_1 < \alpha_2 < \cdots < \alpha_m = 1$, which are the allowed bin sizes. At the time a bin is opened, the algorithm may choose the bin's capacity. Now the goal is to minimize the sum of the capacities of the bins used.

In the *online* versions of these problems, each piece must be assigned in turn without knowledge of the next pieces. Since it is impossible, in general, to produce the best possible solution when computation occurs online, we consider approximation algorithms.

A bin packing algorithm uses *bounded space* if it has only a constant number of bins available to accept items at any point during processing. These bins are called *open* bins. Bins which have already accepted some items, but which the algorithm

*Received by the editors March 28, 2000; accepted for publication June 4, 2001; published electronically October 4, 2001. A preliminary version of this paper was presented at the 27th International Colloquium on Automata, Languages, and Programming, Geneva, Switzerland, 2000.
http://www.siam.org/journals/sidma/14-4/36994.html
†Department of Computer Science, 298 Coates Hall, Louisiana State University, Baton Rouge, LA 70803 (sseiden@acm.org).

no longer considers for packing, are *closed* bins. The bounded space assumption is a quite natural one, especially so in online bin packing. Essentially the bounded space restriction guarantees that output of packed bins is steady and that the packer does not accumulate an enormous backlog of bins which are output only at the end of processing.

The standard measure of algorithm quality for bin packing is the *asymptotic performance ratio*, which we now define. For a given input sequence σ, let $\text{cost}_{\mathcal{A}}(\sigma)$ be the sum of the capacities of the bins used by algorithm $\mathcal{A}$ on σ. Let $\text{cost}(\sigma)$ be the minimum possible cost to pack pieces in σ. The asymptotic performance ratio for an algorithm $\mathcal{A}$ is defined to be

$$R_{\mathcal{A}}^{\infty} = \limsup_{n\to\infty} \sup_{\sigma} \left\{ \frac{\text{cost}_{\mathcal{A}}(\sigma)}{\text{cost}(\sigma)} \middle| \text{cost}(\sigma) = n \right\}.$$

The *optimal asymptotic performance ratio* is defined to be

$$R_{\text{OPT}}^{\infty} = \inf_{\mathcal{A}} R_{\mathcal{A}}^{\infty}.$$

Our goal is to find an algorithm with an asymptotic performance ratio close to R_{OPT}^{∞}.

We now briefly review what is known about classical and variable-sized online bin packing.

The classical online bin packing problem was first investigated by Johnson [8, 9]. He showed that the NEXT FIT algorithm has a performance ratio of 2. Subsequently, it was shown by Johnson et al. that the FIRST FIT algorithm has a performance ratio of $\frac{17}{10}$ [10]. Yao showed that REVISED FIRST FIT has a performance ratio of $\frac{5}{3}$ and further showed that no online algorithm has a performance ratio less than $\frac{3}{2}$ [19]. Brown and Liang independently improved this lower bound to 1.53635 [1, 13]. Define

$$u_{i+1} = u_i(u_i - 1) + 1, \qquad u_1 = 2,$$

and

$$h_{\infty} = \sum_{i=1}^{\infty} \frac{1}{u_i - 1} \approx 1.69103.$$

Lee and Lee showed that the HARMONIC algorithm, which uses bounded space, achieves a performance ratio arbitrarily close to h_{∞} [12]. They further showed that all bounded space online algorithms achieve a performance ratio of at least h_{∞} [12]. In addition, they developed the REFINED HARMONIC algorithm, which they showed to have a performance ratio of $\frac{373}{228} < 1.63597$. The next improvement was MODIFIED HARMONIC, which Ramanan et al. showed to have a performance ratio of $\frac{538}{333} < 1.61562$ [14]. Richey [15] presents an algorithm called HARMONIC+1 and claims that it has a performance ratio of 1.58872. Seiden [16] shows that Richey's claim is incorrect and presents an algorithm HARMONIC++ with a performance ratio of 1.58889. The lower bound for online bin packing was improved to 1.5401 by van Vliet [17].

The variable-sized bin packing problem was first investigated by Friesen and Langston [6, 7]. Kinnerly and Langston gave an online algorithm with a performance ratio of $\frac{7}{4}$ [11]. Csirik proposed the VARIABLE HARMONIC (VH) algorithm and showed that it has a performance ratio of at most h_{∞} [4]. This algorithm is based on the HARMONIC algorithm of Lee and Lee [12]. Like HARMONIC, it uses bounded space. Csirik also showed that if the algorithm has two bin sizes, 1 and $\alpha < 1$, and if it is

allowed to pick α, then a performance ratio of $\frac{7}{5}$ is possible [4]. Subsequent authors have investigated this problem [2, 20], but Csirik's result yields the best performance ratio to date. No lower bounds are known.

In this work, we investigate the VH algorithm. We give a precise analysis of its performance ratio, and show that it is an optimal bounded space algorithm, by providing the first lower bound for variable-sized bin packing. An interesting feature of the analysis is that although it is shown that our algorithm achieves a performance ratio arbitrarily close to the optimum value, it is not known precisely what that value is. The case where bins of capacity 1 and $\alpha \in (0, 1)$ are used is studied in greater detail. It is shown that among algorithms which are allowed to choose α, the optimal performance ratio (and that of VH) lies in $[1.37530, 1.37532]$.

2. The VH Algorithm. Before we describe the algorithm we require a few definitions.

VH uses the NEXT FIT algorithm [8, 9] as a subroutine. This online algorithm maintains a single *open* bin. If the current item fits into the open bin, it is placed there. Otherwise, the open bin is *closed* and a new open bin is allocated. Obviously, this algorithm is online, runs in linear time, and uses constant space.

VH operates by classifying pieces according to a set of predefined intervals. The algorithm has a parameter $\epsilon \in (0, 1]$. $\mathbb{N} = \{0, 1, 2, \ldots\}$ is the set of natural numbers and $\mathbb{N}^+ = \mathbb{N} - \{0\}$. We define

$$T_i = \left\{ \frac{\alpha_i}{j} \middle| j \in \mathbb{N}^+, \alpha_i/j > \epsilon \right\}, \qquad T = \bigcup_{i=1}^{m} T_i.$$

Let the members of T be $t_1 > t_2 > \cdots > t_n$. We define $t_{n+1} = \epsilon$ and $t_{n+2} = 0$. The interval I_k is defined to be $(t_{k+1}, t_k]$ for $k = 1, \ldots, n + 1$. Note that these intervals are disjoint and that they cover $(0, 1]$.

A piece of size s has *type* k if $s \in I_k$. A piece is *big* if it has type $k \leq n$. For $k \leq n$, the *class* and *order* of interval I_k are i and j, respectively, if $t_k = \alpha_i/j$ (breaking ties arbitrarily). We extend the definitions of class and order to include pieces in the natural way.

We illustrate these definitions with the following example: We pick $m = 2$, $\alpha_1 = \frac{3}{5}$, $\alpha_2 = 1$, and $\epsilon = \frac{1}{10}$. This results in $n = 13$ with the set of intervals displayed in Table 1. Note that $t_7 = \frac{1}{5} = (3/5)/3$. Therefore, I_7 could be assigned class 1 and order 3, or class 2 and order 5. We have arbitrarily chosen the latter.

The algorithm packs pieces of different types independently, i.e. pieces of differing types never appear in the same bin. Pieces of type $n + 1$ are packed using NEXT FIT into bins of capacity 1. Pieces of class i and order j are packed j to a bin in bins of capacity α_i. The algorithm keeps one open bin for each type, into which pieces are packed until the indicated number is reached, at which point it is closed and a new open bin is allocated. Since the number of types (and thus the number of open bins) is constant, the algorithm is online, runs in linear time, and uses constant space. Note that when $m = 1$, the definition of VH corresponds exactly with that of HARMONIC.

3. An upper bound for VH. The analysis is based on *weighting functions* as in [12, 4]. A weighting function for algorithm $\mathcal{A}$ is a function $w_{\mathcal{A}} : (0, 1] \mapsto [0, 1]$ with the property

$$\text{cost}_{\mathcal{A}}(\sigma) \leq \sum_{\ell=1}^{N} w_{\mathcal{A}}(p_\ell) + O(1)$$

TABLE 1
An example with $m = 2$, $\alpha_1 = \frac{3}{5}$, $\alpha_2 = 1$, *and* $\epsilon = \frac{1}{10}$.

k	$(t_{k+1}, t_k]$	Class	Order
1	$(\frac{3}{5}, 1]$	2	1
2	$(\frac{1}{2}, \frac{3}{5}]$	1	1
3	$(\frac{1}{3}, \frac{1}{2}]$	2	2
4	$(\frac{3}{10}, \frac{1}{3}]$	2	3
5	$(\frac{1}{4}, \frac{3}{10}]$	1	2
6	$(\frac{1}{5}, \frac{1}{4}]$	2	4
7	$(\frac{1}{6}, \frac{1}{5}]$	2	5
8	$(\frac{3}{20}, \frac{1}{6}]$	2	6
9	$(\frac{1}{7}, \frac{3}{20}]$	1	4
10	$(\frac{1}{8}, \frac{1}{7}]$	2	7
11	$(\frac{3}{25}, \frac{1}{8}]$	2	8
12	$(\frac{1}{9}, \frac{3}{25}]$	1	5
13	$(\frac{1}{10}, \frac{1}{9}]$	2	9
14	$(0, \frac{1}{10}]$	—	—

for all input sequences $\sigma = p_1, \ldots, p_N$. Intuitively, the weight of a piece indicates the maximum fraction of a bin that it can occupy. We use the following function:

$$w_{\mathrm{VH}}(x) = \begin{cases} t_k & \text{if } x \in I_k \text{ with } k \le n, \\ \dfrac{1}{1-\epsilon} x & \text{if } x \in I_{n+1}. \end{cases}$$

LEMMA 3.1. *For all* σ,

$$\mathrm{cost}_{\mathrm{VH}}(\sigma) \le \sum_{\ell=1}^{N} w_{\mathrm{VH}}(p_\ell) + n + 1.$$

Proof. We first consider the cost of bins used to pack pieces of an arbitrary individual type. NEXT FIT is used to pack type $n+1$ pieces into bins of capacity 1. Each of these pieces is of size ϵ or less. Therefore, each of these bins is filled to within ϵ of its capacity; i.e., at least $1-\epsilon$ of the bin's capacity is used. Let X be the total size of type $n+1$ pieces. The number of bins used is at most

$$\left\lceil \frac{1}{1-\epsilon} X \right\rceil \le \frac{1}{1-\epsilon} X + 1 = \frac{1}{1-\epsilon} \sum_{p_\ell \in I_{n+1}} p_\ell + 1 = \sum_{p_\ell \in I_{n+1}} w_{\mathrm{VH}}(p_\ell) + 1.$$

Now consider pieces of a fixed type $k \le n$. Let f be the number of such pieces. Let i and j be the class and order of I_k, respectively. These pieces are packed j to a bin in bins of capacity α_i, and therefore the number of bins used is $\lceil f/j \rceil$. The cost to the algorithm is

$$\alpha_i \left\lceil \frac{f}{j} \right\rceil \le \frac{\alpha_i f}{j} + 1 = t_k f + 1 = \sum_{p_\ell \in I_k} w_{\mathrm{VH}}(p_\ell) + 1.$$

Summing over all types gives the desired result. □

To facilitate the analysis, we require a few definitions related to the optimal packing. Define n_i to be the number of bins of capacity α_i in the optimal packing and define $B_{i,b}$ to be the set of pieces in the bth bin of size α_i.

We claim that when upper bounding the performance ratio of an online algorithm, we may assume that each bin in the optimal packing is full. To demonstrate the claim, suppose bin b of size α_i in the optimal packing is not full. Let $x = \sum_{p \in B_{i,b}} p$. Then add a piece of size $\alpha_i - x$ to the end of the input sequence. The cost of the optimal solution does not increase, whereas the cost to the online algorithm cannot decrease, which implies the claim.

Suppose that for all σ, i, and b we have

$$\sum_{p_\ell \in B_{i,b}} w_{\mathrm{VH}}(p_\ell) \le c\,\alpha_i. \tag{1}$$

Then, for all σ, the cost incurred by VH is

$$\begin{aligned}
\mathrm{cost}_{\mathrm{VH}}(\sigma) &\le \sum_{\ell=1}^{N} w_{\mathrm{VH}}(p_\ell) + n + 1 \\
&= \sum_{i=1}^{m} \sum_{b=1}^{n_i} \sum_{p_\ell \in B_{i,b}} w_{\mathrm{VH}}(p_\ell) + n + 1 \\
&\le \sum_{i=1}^{m} \sum_{b=1}^{n_i} c\,\alpha_i + n + 1 \\
&= c \sum_{i=1}^{m} n_i \alpha_i + n + 1 \\
&= c\,\mathrm{cost}(\sigma) + n + 1,
\end{aligned}$$

where the first inequality follows from Lemma 3.1, the second inequality follows from (1), and the last step follows from the definition of the optimal offline cost. This would show the desired result, since $n + 1$ is constant with respect to $\mathrm{cost}(\sigma)$.

To prove (1), we are led to consider the following optimization problem: Maximize

$$\sum_{p \in X} w_{\mathrm{VH}}(p)/\alpha_i$$

subject to $\sum_{p \in X} p = \alpha_i$ and $i \in \{1, \ldots, m\}$. We further rewrite this to get the following definition.

DEFINITION 3.2. *$\mathcal{P}_1(\epsilon)$ is the following mathematical program: Maximize*

$$\frac{1}{\alpha_i}\left(\sum_{k=1}^{n} q_k\, t_k + \frac{\alpha_i - y}{1 - \epsilon}\right) \tag{2}$$

subject to

$$y = \sum_{k=1}^{n} q_k\, t_{k+1}, \tag{3}$$

$$y < \alpha_i, \tag{4}$$

$$q_k \in \mathbb{N}, \qquad 1 \le k \le n, \tag{5}$$

$$i \in \{1, \ldots, m\}. \tag{6}$$

Intuitively, $\mathcal{P}_1(\epsilon)$ upper bounds the amount of weight that fits into a single bin relative to the bin's size. The capacity of the bin is α_i. The number of type k pieces

in the bin is q_k. Since a type k piece is strictly larger than t_{k+1}, the total size of big pieces is strictly lower bounded by y. Therefore, strict inequality is required in (4) to ensure that all pieces fit in the bin. The total size of small pieces is $\alpha_i - y$. The objective function (2) calculates the weight in the bin divided by the bin size.

We have shown the following theorem.

THEOREM 3.3. *The performance ratio of* VH *is upper bounded by the value of* $\mathcal{P}_1(\epsilon)$.

In Appendix A, we develop a branch and bound algorithm which allows us to evaluate $\mathcal{P}_1(\epsilon)$ to any fixed accuracy. We also show that several simple greedy methods for solving $\mathcal{P}_1(\epsilon)$ do not always yield the optimal solution.

4. A lower bound for bounded space algorithms. We now consider lower bounds for bounded space algorithms. For this purpose, we need the following definition.

DEFINITION 4.1. $\mathcal{P}_2(\epsilon)$ *is the following mathematical program: Maximize*

$$\frac{1}{\alpha_i}\left(\sum_{k=1}^{n} q_k\, t_k + \alpha_i - y\right) \tag{7}$$

subject to (3)–(6).

Note that $\mathcal{P}_1$ and $\mathcal{P}_2$ differ only in their objective functions; their sets of constraints are the same. Therefore, any feasible solution to $\mathcal{P}_1$ is also feasible for $\mathcal{P}_2$. Further note that the difference between (2) and (7) vanishes as $\epsilon \to 0$. This being the case, the following result implies our main result, the optimality of VH.

LEMMA 4.2. *Any feasible solution to* $\mathcal{P}_2(\epsilon)$ *with objective value* c *yields a lower bound of* c *for all bounded space variable-sized bin packing algorithms.*

Proof. Let $q_1, \ldots, q_n, i$ be a feasible solution and let c be the corresponding value of (7). Define $Q = \sum_{k=1}^{n} q_k + 1$. We create an input with $N\,Q$ pieces. Let $\delta > 0$ be a real number. The input consists of $n+1$ groups of pieces. All pieces in a group are the same size. The kth group contains $q_k N$ pieces of size $t_{k+1} + \delta$ for $1 \le k \le n$. The last group contains N pieces of size $z = \alpha_i - y - (Q-1)\delta$. We require that δ be chosen such that $z \ge 0$ and $t_{k+1} + \delta \in I_k$ for $1 \le k \le n$.

The optimal solution uses N bins of size α_i. Each of these bins contains q_k items from group k for $1 \le k \le n$ and one item from the last group.

Consider the number of bins used by an arbitrary bounded space online algorithm $\mathcal{A}$ on this input. Since $\mathcal{A}$ is online and uses bounded space, at most a constant number (with respect to N) of pieces in group k can be packed with pieces from other groups. Therefore, the cost to pack items in the last group is at least $Nz - O(1)$. Further, the minimum cost to pack the items in group $k \le n$ is

$$\min_{1\le \ell \le m} \frac{\alpha_\ell\, q_k N}{\lfloor \alpha_\ell/(t_{k+1}+\delta)\rfloor} - O(1) = q_k N \min_{1\le \ell \le m} \frac{\alpha_\ell}{\lfloor \alpha_\ell/(t_{k+1}+\delta)\rfloor} - O(1).$$

We assert that

$$\min_{1\le \ell \le m} \frac{\alpha_\ell}{\lfloor \alpha_\ell/(t_{k+1}+\delta)\rfloor} \ge t_k. \tag{8}$$

Suppose for a contradiction that (8) is false. Then there exists an ℓ such that

$$\frac{\alpha_\ell}{\lfloor \alpha_\ell/(t_{k+1}+\delta)\rfloor} < t_k.$$

Let f be the integer $\lfloor \alpha_\ell/(t_{k+1}+\delta)\rfloor$. Note that since

$$\frac{(t_{k+1}+\delta)f}{\alpha_\ell} = \frac{t_{k+1}+\delta}{\alpha_\ell}\left\lfloor \frac{\alpha_\ell}{t_{k+1}+\delta}\right\rfloor \le 1,$$

we have $\alpha_\ell/f \ge t_{k+1}+\delta$. Therefore, we have $\alpha_\ell/f \in [t_{k+1}+\delta, t_k) \subset I_k$. However, by the definition of I_k, we have no number of the form α_ℓ/f, f an integer, within I_k. Therefore, we have reached a contradiction.

Given (8), the asymptotic performance ratio of $\mathcal{A}$ is lower bounded by

$$\lim_{N\to\infty} \frac{N\left(\sum_{k=1}^{n} t_k\, q_k + \alpha_i - y - (Q-1)\delta\right) - O(n)}{\alpha_i N} = c - \frac{(Q-1)\delta}{\alpha_i},$$

which can be made arbitrarily close to c by choosing sufficiently small δ. □

As mentioned previously, along with Theorem 3.3, this directly implies our main result.

THEOREM 4.3. VH *is an optimal bounded space online algorithm.*

5. The two-capacity problem. We now focus our attention on the case where $m = 2$, which we call the *two-capacity* problem. This problem is also studied by Csirik [4], who shows that a performance ratio of $\frac{7}{5}$ is possible if bins of size 1 and $\frac{7}{10}$ are used.

In order to simplify notation, we denote the size of the smaller bin as α. The size of the larger bin is 1, as before. We investigate how the optimal asymptotic performance ratio $R^{\infty}_{\mathrm{OPT}}$ varies as a function of α. We therefore consider the values of $\mathcal{P}_1$ and $\mathcal{P}_2$ to be functions of α, as well as ϵ. As we have shown in the preceding sections, $\mathcal{P}_2(\epsilon,\alpha) \le R^{\infty}_{\mathrm{OPT}}(\alpha) \le \mathcal{P}_1(\epsilon,\alpha)$.

Using the algorithm developed in Appendix A, we can compute a lower bound for any fixed value of α. However, what we would like to do is prove a lower bound which holds for all $\alpha \in (0,1]$. In Appendix C, we show how a lower bound for the entire interval can be derived from lower bounds for a specific set of finite points. This allows us to prove the following result.

THEOREM 5.1.

$$1.37532 > \frac{395101163}{287280000} \ge \inf_{\alpha\in(0,1]} R^{\infty}_{\mathrm{OPT}}(\alpha) \ge \frac{78392621}{57000000} > 1.37530.$$

6. Conclusions. We have shown the optimality of VH among bounded space algorithms for variable-sized bin packing. A number of open questions remain:

1. Can the amount of space used be reduced as in the work of Woeginger [18] for classical bounded space bin packing?
2. What general lower bounds can be proved for variable-sized bin packing?
3. Can MODIFIED HARMONIC be adapted to the variable-sized problem [14]?

Appendix A. Evaluating $\mathcal{P}_1$ and $\mathcal{P}_2$. We present a branch and bound algorithm to evaluate $\mathcal{P}_1(\epsilon)$ and $\mathcal{P}_2(\epsilon)$. We need to have an upper bound on the objective function value. To begin, we derive this upper bound.

Define

$$e_k = \frac{t_k}{t_{k+1}} \qquad \text{for } 1 \le k \le n,$$
$$e_{n+1} = \frac{1}{1-\epsilon},$$
$$E_\ell = \max_{\ell<k\le n+1} e_k \qquad \text{for } 0 \le k \le n.$$

e_k is called the *expansion* of type k. Intuitively, the expansion e_k is the maximum weight to size ratio for an item of type k.

LEMMA A.1. *Let $q_1, \ldots, q_n$, i be a feasible solution to $\mathcal{P}_1$. For all $0 \le \ell \le n$, the objective value* (2) *is at most*

$$\frac{1}{\alpha_i}\left(\sum_{k=1}^{\ell} q_k\, t_k + \left(\alpha_i - \sum_{k=1}^{\ell} q_k\, t_{k+1}\right) E_\ell\right). \tag{9}$$

Proof. To prove this, we note that

$$\begin{aligned}\sum_{k=\ell+1}^{n} q_k\, t_k + \frac{\alpha_i - y}{1-\epsilon} &= \sum_{k=\ell+1}^{n} q_k\, e_k t_{k+1} + (\alpha_i - y) e_{n+1} \\ &\le E_\ell \left(\sum_{k=\ell+1}^{n} q_k\, t_{k+1} + \alpha_i - y\right) \\ &= \left(\alpha_i - \sum_{k=1}^{\ell} q_k\, t_{k+1}\right) E_\ell. \qquad \square\end{aligned}$$

The algorithm to compute $\mathcal{P}_1(\epsilon)$ is displayed in Figures A.1 and A.2. By simply redefining the value of e_{n+1} to be 1, we compute $\mathcal{P}_2(\epsilon)$ instead.

The main routine of the algorithm initializes variables and iterates over the possible values of i. The variable x stores the maximum objective value found at any point in the computation.

The real work of the algorithm is done in the subroutine TRYALL. This subroutine traverses an implicit data structure which we call the *feasible solution tree*. This is a rooted tree with $n+1$ levels. The edges of the tree are labeled with natural numbers. All leaves are at level $n+1$. Along any path from the root to a leaf, the label of the ℓth edge represents the value of q_ℓ. Each such path specifies a feasible solution. TRYALL recursively traverses the feasible solution tree, avoiding subtrees which cannot improve on the best solution found so far.

During the execution of TRYALL, ℓ represents our current level in the tree. The value of y assigned in the first step of TRYALL corresponds exactly to the value y in $\mathcal{P}_1$, since $q_\ell, \ldots, q_n$ are all zero. If $\ell = n+1$, then we have reached a leaf. In this case, the value z assigned in the first step of the algorithm is exactly the objective value. When $\ell \le n$, Lemma A.1 implies the value of z is an upper bound on any objective value in the current subtree. If z does not exceed x, then we do not explore this subtree. Otherwise, we recurse for each of the possible values of q_ℓ from the largest down to zero. This heuristic drastically decreases the running time of the algorithm, as the first solution found is the greedy solution. The greedy solution is often close to the optimal one, as we see in the next section.

```
x ← 1.
For i ∈ {1, ..., m} do:
    Initialize q_k ← 0 for 1 ≤ k ≤ n.
    TRYALL(1).
Return x.
```

FIG. A.1. *The algorithm for computing $\mathcal{P}_1(\epsilon)$.*

TRYALL(ℓ):
$y \leftarrow \sum_{k=1}^{\ell-1} q_k\, t_{k+1}$.
$z \leftarrow \frac{1}{\alpha_i}\left(\sum_{k=1}^{\ell-1} q_k\, t_k + (\alpha_i - y)\, E_{\ell-1}\right)$.
If $z > x$ then:
    if $\ell = n+1$ then:
        $x \leftarrow z$.
    Else:
        $q_\ell \leftarrow \lceil (\alpha_i - y)\,/t_{\ell+1} \rceil$.
        While $q_\ell > 0$ do:
            $q_\ell \leftarrow q_\ell - 1$.
            TRYALL($\ell + 1$).

FIG. A.2. *The subroutine* TRYALL.

Appendix B. Remarks on the greedy lower bound. We should note that, in order to get a lower bound, we do not have to find the optimum value of $\mathcal{P}_2(\epsilon)$. Lemma 4.2 implies that any feasible solution works.

$x \leftarrow 1$.
For $i \in \{1, \ldots, m\}$ do:
    $k \leftarrow 1$.
    $y \leftarrow \alpha_i$.
    While $k \le n$ do:
        $q_{\pi(k)} \leftarrow \lceil (\alpha_i - y)/t_{\pi(k)+1} \rceil - 1$.
        $y \leftarrow y - q_{\pi(k)}\, t_{\pi(k)+1}$.
        $k \leftarrow k + 1$.
    $x \leftarrow \max\{x, (\sum_{k=1}^{n} q_k\, t_k + \alpha_i - y)/\alpha_i\}$.
Return x.

FIG. B.1. *The greedy algorithm.*

We consider the greedy solutions to $\mathcal{P}_2(\epsilon)$. Let π be a permutation on $\{1, \ldots, n\}$. We shall consider several choices for π. Given π, the solution is constructed by the procedure in Figure B.1. Intuitively, the greedy solution takes as many pieces of type $\pi(1)$ as possible, then as many pieces of type $\pi(2)$ as possible, etc.

There are three obvious choices for π:

1. Let π be the identity permutation. This greedily picks items by size alone.

2. Let π be the permutation such that $e_{\pi(1)} \ge e_{\pi(2)} \ge \cdots \ge e_{\pi(n)}$, and if $e_k = e_\ell$, then $\pi(k) < \pi(\ell) \Leftrightarrow k < \ell$. This greedily picks items by expansion. If two items have the same expansion, the larger item is given preference.

3. Let π be as in the previous case, except that if $e_k = e_\ell$, then $\pi(k) < \pi(\ell) \Leftrightarrow k > \ell$. If two items have the same expansion, the smaller item is given preference.

Note that for classical bounded space bin packing, all three definitions coincide. Lee and Lee [12] have demonstrated that for the classical problem, this greedy solution is optimal. However, it is not the case that the greedy solution yields the best lower bound for variable-sized bin packing: Consider $m = 2$, $\epsilon = \frac{1}{10}$, $\alpha_1 = \frac{11}{25}$, and $\alpha_2 = 1$. The value of $\mathcal{P}_2(\epsilon)$ is $\frac{122}{75} > 1.62666$. The first greedy solution yields a lower bound of $\frac{39}{25} = 1.56$. The second and third greedy solutions both yield a lower bound of $\frac{97}{60} < 1.61667$.

Appendix C. The two-capacity problem. We return to the two-capacity problem, as defined in section 5. We would like to prove a lower bound on $R^{\infty}_{\mathrm{OPT}}(\alpha)$ which holds for all $\alpha \in (0,1]$. To further this goal, we study the structure of $\mathcal{P}_2(\epsilon,\alpha)$ in more detail.

Since we need only to lower bound $\mathcal{P}_2(\epsilon,\alpha)$, for the discussion that follows we fix $i=2$. We consider only $\epsilon = 1/M$ for $M \in \mathbb{N}^+$.

Note that as α varies, the values in T_1 change, while the values in T_2 are fixed. For certain values of α, it may happen that a point in T_1 is coincident with one in T_2. We call such a point an *interesting* point. At an interesting point, a combinatorial change occurs in the structure of $t_1,\ldots,t_n$, since two points exchange order. We also include zero as an interesting point. It is not hard to see that the set of interesting points is

$$\{0\} \cup \bigcup_{g=1}^{M} \bigcup_{j=\ell}^{M} \frac{g}{j}.$$

Rename the points in this set to be $0 = s_1 < s_2 < \cdots < s_L = 1$. Define $S_k = (s_{k-1}, s_k]$. We shall explain how to lower bound the value of $\mathcal{P}_2(\epsilon,\alpha)$ for $\alpha \in S_k$.

We further split each interval S_k into disjoint subintervals $(u_1,u_2],(u_2,u_3],\ldots,(u_{\ell-1},u_\ell]$ with $u_1 = s_{k-1}$ and $u_\ell = s_k$. For some $2 \le f \le \ell$, suppose that $q_1,\ldots,q_n,i=2$ is a feasible solution to $\mathcal{P}_2(\epsilon,\alpha)$ at $\alpha = u_f$. We assert that this is a feasible solution for all $\alpha \in (u_{f-1},u_f]$. To see this, note that within $(s_{k-1},u_f]$, y is a nondecreasing linear function of α for any fixed assignment to $q_1,\ldots,q_n,i$. Therefore, (4) remains valid throughout $(s_{k-1},u_f] \supset (u_{f-1},u_f]$. Furthermore, the objective (7) is also a linear function of α for any fixed assignment to $q_1,\ldots,q_n,i$.

To get a lower bound for $(u_{f-1},u_f]$, we evaluate $\mathcal{P}_2(\epsilon,u_f)$ (fixing $i=2$). We record the assignment $q_1,\ldots,q_n$ which gives the highest lower bound. Substituting these values into (7), we get a linear function. This is minimized at either $\alpha = u_{f-1}$ or $\alpha = u_f$. We evaluate the function at these two points to get the desired lower bound on $(u_{f-1},u_f]$. Iterating over all intervals and subintervals, we can compute a lower bound for all α.

As an example, let $\epsilon = \frac{1}{10}$. This yields 33 interesting points and 32 intervals. For simplicity's sake, we do not divide an interval into subintervals. We find that $\mathcal{P}_2(\frac{1}{10},\alpha)$ is lower bounded by the set of linear functions given in Figure C.1. We get a lower bound of $\frac{273}{200} = 1.365$ at $\alpha = \frac{7}{10}$. We have combined adjacent intervals where possible. For example, we find that the lower bound function in both $(\frac{3}{4},\frac{7}{9}]$ and $(\frac{7}{9},\frac{4}{5}]$ is $\frac{53}{60} + \frac{2}{3}\alpha$, and so we have one entry for $(\frac{3}{4},\frac{4}{5}]$.

Proof of Theorem 5.1. $\mathcal{P}_1(1/100, 57143/80000)$ is $395101163/287280000$. From the table in Figure C.1, we find that if $\mathcal{P}_2(\epsilon,\alpha) < 78392621/57000000$, then we must have $\alpha \in (7/10, 3/4]$. We determine the intervals S_k for $\epsilon = 1/100$ and eliminate those which do not overlap $(7/10, 3/4]$. This leaves 154 intervals to be checked. These intervals were further divided into subintervals by including all points $7/10 + j/100000$ for $1 \le j < 2500$. We have verified using `Mathematica` that the minimum lower bound over this set of subintervals is $78392621/57000000$. □

To get a better picture of $R^{\infty}_{\mathrm{OPT}}(\alpha)$, we have computed values of $\mathcal{P}_1(\frac{1}{100},\alpha)$ and $\mathcal{P}_2(\frac{1}{100},\alpha)$ for $\alpha = j/10000$, $1 \le j \le 10000$, using `Mathematica`. The maximum difference between the two values at any of these points was less than 0.00014. Since the difference is so small, we display only $\mathcal{P}_1$ in Figure C.2.

Low α	High α	Function
0	$\frac{1}{7}$	$\frac{71}{42}$
$\frac{1}{7}$	$\frac{1}{6}$	$\frac{32}{21} + \alpha$
$\frac{1}{6}$	$\frac{2}{7}$	$\frac{71}{42}$
$\frac{2}{7}$	$\frac{3}{10}$	$\frac{283}{168}$
$\frac{3}{10}$	$\frac{1}{3}$	$\frac{32}{21} + \frac{\alpha}{2}$
$\frac{1}{3}$	$\frac{3}{8}$	$\frac{17}{9} - \frac{2\,\alpha}{3}$
$\frac{3}{8}$	$\frac{2}{5}$	$\frac{49}{30}$
$\frac{2}{5}$	$\frac{3}{7}$	$\frac{25}{21} + \alpha$
$\frac{3}{7}$	$\frac{4}{9}$	$\frac{4}{3} + \frac{2\,\alpha}{3}$
$\frac{4}{9}$	$\frac{1}{2}$	$\frac{43}{42} + \frac{4\,\alpha}{3}$
$\frac{1}{2}$	$\frac{5}{9}$	$\frac{31}{15} - \frac{4\,\alpha}{5}$
$\frac{5}{9}$	$\frac{5}{8}$	$\frac{13}{6} - \alpha$
$\frac{5}{8}$	$\frac{2}{3}$	$\frac{7}{4} - \frac{\alpha}{2}$
$\frac{2}{3}$	$\frac{7}{10}$	$\frac{25}{12} - \alpha$
$\frac{7}{10}$	$\frac{5}{7}$	$\frac{7}{8} + \frac{7\,\alpha}{10}$
$\frac{5}{7}$	$\frac{3}{4}$	$\frac{8}{9} + \frac{2\,\alpha}{3}$
$\frac{3}{4}$	$\frac{4}{5}$	$\frac{53}{60} + \frac{2\,\alpha}{3}$
$\frac{4}{5}$	$\frac{5}{6}$	$\frac{1}{42} + \frac{17\,\alpha}{10}$
$\frac{5}{6}$	$\frac{6}{7}$	$\frac{4}{21} + \frac{3\,\alpha}{2}$
$\frac{6}{7}$	$\frac{9}{10}$	$\frac{1}{3} + \frac{4\,\alpha}{3}$
$\frac{9}{10}$	1	$\frac{1}{42} + \frac{5\,\alpha}{3}$

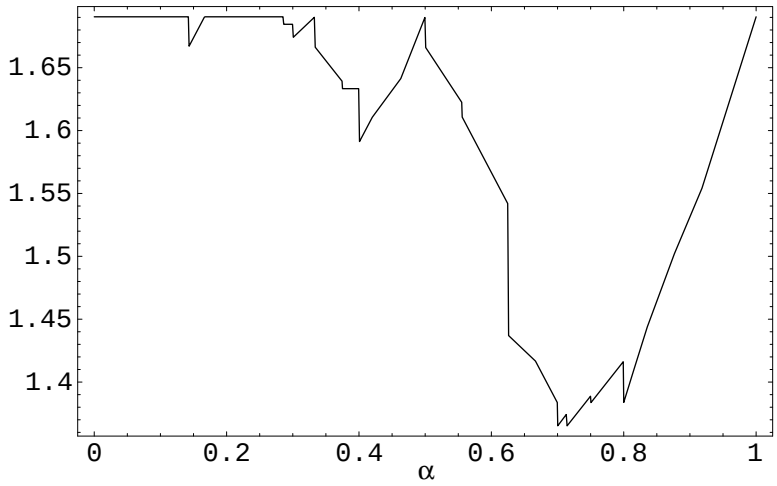


FIG. C.1. *The lower bound function for* $\epsilon = \frac{1}{10}$.

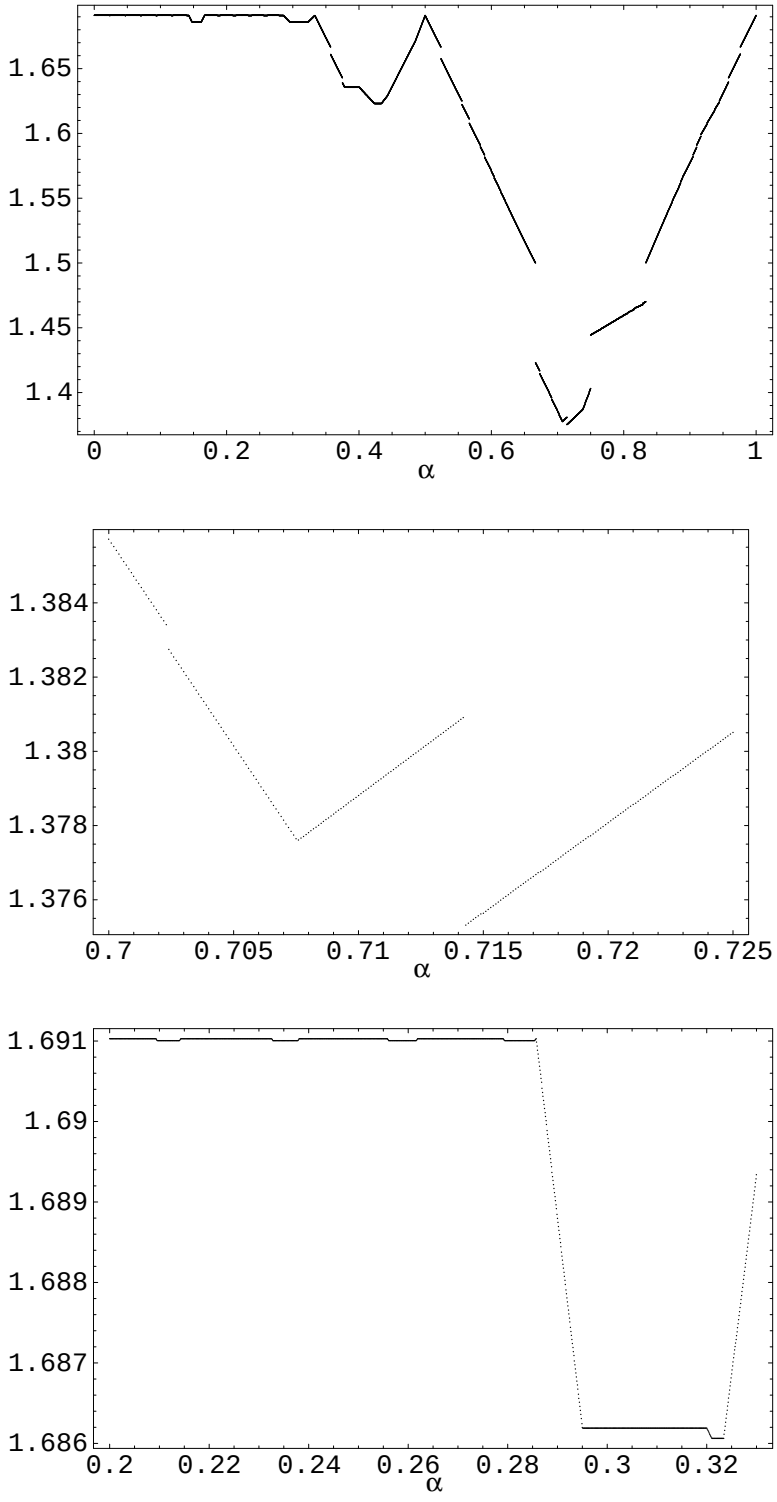


FIG. C.2. *Values of* $\mathcal{P}_1(\frac{1}{100}, \alpha)$ *with close-ups of the regions* [.7,.725] *and* [.2,.33].

REFERENCES

[1] D. J. Brown, *A Lower Bound for On-Line One-Dimensional Bin Packing Algorithms*, Tech. Rep. R-864, Coordinated Sci. Lab., University of Illinois at Urbana-Champaign, Urbana-Champaign, IL, 1979.

[2] R. Burkard and G. Zhang, *Bounded space on-line variable-sized bin packing*, Acta Cybernet., 13 (1997), pp. 63–76.

[3] E. G. Coffman, M. R. Garey, and D. S. Johnson, *Approximation algorithms for bin packing: A survey*, in Approximation Algorithms for NP-hard Problems, D. Hochbuam, ed., PWS, Boston, MA, 1997, pp. 46–93.

[4] J. Csirik, *An on-line algorithm for variable-sized bin packing*, Acta Inform., 26 (1989), pp. 697–709.

[5] J. Csirik and G. Woeginger, *On-line packing and covering problems*, in On-Line Algorithms—The State of the Art, A. Fiat and G. Woeginger, eds., Lecture Notes in Comput. Sci. 1136, Springer-Verlag, Berlin, 1996, pp. 147–177.

[6] D. K. Friesen and M. A. Langston, *A storage-size selection problem*, Inform. Process. Lett., 18 (1984), pp. 295–296.

[7] D. K. Friesen and M. A. Langston, *Variable sized bin packing*, SIAM J. Comput., 15 (1986), pp. 222–230.

[8] D. S. Johnson, *Near-Optimal Bin Packing Algorithms*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1973.

[9] D. S. Johnson, *Fast algorithms for bin packing*, J. Comput. Systems Sci., 8 (1974), pp. 272–314.

[10] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham, *Worst-case performance bounds for simple one-dimensional packing algorithms*, SIAM J. Comput., 3 (1974), pp. 299–325.

[11] N. Kinnersley and M. Langston, *Online variable-sized bin packing*, Discrete Appl. Math., 22 (1989), pp. 143–148.

[12] C. Lee and D. Lee, *A simple on-line bin-packing algorithm*, J. ACM, 32 (1985), pp. 562–572.

[13] F. M. Liang, *A lower bound for online bin packing*, Inform. Process. Lett., 10 (1980), pp. 76–79.

[14] P. Ramanan, D. Brown, C. Lee, and D. Lee, *On-line bin packing in linear time*, J. Algorithms, 10 (1989), pp. 305–326.

[15] M. B. Richey, *Improved bounds for harmonic-based bin packing algorithms*, Discrete Appl. Math., 34 (1991), pp. 203–227.

[16] S. Seiden, *On the online bin packing problem*, in Proceedings of the 28th International Colloquium on Automata, Languages, and Programming, Crete, Greece, 2001, pp. 237–249.

[17] A. van Vliet, *An improved lower bound for online bin packing algorithms*, Inform. Process. Lett., 43 (1992), pp. 277–284.

[18] G. Woeginger, *Improved space for bounded-space, on-line bin-packing*, SIAM J. Discrete Math., 6 (1993), pp. 575–581.

[19] A. C. C. Yao, *New algorithms for bin packing*, J. ACM, 27 (1980), pp. 207–227.

[20] G. Zhang, *Worst-case analysis of the FFH algorithm for online variable-sized bin packing*, Computing, 56 (1996), pp. 165–172.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 471–480

CLASSIFICATION OF HOMOMORPHISMS TO ORIENTED CYCLES AND OF k-PARTITE SATISFIABILITY*

TOMÁS FEDER†

Abstract. We show that, for every choice of an oriented cycle H, the problem of whether an input digraph G has a homomorphism to H is either polynomially solvable or NP-complete. Along the way, we obtain simpler proofs for two known polynomial cases, namely, oriented paths and unbalanced oriented cycles, and exhibit two new simple polynomial cases of balanced oriented cycles. The more difficult cases of the classification are handled by means of a new problem, the bipartite boolean satisfiability problem. In general, the k-partite boolean satisfiability problems are shown to be either polynomially solvable or NP-complete, thus generalizing Schaefer's classification of boolean satisfiability problems.

Key words. digraph homomorphism, oriented cycles, constraint satisfaction

AMS subject classifications. 05C99, 05C20, 05C38

PII. S0895480199383353

1. Introduction.

A large class of problems in artificial intelligence and other areas of computer science can be viewed as *constraint-satisfaction problems*. This includes problems in machine vision, belief maintenance, scheduling, temporal reasoning, graph theory, and satisfiability. An instance of constraint-satisfaction is given by a pair I, T of finite relational structures over the same vocabulary. (The vocabulary is the list of relation names and their arities.) The instance is satisfied if there is a homomorphism from I to T; that is, there exists a mapping h such that for every tuple $(x_1, \ldots, x_k) \in R_i$ in I, we have $(h(x_1), \ldots, h(x_k)) \in R_i$ in T. Intuitively, the elements of I should be thought of as variables, and the elements of T should be thought of as possible values for the variables. The tuples in the relations of I and T should be viewed as constraints on the set of allowed assignments of values to variables. The set of allowed assignments is nonempty iff there exists a homomorphism from I to T.

It is well known that the constraint-satisfaction problem is NP-complete. In practice, however, one often encounters the situation where the structure T (which we call the *template*) is fixed, and it is only the structure I (which we call the *instance*) that varies.

Feder and Vardi [1] considered constraint-satisfaction with respect to fixed templates and asked whether each template defines either a polynomially solvable problem or an NP-complete problem. In other words, they asked whether there is a *dichotomy*; i.e., there is no constraint-satisfaction problem with respect to a fixed template that is not polynomially solvable or NP-complete. Such dichotomies have been shown in several special cases of constraint-satisfaction; see [1] for an overview. They showed that the class of constraint-satisfaction problems with respect to fixed templates contains as special cases the problems of k-satisfiability, k-colorability, systems of linear equations modulo q, and labeled graph isomorphism; they also showed that the general case is equivalent to the case of digraphs, known as digraph homomorphism.

That is, let H be a fixed digraph. For an input digraph G, the problem is to decide whether there is a homomorphism from G to H, i.e., a mapping from the vertices of G

*Received by the editors February 1, 1999; accepted for publication (in revised form) August 6, 2001; published electronically October 4, 2001.
http://www.siam.org/journals/sidma/14-4/38335.html
†268 Waverley St., Palo Alto, CA 94301 (tomas@theory.stanford.edu).

to vertices of H such that adjacent vertices in G map to adjacent vertices in H in the same direction. As said above, classifying the digraphs H as defining polynomially solvable or NP-complete problems would yield such a classification on templates for constraint-satisfaction.

The digraph homomorphism problem has been extensively studied; see, e.g., [2, 3, 7, 8, 9]. One of the cases that they focused on is that of oriented cycles, i.e., the case where H is a cycle with each of its edges oriented in one of the two possible directions. Partial results indicating that some cycles define polynomial [9] and some define NP-complete [2] problems were obtained. In particular, the cases of oriented paths and of unbalanced oriented cycles were classified as polynomially solvable. In this paper, we complete the classification by showing that each oriented cycle H gives rise to either a polynomially solvable or an NP-complete digraph homomorphism problem. The approach is based on ideas of Feder and Vardi [1], which suggest several classes of polynomially solvable constraint-satisfaction problems, and uses explicitly the classification for the boolean case of constraint-satisfaction by Schaefer [10].

What is needed is in fact a generalization of the boolean constraint-satisfaction classification by Schaefer, namely, one where the template does not contain just two elements, yet every element of the input structure is restricted in the instance to map to one of two specific elements. That is, every element ranges over a set $S_i = \{0_i, 1_i\}$ for k different sets S_i (that is, what we call the k-partite boolean satisfiability problem), and we classify such problems as polynomially solvable or NP-complete.

2. Some polynomial cases. Feder and Vardi [1] observed that all known polynomially solvable cases of constraint satisfaction can be solved by means of Datalog and group theory, defining the bounded width and subgroup classes of problems, respectively. Within the bounded width class, they identified several subclasses for which membership is decidable: in particular, the width 1 class (also known as the tree duality class [5], with its extension the extended tree duality class [1]), and the bounded strict width class. This last class, the bounded strict width class, will be of special interest here.

The bounded width class uses a canonical algorithm, for problems of width (l, k), that involves inferring all possible constraints on l variables at a time by considering k variables at a time. We may in addition require that if this inference process does not reach a contradiction (the empty set), then it should be possible to obtain a solution by greedily assigning values to the variables one at a time while satisfying the inferred 1-constraints. We say that a constraint-satisfaction problem that can be solved in this way has *strict width* (l, k) and say that it has *strict width* l if it has strict width (l, k) for some k. It turns out that strict width l is equivalent to strict width (l, k), for all $k > 1$, so we can assume $k = l + 1$.

This intuition behind strict width l can also be captured in two other ways. First, we can require that if we have an instance and after assigning specific values to some of the variables we obtain an instance with no solution, then some l out of the specific value assignments chosen are sufficient to give an instance with no solution. We refer to this property as the l-Helly property. Second, we could require that there exists a function g that maps $l + 1$ elements from the domain of the structure T to another element with the property that if all but at most one of the $l + 1$ arguments are equal to some value b, then the value of g is also b. Furthermore, for all relations R in the template, if we have $l + 1$ tuples satisfying R, then the tuple obtained by applying g componentwise also satisfies R. We call this property the l-mapping property. The following is from Feder and Vardi [1].

THEOREM 1. *Strict width l, the l-Helly property, and the l-mapping property are equivalent.*

We shall show here that several cases of digraph homomorphism have strict width 2, by using the 2-mapping property, and are therefore polynomially solvable. That is, fix a digraph H. Then the digraph homomorphism problem for H has strict width 2 iff there is a mapping $t = g(x, y, z)$ from triples of vertices in H to vertices in H such that $g(x, x, x) = g(x, x, y) = g(x, y, x) = g(y, x, x) = x$, and if (x, x'), (y, y'), (z, z') are edges of H, then $(t, t') = (g(x, y, z), g(x', y', z'))$ is also an edge of H.

Oriented paths were shown to be polynomially solvable in [2, 3, 8]. Here we show this by establishing strict width 2. This stronger result will be of use later in connection to the 2-Helly property.

THEOREM 2. *Oriented paths have strict width* 2.

Proof. Oriented paths have vertices numbered $1, 2, \ldots, r$, with all edges from i to $i + 1$, or from $i + 1$ to i, and exactly one of both for each i. Let $g(x, y, z) =$ median (x, y, z), i.e., the middle value out of x, y, z. Note that two edges (x, x') and (y, y') with $x < y$ give $x' \leq y'$, so for the definition of the 2-mapping property, the order of vertices is preserved by the adjacency relation, and hence the same argument is selected by g before and after traversing an edge; that is, the 2-mapping property holds. □

An oriented cycle is *balanced* if it has the same number of edges on the cycle in one direction and in the other, *unbalanced* otherwise. It is known [2, 12] that unbalanced cycles are polynomially solvable. Here we show this by establishing strict width 2.

THEOREM 3. *Unbalanced cycles have strict width* 2.

Proof. Traverse the cycle in one direction, obtaining vertices $0, 1, \ldots, k$, where vertices 0 and k designate the same vertex. Define level(i) in such a way that if $(i, i + 1)$ is an edge, then level($i + 1$) = level(i) + 1; otherwise, if $(i + 1, i)$ is an edge, then level($i + 1$) = level(i) − 1. Notice that level (k) ≠ level(0), since the oriented cycle is unbalanced. Assume level(k) < level(0) (by an arbitrary choice of direction) and also that the smallest level is 0. The levels thus look in order like this: $l + m, \ldots, 0, \ldots, 0, \ldots, 0, \ldots, l$ for some $m > 0$. Adding m to the levels starting from the first 0, we obtain $l + m, \ldots, 0 \equiv m, \ldots, m, \ldots, m, \ldots, l + m$; i.e., by again choosing the starting vertex of the ordering, we have $m, \ldots, 0$ with a single 0.

Now that we have ensured that level(0) = m, level(k) = 0, and level(i) > 0 for $i \neq k$, define the lexicographic ordering on pairs (level(i), i). We refer to the sets of vertices whose levels differ by a multiple of m as *layers*. To define the 2-mapping g, if the three arguments are in three different layers, we always return the first argument; if only two of the three arguments are in the same layer, we always return the first of these two arguments; the only interesting case thus has all three arguments in the same layer. For the layer containing the endpoint $k \equiv 0$, the two lexicographically smallest pairs are precisely the two corresponding $(0, k)$ and $(m, 0)$. Also note that if (x, x') and (y, y') are edges with $x < y$ in the lexicographic ordering, and with x and y in the same layer, then x' and y' are in the same layer, and $x' \leq y'$ in the lexicographic ordering. We can therefore define $g(x, y, z) =$ median(x, y, z) in the lexicographic ordering, as for paths in the previous theorem, with x, y, z in the same layer, thus obtaining the 2-mapping property. □

We now turn to balanced cycles. For balanced cycles, the above procedure for assigning levels from the previous theorem assigns the same level to vertices 0 and k, i.e., $m = 0$. Denote the lowest and highest levels by l and h. Then, as we traverse the cycle, we encounter a sequence of l's and h's, which is in general of the form $(l^+h^+)^i$

with $i \geq 1$. Here l^+h^+ denotes one or more repetitions of l's followed by one or more repetitions of h's, and i indicates the number of times this pattern is encountered. We first consider the case $i = 1$ and show that it has strict width 2. We next consider the case $i = 2$ and show that when there are no repetitions, i.e., when the pattern is $lhlh$, it also has strict width 2. The general case of $i = 2$, i.e., $l^+h^+l^+h^+$, will be considered in section 4 and shown to be always either polynomial or NP-complete. The case of $i = 3$ will be shown to be NP-complete in section 3. This will complete the classification.

For the two cases that we show have strict width 2 by means of an appropriate 2-mapping g, it is again sufficient to define g when all three arguments are in the same level.

THEOREM 4. *Balanced cycles of the type l^+h^+ have strict width* 2.

Proof. Let l_1 and l_2 be the first and last l's in l^+h^+. Then the cycle consists of a path p_1, from l_1 to l_2, that encounters no other l's, but encounters one or more h's, and a path p_2 from l_1 to l_2 that may encounter more l's (although possibly $l_1 = l_2$) but does not encounter any h's. Number the vertices $0, 1, \ldots, k_1$ on p_1 without assigning a number to l_2 if $l_1 = l_2$ and number the vertices $0, 1, \ldots, k_2$ on p_2 without assigning a number to either l_1 or l_2. For the numbered vertices i on p_1, assign them the pair $(i, 0)$. Let j be a specific numbered vertex on p_1 of level h. To the numbered vertices i on p_2, assign the pair (j, i) and consider the lexicographic ordering on pairs. Once again, it can be seen that edges don't cross. Therefore the function $g(x, y, z) = \text{median}$ (x, y, z), where the median is taken in the lexicographic ordering, and x, y, z belong here to the same level, proves the 2-mapping property. □

THEOREM 5. *Balanced cycles of the type lhlh have strict width* 2.

Proof. Assign subindices to the two occurrences of l and of h, so that the type can be denoted by $l_1h_1l_2h_2$. Consider the four paths $1 = l_1h_1$, $2 = l_2h_1$, $3 = l_2h_2$, $4 = l_1h_2$ on the cycle; given three paths $i - 1$, i, $i + 1$ out of these four (modulo 4), the *middle path* is i. The mapping $g(x, y, z)$ with the three arguments at the same level is defined as follows: (0) For levels l and h, it is just majority; (1) If x, y, z belong to three different paths, return the one that belongs to the middle path; (2) If x, y, z belong to the same path, return the one in the middle position (median) on the path; (3) If exactly two out of x, y, z belong to the same path, return the one among the two occurring earliest on the path. This definition satisfies the 2-mapping property. □

3. Some NP-complete cases. We saw in the last section that unbalanced cycles are polynomial and that balanced cycles can be categorized by their type $(l^+h^+)^k$ with $k \geq 1$. We saw that the case $k = 1$ is polynomial; the case $k = 2$ is left for the next section, although we saw that the special case *lhlh is* polynomial. Here we show that the case $k \geq 3$ is NP-complete. The reduction is from k-colorability.

Given an oriented path, we assign levels to the vertices of the path so that if (i, j) is an edge on the path, then $\text{level}(j) = \text{level}(i) + 1$. We say that a path is of *type* r if it starts on level 0, it ends on level r, and all intermediate vertices are on levels i with $0 \leq i \leq r$. The following lemma was also proved in [11].

LEMMA 1. *Let p_1 and p_2 be two paths of type r. Then there is a path p of type r that maps homomorphically to p_1 and to p_2 in such a way that the starting vertex of p maps to the starting vertices of p_1 and p_2, and the ending vertex of p maps to the ending vertices of p_1 and p_2.*

Proof. The proof is first by induction on r and then by induction on the number of vertices of p_1 and p_2 for r fixed. Suppose that neither p_1 nor p_2 have vertices at

level 0 other than their starting vertices. Then we can make p advance on both paths to level 1 and reduce the problem to type $r-1$.

So we may assume that at least one of p_1 and p_2, say p_1, has a vertex at level 0 other than the starting vertex. Let v be the earliest occurring vertex on p_1 at level 0 other than the starting vertex. Let w_i be the earliest occurring vertex on p_i at level r, and let u_i be the vertex preceding w_i at level $r-1$. By inductive hypothesis on r, path p may be started by mapping all the way to the u_i in each p_i and then from u_i to w_i by traversing a single edge.

Now we may extend p mapping from w_1 to v and from w_2 to the starting vertex of p_2. This is by induction on the number of vertices on the two paths with r fixed. Finally, we may extend p mapping from v to the ending vertex of p_1 and from the starting vertex of p_2 to the ending vertex of p_2 again by induction on the number of vertices on the two paths with r fixed. □

We shall refer to a path p, constructed as in the preceding lemma so as to map to any number of given paths, as a *generic path*. Given an oriented cycle of type $(l^+h^+)^k$, we can construct a path that can map precisely to the paths joining any two consecutive l and h. This is done by beginning with an edge from l to $l+1$, then a generic path $l+1$ to $h-1$, then an edge from $h-1$ to h. We can also construct paths that allow us to move between l's in the same l^+. This is done with a generic path from l to $h-1$ followed by a generic path from $h-1$ to l. This allows us to think of the oriented cycle as a cycle $(0,1,2,3,\ldots,2k-1)$ with edges oriented from the even number (corresponding to a single l^+) to the adjacent odd numbers (corresponding to single h^+) modulo $2k$.

A digraph H is a *core* unless there is a proper subdigraph H' and a homomorphism from H to H'; if H is not a core, then such an H' which is a core is called the core of H; otherwise, H *is* its own core. Clearly, if H has core H', then digraph homomorphism for H is equivalent to digraph homomorphism for H'. In the case of an oriented cycle H, if it is not a core, then its core is an oriented path and digraph homomorphism is polynomially solvable, as we saw in the previous section.

Therefore we assume that H is a core. Now, if the instance G contains a copy of H, then this copy must map to H by the identity mapping up to isomorphisms of H; since we can disregard isomorphisms in the solution, we can assume that a specific copy of H in G maps to H via the identity mapping and identify this copy with H itself.

We say that a binary relation can be represented in an instance of the problem if there is an instance G of the problem such that for two specific vertices x, y in G, the possible images that these two vertices can simultaneously have in a homomorphism to H are precisely those in the binary relation.

LEMMA 2. *The binary relation relating opposite numbers i and $i+k$ modulo $2k$ with i even on the cycle can be represented in an instance.*

Proof. Join the fixed cycle $(0,1,\ldots,2k-1)$ to a cycle $(0',1',\ldots,(2k-1)')$ by edges joining i to i'. Then either all i' map to $i-1$ or all map to $i+1$. Repeating this construction, we can ensure that either all i'' map to $i'-1$ or all map to $i'+1$; hence, either they all map to $i-2$, all to i, or all to $i+2$. After repeating this construction $k-1$ times, we obtain a cycle that can be in any of the k possible positions of the same parity, and we can then select two opposite vertices on this cycle. □

LEMMA 3. *The binary relation relating any two different i, j between 0 and $2k-1$ with both i, j can even be represented in an instance.*

Proof. Find for i the opposite vertex on the cycle as in the preceding lemma and

then attach to this opposite vertex a path of length $k-2$ ending in j with its edges oriented appropriately. Then j can go to vertices which are also even as i but not to i itself. □

The relation in the lemma relates two out of k colors precisely when they are different; i.e., it is the k-colorability relation. This gives the following theorem.

THEOREM 6. *Balanced cycles of type* $(l^+h^+)^{\geq 3}$ *are NP-complete for cores (and polynomial for noncores).*

4. The remaining cases. The remaining case consists of balanced cycles of the form $l^+h^+l^+h^+$, which we denote, with the purpose of distinguishing groups of l and h, by $l_0^+h_0^+l_1^+h_1^+$. We have seen that the case $l_0h_0l_1h_1$, with no repetitions, is polynomially solvable. We might thus hope to complete the classification by a case analysis on the number of repetitions. One can observe, however, that for each remaining choice of the number of repetitions, there are both polynomially solvable and NP-complete cases. That is, the complexity of the problem depends on the internal structure of the paths themselves joining l and h vertices to each other and among themselves. It turns out that what really matters is whether there exist digraphs that can map to some of these paths but not to certain others. The possible patterns of which subsets of paths can thus be obtained give rise to a bipartite boolean satisfiability instance, and we use here a classification result for such problems, given in the next section, to complete the classification of oriented cycles.

To understand the characterization given below, consider an instance consisting of a digraph G. We can assume that every oriented cycle in G is balanced, since the oriented cycle H to which G must be mapped is balanced. This means that G has a level structure; i.e., we can assign levels to the vertices of G so that if (u,v) is an edge of G, then $\text{level}(v) = \text{level}(u)+1$. Furthermore, the number of levels of G must be no larger than the number of level of the balanced cycle H. Furthermore, if the number of levels of G is smaller, then G does not map to both l and h vertices in H, so G maps to paths in H, and we have seen that this case is polynomially solvable.

We therefore assume that G has the same number of levels as H and label these levels the same as in H. Now consider all the l vertices and all the h vertices in G. Let S be the set of those l vertices that are joined in G to h vertices by an oriented path not going through any other l vertices and those h vertices that are joined in G to l vertices by an oriented path not going through any other h vertices.

If we remove from G the set S, the digraph G becomes disconnected into connected components. Let the *boundary of* a connected component be set of l and h vertices in S incident on the component; i.e., they connect the component to the rest of G. Notice that each vertex in S must be in the boundary of at least one component containing both l's and h's by the way S was defined.

The next step is to ensure that the boundary of each component contains at most two vertices. The easiest case is that of components that have both l and h boundary vertices. Such components must map to one of the four paths in H joining consecutive l and h vertices. These four paths contain only one l and one h vertex, so all l vertices in the component must map to the same l, and similarly for the h vertices. We may thus collapse together the l vertices in the component and similarly for the h vertices. We thus have precisely two boundary vertices.

The more difficult case is that of components that have only h boundary vertices or only l boundary vertices, say, only h boundary vertices. Such components must map to one of the two maximal paths containing h vertices but no l vertex. We know from section 2 that these paths have strict width 2, and that this property is

equivalent to the 2-Helly property. That is, we can replace a component having some number $r > 2$ of boundary vertices, with several copies of the component, one for each choice of two out of the r boundary vertices, where only these two boundary vertices are attached at their proper places in G. The condition on possible images for the boundary vertices as imposed by the single component is the same as the condition imposed by its copies on pairs of boundary vertices. We thus have copies with at most two boundary vertices.

Now if a component contains an l and an h vertex, they must map to an l_x and an h_y vertex, respectively, where x, y are boolean variables, and they range over $\{0, 1\}$. We thus associate with the component the pair of variables (x, y). Note that a choice of values x, y tells us to which path the component must map. The paths out of the four possible choices where such a component can map gives us a boolean constraint on x, y.

Now consider the case of a component whose at most two boundary vertices are h vertices; the case of l vertices is similar. Say there are two boundary h vertices; the case of a single boundary h vertex is a special case. This component must map to one of two possible maximal paths without l vertices. The two boundary vertices correspond to two pairs (x, y) and (x', y') of boolean variables as described above, since they are also boundary vertices of components having both l and h vertices. We must have $y = y'$. We thus get a boolean constraint on x, x', y.

We have assumed that the x, y come in pairs. However, it is easy to state $y = y'$ for two pairs, (x, y) and (x', y'), by requiring vertices to be h vertices from the same h^+. This is done as in Lemma 1 in the preceding section by connecting two h vertices by a generic path that does not reach level l. We can similarly state $x = x'$. Thus we can assume that variables x and y come separately, in two different sets of boolean variables, and that we have a collection of boolean constraint types of one of the three forms $R(x, y)$, $R(x, x', y)$, and $R(x, y, y')$. Boolean constraint-satisfaction problems were classified as polynomial or NP-complete by Schaefer [10]. This kind of bipartite boolean constraint-satisfaction problem, with two different types of boolean variables, will be classified later as polynomial or NP-complete, as an extension of Schaefer's result. This gives the following theorem.

THEOREM 7. *Balanced cycles of type $l^+h^+l^+h^+$ are either polynomially solvable or NP-complete.*

To decide the complexity of a problem, we must determine which are the boolean constraints of a given type that can be stated by an instance, that is, which relations on two vertices can be stated by an instance of the problem. This question is polynomially decidable, for a chosen such relation, by deciding for each choice of one of r solutions for the two vertices, where all the other vertices will map, in n^r possible ways, where n is the number of vertices in H. We can thus construct an instance G with n^r vertices and all the constraints (edges) consistent with where the vertices map in the r solutions. Then check that these are the only possible solutions for the resulting instance, as far as the two special vertices are concerned. Polynomiality follows from the fact that r is here a constant, the components thus defined map to paths, as described before, and the problem for paths is polynomial.

In Schaefer's classification, there are three polynomially solvable cases, namely, Horn clauses, 2SAT, and linear equations modulo 2. In terms of the approach of Feder and Vardi [1], the first two are explained by Datalog, i.e., of bounded width, while the third is explained by group theory; i.e., it is a subgroup problem. All the previous cases of oriented cycles that were shown to be polynomial belong to the bounded

width case. Is it possible that the polynomial cases in the last theorem include cases that are not of bounded width, yet polynomial because of group theory? For this to be the case, it would have to happen that some relation $R(x, x', y)$ is either $x+x'+y = 0$ or $x + x' + y = 1$ modulo 2. In either case, for a fixed y, we would have a constraint $x + x' = 1$ modulo 2. That is, a component with two h boundary vertices maps the two boundary vertices to two different boundary vertices of the h^+ sector in the two possible ways. However, then the two boundary vertices are separated by vertices that map to the same vertex in both solutions. (Just consider how a path joining the two boundary vertices can map in the two directions and infer that some vertex on the path must map to the same vertex in both solutions.) Then the two solutions can be combined to make the two boundary vertices map to the same vertex, a contradiction. Therefore all polynomial problems are of bounded width.

THEOREM 8. *All oriented cycles define digraph homomorphism problems that are either of bounded width, and hence polynomially solvable, or NP-complete.*

5. The k-partite boolean satisfiability problem. Schaefer showed that all satisfiability problems on a boolean domain are either polynomial or NP-complete [10]. Here we consider a k-partite generalization of the problem. There are k types of variables, each of which ranges over the boolean domain $S_i = \{0_i, 1_i\}$ for $1 \leq i \leq k$. There are also a number of constraint types $R(x_1, x_2, \ldots, x_r)$ that can be imposed on r variables at a time, where x_j ranges over some corresponding S_i.

We show that such k-partite satisfiability problems are also either polynomial or NP-complete. As an application of the bipartite case of this classification, we obtain the above classification of homomorphisms to oriented cycles as polynomial or NP-complete for each choice of an oriented cycle.

We first consider the bipartite case. Suppose that variables $x, x', x'', \ldots$ range over $A = \{0_A, 1_A\}$ and variables $y, y', y'', \ldots$ range over $B = \{0_B, 1_B\}$. If the constraints contain one of the two conditions $x = y$ or $x \neq y$, then we can identify each variable in A with a variable in B, and vice versa, so we actually have a single domain $A = B = \{0, 1\}$ and we can apply Schaefer's classification result to infer that the problem is either in P or NP-complete. Therefore we assume that neither $x = y$ nor $x \neq y$ can be stated.

What then are the binary relations involving one element from A and one from B? They can only be 2SAT clauses. Say the clause $x \vee y$ is present, while the other cases are analogous. Then the clause $\neg x \vee \neg y$ is not present; otherwise, both combined would give $x \neq y$. Similarly, at most one of the two clauses $x \vee \neg y$ and $\neg x \vee y$ can be present; otherwise, both combined would give $x = y$. Say only $x \vee \neg y$ out of these two may be present. Then the at most two clauses present involve the variable x positive (no negation).

We show that at least one 2SAT clause must be present; otherwise, the problems defined by A and B are independent, so the problem is in P if both the A and B parts are in P and NP-complete if at least one of them is NP-complete. For suppose that the two parts are not independent. Then there is a relation R such that $R(x, y')$ and $R(x', y)$ hold, but $R(x, y)$ does not, where the x's and y's are vectors of variables. Replace x and y by minimal subvectors such that $R(x, y)$ still does not hold when the suppressed variables are free. This means that complementing one value in x or y makes R true. Now set all but one of the variables in x to their assigned value in x, leaving the single value x_1, and similarly set all but one of the variables in y to its assigned value in y, leaving the single value y_1. Then $R'(x_1, y_1)$ is false, but both $R'(x_1, \neg y_1)$ and $R'(\neg x_1, y_1)$ are true, giving, say, if $x_1 = y_1 = 0$, either $x \vee y$ or $x \neq y$.

Therefore we assume that $x \vee y$ is present, that $x \vee \neg y$ may be present, and that no other binary condition involving A and B is present. We claim that after suppressing all the variables whose values are forced to 0 or 1, a solution exists iff a solution exists after setting all variables x in A to 1. Therefore the satisfiability problem is equivalent to the satisfiability problem for B, hence either in P or NP-complete, by Schaefer's result.

To prove the claim, suppose to the contrary that there is a solution to the problem that is no longer a solution when all x variables from A are replaced by 1. Consider a condition $R(x_1, \ldots, x_r, y_1, \ldots, y_s)$ such that when we assign 1 to the x_i and the values v_i from the solution to the y_i, we end up having $R(1, \ldots, 1, v_1, \ldots, v_s)$ false. Replace a maximal number of the $x_i = 1$ by free variables, so that the resulting relation $R(1, \ldots, 1, x_{t+1}, \ldots, x_r, v_1, \ldots, v_s)$ is still false for all values of x_i. Simplifying the notation by suppressing the free variables x_i, we have $R(1, \ldots, 1, v_1, \ldots, v_s)$ false, yet R becomes true when one of the first t 1's is replaced by 0. If $t = 0$, then $R(v_1, \ldots, v_s)$ false indicates that these are forbidden values in B, contrary to the assumption that there is a solution with those values. If $t = 1$ and $s = 0$, then $R(1)$ is false, so the single argument of R was forced to 0 in the preprocessing stage. Suppose $t = 1$ and $s \geq 1$. We thus have $R(1, v_1, \ldots, v_s)$ false but $R(0, v_1, \ldots, v_s)$ true. Again replace a maximal number of v_i by free variables y_i, so that $R(1, v_1, \ldots, v_u, y_{u+1}, \ldots, y_s)$ is still false for all values of y_i. Simplifying the notation by suppressing the free variables y_i, we have that $R(1, v_1, \ldots, v_u)$ is false but becomes true when any of the $u + 1$ arguments is complemented. Fix the last $u - 1$ at their assigned v_i. We then have $R(1, v_1)$ false but $R(1, \neg v_1)$, $R(0, v_1)$ true, so if, say, $v_1 = 0$ we obtain either $\neg x \vee y$ or $x = y$, two cases that were not allowed. The remaining case has $t \geq 2$. Fix $t - 2$ of the 1's to 1, and fix all remaining variables to the given v_i, so that we get $R(1, 1)$ false and $R(0, 1)$, $R(1, 0)$ true. This gives either $\neg x \vee \neg x'$ or $x \neq x'$, which combined with the clause $x' \vee y$ gives $\neg x \vee y$, a type of clause that was not allowed. This completes the proof.

THEOREM 9. *All cases of bipartite satisfiability are either in P or NP-complete.*

To generalize the argument to the k-partite case, where there are k boolean sets S_i, introduce two literals x_i and $\neg x_i$ for each S_i; write each 2SAT clause involving two sets S_i and S_j as two implications, say, $x_i \vee x_j$ is written as $\neg x_i \rightarrow x_j$ and $\neg x_j \rightarrow x_i$; and view the resulting implications as a digraph. If the digraph contains a cycle, then equality (or complementarity) can be forced among variables in the cycle, thus reducing the number of parts by collapsing together the S_i in the cycle. If the digraph is acyclic, then some literal, say $\neg x_1$, is a source, in which case x_1 is a sink. Thus all clauses involving x_1 are of the form $x_1 \vee x_i$ or $x_1 \vee \neg x_i$. Again we can set all variables in S_1 to 1 and reduce the problem to a problem with $k - 1$ parts. If S_1 involves no clauses, then it is an isolated part, and its constraints are separate from the rest of the problem. Thus the problem reduces to the union of disjoint problems, it is polynomial if all resulting parts are polynomial and NP-complete otherwise if some part is NP-complete.

THEOREM 10. *All cases of k-partite satisfiability are either in P or NP-complete.*

Acknowledgments. The author had valuable conversations with Yossi Azar and Carlos Subi.

REFERENCES

[1] T. FEDER AND M. Y. VARDI, *The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory*, SIAM J. Comput., 28

(1998), pp. 57–104.
[2] W. Gutjahr, *Graph Colourings*, Ph.D. Thesis, Free University, Berlin, 1991.
[3] W. Gutjahr, E. Welzl, and G. Woeginger, *Polynomial graph colourings*, Discrete Appl. Math., 35 (1992), pp. 29–46.
[4] P. Hell and J. Nešetřil, *On the complexity of H-coloring*, J. Combin. Theory Ser. B, 48 (1990), pp. 92–110.
[5] P. Hell, J. Nešetřil, and X. Zhu, *Duality and polynomial testing of tree homomorphisms*, Trans. Amer. Math. Soc., 348 (1996), pp. 1281–1297.
[6] P. Hell, J. Nešetřil, and X. Zhu, *Complexity of tree homomorphisms*, Discrete Appl. Math., 70 (1996), pp. 23–36.
[7] P. Hell, J. Nešetřil, and X. Zhu, *Duality of graph homomorphisms*, in Combinatorics, Paul Erdös is Eighty, Bolyai Soc. Math. Stud. 2, János Bolyai Math. Soc., Budapest, 1996, pp. 271–282.
[8] P. Hell and X. Zhu, *Homomorphisms to oriented paths*, Discrete Math., 132 (1994), pp. 107–114.
[9] P. Hell and X. Zhu, *The existence of homomorphisms to oriented cycles*, SIAM J. Discrete Math., 8 (1995), pp. 208–222.
[10] T. J. Schaefer, *The complexity of satisfiability problems*, in Proceedings of the 10th ACM Symposium on Theory of Computing, 1978, pp. 216–226.
[11] X. Zhu, *A simple proof of the multiplicativity of directed cycles of prime power length*, Discrete Appl. Math., 36 (1992), pp. 313–315.
[12] X. Zhu, *A polynomial algorithm for homomorphisms to oriented cycles*, J. Algorithms, 19 (1995), pp. 333–345.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 481–497

EQUIREPLICATE BALANCED BINARY CODES FOR OLIGO ARRAYS*

NOGA ALON†, CHARLES J. COLBOURN‡, ALAN C. H. LING§, AND MARTIN TOMPA¶

Abstract. In the manufacture of oligo arrays for DNA hybridization experiments, manufacturing defects must be detected and their position determined. The design of manufacturing protocols for such oligo arrays leads to a combinatorial problem, requiring certain binary codes which have an additional balance property. Constructions using block designs and packings for these codes, within a range of interest in a practical manufacturing application, are developed. The focus is on equireplicate codes, constant weight codes in which every bit position is a one equally often.

Key words. binary code, oligo array, packing, block design, t-design

AMS subject classification. 05B05

PII. S0895480101383895

1. Introduction.

Let X be a set of v *elements* or *points*. Let $\mathcal{B}$ be a collection of b subsets of X, called *blocks*. Then $(X, \mathcal{B})$ is a (v, b)-*set system*. The *block sizes* of $(X, \mathcal{B})$ are the cardinalities of the b blocks in $\mathcal{B}$; when all blocks have cardinality k, the set system is k-*uniform*. We often write (v, b, k)-*set system* to denote a k-uniform (v, b)-set system.

In an application to quality control in the manufacture of oligo arrays described in the next section, certain (v, b, k)-set systems are of particular interest. For each point $x \in X$, we define the *replication number* of x to be the number of blocks containing x. The set system is r-*equireplicate* if every point has replication number r. We call a (v, b, k)-set system d-*discriminated* if, for every point $x \in X$, the replication number r_x satisfies $d \le r_x \le b - d$; and, for every two distinct points $x, y \in X$, the number of blocks containing *exactly one* of x and y is at least d. In other words, if λ_{xy} represents the number of blocks containing both x and y, we require that $r_x + r_y - 2\lambda_{xy} \ge d$. A d-discriminated (v, b, k)-set system is henceforth denoted by (v, b, k, d)-*balanced binary code*, or (v, b, k, d)-*bbc* for short.

Table 1 gives an example of a $(28, 14, 10, 5)$-bbc, which is 5-equireplicate. This was constructed using the method described in [3].

The connection to codes arises as follows. If we form the $b \times v$ incidence matrix of the set system, then each row has weight k and each column has weight at least d and at most $b - d$. Hence each column differs from the all-zero vector and from the all-one vector in at least d positions. Moreover, since two points satisfy $r_x + r_y - 2\lambda_{xy} \ge d$, we

*Received by the editors January 22, 2001; accepted for publication June 8, 2001; published electronically October 4, 2001.

http://www.siam.org/journals/sidma/14-4/38389.html

†Department of Mathematics, Raymond and Beverley Sackler Faculty of Exact Sciences, Tel Aviv University, Ramat Aviv, Tel Aviv, 69978 Israel (noga@math.tau.ac.il). The research of this author was supported by a USA-Israel BSF grant.

‡Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 (Charles.Colbourn@asu.edu). The research of this author was supported by ARO grant DAAG55-98-1-0272 and DOE grant DE-FG02-00ER45828.

§Department of Computer Science, University of Vermont, Burlington, VT 05405 (aling@emba.uvm.edu).

¶Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350 (tompa@cs.washington.edu). The research of this author was supported by NSF grant DBI-9974498.

TABLE 1
A 5-equireplicate $(28, 14, 10, 5)$*-bbc set system.*

0	4	7	9	13	14	18	20	23	27
2	3	4	9	15	16	18	23	25	26
4	6	8	13	15	19	20	21	24	25
0	3	4	5	12	16	19	21	22	27
1	2	6	9	10	13	16	17	21	27
1	5	10	11	13	14	16	19	23	25
3	7	8	9	10	11	12	20	21	25
0	1	2	7	11	15	21	22	23	24
0	2	3	5	8	11	13	17	18	24
0	1	8	9	12	14	15	17	19	26
1	5	6	7	12	18	24	25	26	27
6	11	12	14	15	16	17	18	20	22
3	10	17	19	20	22	23	24	26	27
2	4	5	6	7	8	10	14	22	26

TABLE 2
A 5-equireplicate $(28, 14, 10, 5)$*-bbc incidence matrix.*

1000100101000110001010010001
0011100001000001101000010110
0000101010000101000111001100
1001110000001000100101100001
0110001001100100110001000001
0100010000110110100100010100
0001000111111000000011000100
1110000100010001000001111000
1011010010010100011000001000
1100000011001011010100000010
0100011100001000001000001111
0000001000011011111010100000
0001000000100000010110111011
0010111110100010000000100010

have that every two columns have Hamming distance at least d. Hence the code whose words are the columns together with the all-zero and all-one vectors has minimum distance (at least) d. For the example in Table 1, the matrix is given in Table 2.

The fundamental existence question for balanced binary codes is to determine, for a given v and k, a code with a "small" number b of rows having "large" discrimination d. (See section 2 for the motivation.) To make this precise, given v, k, and d, we seek the smallest value of b for which a (v, b, k, d)-bbc exists. We begin by establishing a lower bound on b.

PROPOSITION 1.1. *If a* (v, b, k, d)*-bbc exists, then* $b \geq \mathbf{max}\left(\left\lceil \frac{vd}{k} \right\rceil, \left\lceil \frac{vd}{v-k} \right\rceil\right)$.

Proof. The incidence matrix of a (v, b, k, d)-bbc contains bk one entries, since each of the b rows contains k ones. Since each of the v columns contains at least d and at most $b - d$ ones, we have

$$vd \leq bk \leq vb - vd.$$

The bounds follow. □

We call a (v, b, k, d)-bbc *optimal* when b realizes the bound in Proposition 1.1. When a (v, b, k, d)-bbc exists, an additional row can easily be appended to form a $(v, b + 1, k, d)$-bbc; in fact, simply duplicating any of the rows produces the extended bbc. It is therefore natural to study the optimal balanced binary codes.

Let $(V, \mathcal{B})$ be a set system. The *complement* of $(V, \mathcal{B})$, denoted by $(V, \overline{\mathcal{B}})$, has the same set V of elements, and the collection of blocks $\overline{\mathcal{B}} = \{V \setminus D : D \in \mathcal{B}\}$.

LEMMA 1.2. *The complement of a (v, b, k, d)-bbc is a $(v, b, v-k, d)$-bbc. The complement of an equireplicate bbc is also equireplicate. The complement of an optimal bbc is also optimal.*

The following lemma gives a simple characterization of optimal equireplicate bbc's.

LEMMA 1.3. *Suppose B is an equireplicate (v, b, k, d)-bbc with replication number r.*

1. *If $v \geq 2k$, B is optimal if and only if $r = d$.*
2. *If $v < 2k$, B is optimal if and only if $r = b - d$.*

Proof. By Lemma 1.2, assume without loss of generality that $v \geq 2k$. Suppose $r = d$. Since $bk = vr$, both being the number of ones in the incidence matrix of B, we have $b = \frac{vd}{k} = \max(\lceil \frac{vd}{k} \rceil, \lceil \frac{vd}{v-k} \rceil)$, making B optimal. Conversely, suppose B is optimal. Then $b = \lceil \frac{vd}{k} \rceil \leq \frac{vd+k-1}{k}$. By the definition of discrimination, all replication numbers of B are at least d, so $d \leq r = \frac{bk}{v} \leq d + \frac{k-1}{v} < d + 1$. Since r is integral, $r = d$. ∎

Sengupta and Tompa [9] observed that if B_1 is a (v, b_1, k, d_1)-bbc and B_2 is a (v, b_2, k, d_2)-bbc, then $\left[\frac{B_1}{B_2}\right]$, the union of the blocks of B_1 and B_2, is a (v, b_1+b_2, k, d_1+d_2)-bbc; we call this operation *addition*. Unfortunately, the addition of two optimal bbc's need not be optimal. The reason is simple. Since the bound in Proposition 1.1 is the next larger integer, it is possible for the addition of B_1 and B_2 to contain one more row than does an optimal bbc, despite the optimality of B_1 and B_2 individually. Nevertheless, the addition proves to be very useful in limiting the ranges of the discrimination to be examined.

PROPOSITION 1.4. *If B_1 is an optimal equireplicate (v, b_1, k, d_1)-bbc and B_2 is an optimal (v, b_2, k, d_2)-bbc, then $\left[\frac{B_1}{B_2}\right]$ is an optimal $(v, b_1 + b_2, k, d_1 + d_2)$-bbc.*

Proof. By Lemma 1.2, assume without loss of generality that $v \geq 2k$. By Lemma 1.3, then, all replication numbers of B_1 are d_1, so $b_1 k = v d_1$. It follows that $b_1 + b_2 = \frac{vd_1}{k} + \lceil \frac{vd_2}{k} \rceil$. However, since $\frac{vd_1}{k}$ is an integer, we have $b_1 + b_2 = \lceil \frac{v(d_1+d_2)}{k} \rceil$, so that the addition is optimal. ∎

For this reason, the critical ingredients in producing optimal balanced binary codes are those that are equireplicate. In this paper, we provide a number of combinatorial constructions for equireplicate optimal bbc's, primarily within a range of practical interest in the study of the manufacture of oligo arrays. In a companion paper [3], we examine heuristic techniques which we have used for the production of optimal bbc's in the case when replication numbers are not all equal. Combining these techniques yields a powerful existence result for balanced binary codes in the intended application.

An understanding of the application is critical to motivating both the definitions given and to describing the specific bbc's sought. We provide a brief overview of the biotechnology application before pursuing the construction of optimal bbc's. For full details on the application, see Sengupta and Tompa [9].

2. The quality control problem. For this discussion, a *DNA molecule* can be abstracted as a string over the alphabet $\{A, C, G, T\}$. An *oligo array* is a small chip containing approximately 100,000 *spots*, to each of which is attached its own synthesized DNA molecule. Oligo arrays are used to measure how much of each gene

product is produced by a given cell type under given conditions. For more information on oligo arrays, see, for example, Lipshutz et al. [5].

Our application is in the manufacture of oligo arrays rather than their subsequent use. An array is manufactured in a series of steps "labeled" $A, C, G, T, A, C, G, T, A, \ldots$. Initially, every spot's DNA molecule is empty. In preparation for any given step, an arbitrary subset of the spots can be *masked*. If the step is labeled σ, only a spot that is unmasked will have σ appended to the end of its DNA molecule. By appropriate construction of the masks, each spot can be designed to contain an arbitrary DNA sequence.

The manufacturing process is subject to two different sorts of faults: (1) several individual spots may fail, and (2) an entire manufacturing step may fail, affecting all spots unmasked during that step. The goal of quality control is to identify any single failed step, even if e individual spots fail, where e is a parameter of the manufacturing process. A small number of spots on the chip can be used for this quality control purpose.

Hubbell and Pevzner [4] first investigated this problem. The clever idea underlying their approach is to manufacture identical DNA molecules at multiple spots, using different schedules of steps. If no step fails, all such spots should behave identically. If some step fails, the spots behaving incorrectly hopefully provide a "signature" that identifies the failed step.

The problem Hubbell and Pevzner left open was how to design the quality control molecules and schedules to guarantee such signatures, even in the presence of e faulty spots. Sengupta and Tompa [9] reduced this problem to the design of well-discriminated balanced binary codes as described below and supplied an initial collection of good balanced codes.

First they abstracted the quality control problem as that of designing a *QC matrix* Q, which is a 0-1 matrix with a row for each quality control spot, a column for each manufacturing step, and $Q_{ij} = 1$ if and only if spot i is unmasked during step j. Given the spots that subsequently behave incorrectly as a column vector I, identifying the failed step corresponds roughly to finding the column of Q that resembles I with up to e exceptions. Although this resembles the familiar error-correcting code problem, what makes it more complicated is that (1) one cannot compare the behaviors of spots with different DNA sequences, and (2) even for the spots with identical sequences, it may not be possible to distinguish between all such spots behaving correctly and all such spots behaving incorrectly.

In terms that are beyond our scope, but are detailed by Sengupta and Tompa [9], the properties of a good QC matrix Q are as follows:

1. The set of DNA molecules manufactured at the quality control spots "hybridize poorly" to themselves and each other.
2. Q has high "separation" $\mathrm{sep}(Q)$, which ensures sufficient coverage of each step, and sufficient difference between steps to identify the failed step. Sengupta and Tompa proved that $\mathrm{sep}(Q) \geq 2e + 1$ is sufficient to identify any single failed step, even in the presence of e arbitrarily faulty spots.

Sengupta and Tompa [9] designed QC matrices with these properties using a product construction. First they handcrafted some *QC blocks*, which are small QC matrices. An example of a pair of 4×4 QC blocks from their paper is given in Figure 1. They then showed that a certain cross product of any well-discriminated balanced binary code and any QC block yields a QC matrix with the desired properties above. More specifically, if B is a (v, b, k, d)-bbc, then alternately replacing the ones in each

A	C		
		G	T
A			T
	C	G	

A			T
	C	G	
A	C		
		G	T

FIG. 1. *A pair of* 4×4 *QC blocks. For ease of visualization, the figure shows blanks instead of zeros and the manufacturing step's label instead of a one.*

row of B by the two 4×4 QC blocks of Figure 1, and replacing the zeros in B by 4×4 matrices of zeros, produces a $4b \times 4v$ QC matrix Q for which each DNA molecule has length $2k$, the set of DNA molecules hybridizes poorly, and $\text{sep}(Q) = 2d$. An example of this product construction is shown in Figure 2.

This then explains the design problem of section 1. Since the array manufacturer specifies the number of steps ($4v$) and the molecule lengths ($2k$), and the goal is to minimize the number of quality control spots ($4b$) and maximize separation ($2d$), the resulting balanced binary code design problem is to minimize b and maximize discrimination d for a given v and k. For the current photolithographic process, reasonable ranges for the parameters are $16 \leq 2k \leq 20$, $60 \leq 4v \leq 136$, and $4b$ up to a few hundred.

Although Sengupta and Tompa [9] supplied an initial collection of balanced binary codes, they left open the construction of optimal balanced binary codes for arbitrary choices of v, k, and d. The current paper addresses exactly this problem for the relevant parameter ranges given above. The resulting constructions are summarized in Tables 7 and 8.

3. Primal constructions. In this section, we examine constructions for the bbc set system; to distinguish from later constructions, we call this the *primal* set system. Our constructions begin with a useful connection to balanced incomplete block designs. A t-(v, b, r, k, λ) *design* is a pair $(V, \mathcal{B})$ where V is a set of v *elements*, and $\mathcal{B}$ is a collection of k-element subsets of V called *blocks*. Every t-subset of V appears as a subset of exactly λ of the b blocks in $\mathcal{B}$. It follows that every s-subset for $0 \leq s \leq t$ appears in the same number λ_s of blocks (since the block sizes all equal k). In this notation, $b = \lambda_0$, $r = \lambda_1$, and $\lambda = \lambda_t$. When $t = 2$, a t-design is a *balanced incomplete block design*, or simply a *block design*. The connection to bbc's is immediate.

THEOREM 3.1. *When* $v > k > 2$, *every* 2-(v, b, r, k, λ) *design is an optimal equireplicate* $(v, b, k, \min(r, b - r))$-*bbc.*

Proof. The design is a (v, b, k)-set system by construction. To verify that it is $\min(r, b - r)$-discriminated, we observe that the number of blocks containing exactly one of (any) two distinct elements is $2(r - \lambda)$. By Lemma 1.2, we can assume without loss of generality that $v \geq 2k$. Then $2(r - \lambda) \geq r$ since $r = \frac{\lambda(v-1)}{k-1}$. Optimality follows from Lemma 1.3 and the observation that $d = \min(r, b - r) = r$, since $r = bk/v \leq b/2$. ∎

COROLLARY 3.2. *There are equireplicate* $(16, 30, 8, 15)$-, $(18, 34, 9, 17)$-, *and* $(20, 38, 10, 19)$-*bbc's.*

Proof. There exist 2-$(16, 30, 15, 8, 7)$, 2-$(18, 34, 17, 9, 8)$, and 2-$(20, 38, 19, 10, 9)$ designs. (See, e.g., [7].) The first and last are *Hadamard designs* arising from Hadamard matrices; see [1]. ∎

Block designs have been very extensively studied, and much is known about their existence; see [7] for a table giving known existence results for "small" values of r. For

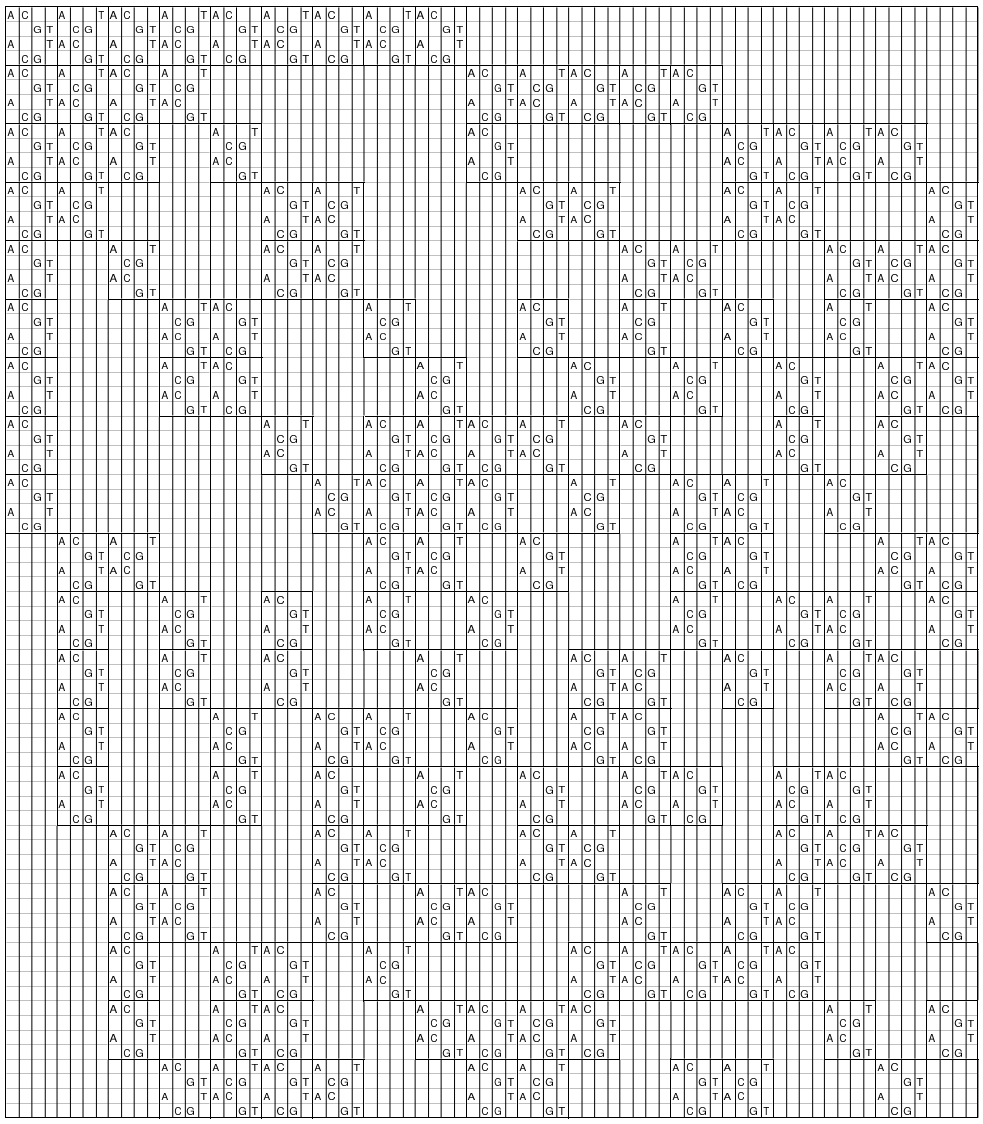


FIG. 2. *The product of a* $(19, 19, 9, 9, 4)2$*-design and the pair of* 4×4 *QC blocks of Figure* 1*, resulting in a* 76×76 *QC matrix* Q *with minimum separation* $sep(Q) = 18$.

our application, the conditions on block designs are too stringent. Indeed, in a block design, every two elements have the property that there are *exactly* $2(r-\lambda)$ blocks containing precisely one of them, and the application does not require this type of uniformity. Consequently, block designs provide only a small fraction of the bbc's needed, even among the optimal equireplicate cases. A more serious drawback arises since b is constrained to be at least v by Fisher's inequality. (See, for example, [1].) Using addition, however, we are most interested in bbc's with b very small.

We therefore relax the requirements by allowing, for each pair of elements, the number of blocks containing exactly one of them to vary, provided that it remains at least d. Translating to the design vernacular, when the bbc is equireplicate, we are specifying that every pair of elements occur together in at most some number λ of blocks.

A t-(v,k,λ) *packing* $(V,\mathcal{B})$ is a (v,b,k)-set system in which every t-subset of elements occurs together in at most λ of the blocks in $\mathcal{B}$. A 2-(v,k,λ) packing in which $v \geq 2k$ and every element has replication number at least r yields a bbc which is $\min(r, 2(r-\lambda))$-discriminated. See [8] for a survey of packings.

Our first construction produces 2-(v,k,λ) packings with $b = v$. We take, as the set of elements, the integers modulo v, $\mathbb{Z}_v$. We choose a single block, B, containing k elements and form $\mathcal{B} = \{B+0, \ldots, B+(v-1)\}$, where the translate $B+i = \{x+i \bmod v : x \in B\}$. To determine the index λ of the packing $(\mathbb{Z}_v, \mathcal{B})$, proceed as follows. Each pair $\{i,j\}$ of elements has an associated *difference* modulo v, namely, $\min(i-j \bmod v, j-i \bmod v)$. If this difference appears as the difference between two elements of B, then the pair occurs in exactly one translate of these two elements unless the difference is precisely half of v, in which case the pair appears in two translates. Hence, to determine the maximum number of times that a pair occurs in the packing, we need only determine how many pairs of elements in B have a specified difference. To handle the case when v is even and the difference examined is $d/2$, we must double the number of occurrences of the difference.

In the construction of bbc's, we may not require the minimum possible value of λ. Indeed, if $v \geq 2k$ and we are to produce a (v,v,k,k)-bbc, we require only that every difference appear at most $\lfloor k/2 \rfloor$ times. A single block of k elements from $\mathbb{Z}_v$ in which every difference is represented at most $\lfloor k/2 \rfloor$ times (except when v is even, we require that $v/2$ be represented at most $\lfloor v/4 \rfloor$ times) is a *near difference set*. When v is odd and every difference is represented the same number of times, the block is a *cyclic difference set*, and these have been studied extensively [1].

In Table 3, we present near difference sets for a number of parameters of interest. These solutions were found using a simple backtracking method.

Such bbc's arising from near difference sets can exist only for some of the parameter sets of interest, namely, those when $b = v$. We therefore examine a more general method. Again we take $\mathbb{Z}_v$ as the set of elements. We form a number of *base blocks* $B_1, B_2, \ldots, B_\ell$. We can again develop each base block modulo v to form v blocks. For certain base blocks, the v blocks in the development are not all distinct. In these cases, we can choose to include only a subset of the blocks. Suppose, for example, that v and k are both even, and that $B_i = \{b_1, \ldots, b_{k/2}, b_1+(v/2), \ldots, b_{k/2}+(v/2)\}$, with $0 \leq b_i < v/2$ when $1 \leq i \leq k/2$. Then $B_i + (v/2) = B_i$. In this case, we can produce only $v/2$ blocks, a *half orbit*, by including $B_i + j$ for $j = 0, \ldots, (v/2)-1$. In Table 4, we present solutions containing one half orbit and one starter block generating v blocks. To prescribe the block for the half orbit, we give only the elements $b_1, \ldots, b_{k/2}$.

TABLE 3
Near difference sets.

v	k	d	Block	v	k	d	Block
9	8	1	0 1 2 3 4 5 6 7	10	8	2	0 1 2 3 4 5 6 7
10	9	1	0 1 2 3 4 5 6 7 8	11	8	3	0 1 2 3 4 5 6 8
11	9	2	0 1 2 3 4 5 6 7 8	11	10	1	0 1 2 3 4 5 6 7 8 9
12	9	3	0 1 2 3 4 5 6 7 9	12	10	2	0 1 2 3 4 5 6 7 8 9
13	8	5	0 1 2 3 4 5 8 10	13	9	4	0 1 2 3 4 5 7 9 10
13	10	3	0 1 2 3 4 5 6 7 8 10	14	8	6	0 1 2 3 4 5 7 10
14	9	5	0 1 2 3 4 5 6 9 11	15	8	7	0 1 2 3 5 7 8 11
15	9	6	0 1 2 3 4 5 6 8 11	15	10	5	0 1 2 3 4 5 6 7 10 12
16	8	8	0 1 2 3 4 7 9 12	16	9	7	0 1 2 3 4 6 7 11 13
16	10	6	0 1 2 3 4 5 6 7 9 12	17	8	8	0 1 2 3 4 6 9 13
17	9	8	0 1 2 3 4 5 8 10 13	17	10	7	0 1 2 3 4 5 7 8 11 13
19	8	8	0 1 2 3 4 6 9 13	19	9	9	0 1 2 3 5 7 12 13 16
19	10	9	0 1 2 3 5 7 12 13 15 16	20	9	9	0 1 2 3 4 7 9 12 16
20	10	10	0 1 2 3 4 6 8 11 14 15	21	8	8	0 1 2 3 5 8 12 16
21	9	9	0 1 2 3 4 7 9 13 18	21	10	10	0 1 2 3 4 5 8 10 13 17
22	9	9	0 1 2 3 4 6 9 13 17	22	10	10	0 1 2 3 4 5 8 10 13 17
23	8	8	0 1 2 3 5 8 12 16	23	9	9	0 1 2 3 4 6 9 13 17
23	10	10	0 1 2 3 4 5 7 10 14 18	24	9	9	0 1 2 3 4 6 9 13 17
24	10	10	0 1 2 3 5 6 11 13 17 20	25	8	8	0 1 2 3 5 8 12 16
25	9	9	0 1 2 3 4 6 9 13 17	26	9	9	0 1 2 4 6 11 12 20 23
26	10	10	0 1 2 3 4 7 9 12 16 20	27	8	8	0 1 2 3 5 8 12 16
27	10	10	0 1 2 3 4 6 9 13 17 22	28	9	9	0 1 2 3 5 8 12 16 21
29	8	8	0 1 2 3 5 8 12 16	29	9	9	0 1 2 3 5 8 12 16 22
29	10	10	0 1 2 3 4 6 9 13 17 23	30	9	9	0 1 2 3 5 8 12 16 21
31	8	8	0 1 2 4 7 12 16 25	31	9	9	0 1 2 3 5 8 12 16 21
31	10	10	0 1 2 3 4 6 9 13 17 22	32	9	9	0 1 2 3 5 8 12 16 22
33	8	8	0 1 2 4 7 11 19 24	33	9	9	0 1 2 3 5 8 12 16 21
33	10	10	0 1 2 3 5 8 12 18 22 27	34	9	9	0 1 2 3 5 8 12 16 21

TABLE 4
One and a half orbits.

v	k	d	Half orbit	Full orbit
10	8	3	0 1 2 3	0 1 2 3 4 5 6 7
12	10	3	0 1 2 3 4	0 1 2 3 4 5 6 7 8 9
14	8	9	0 1 2 4	0 1 2 3 4 6 7 12
16	10	9	0 1 2 3 4	0 1 2 3 4 5 7 8 10 14
22	10	15	0 1 2 3 5	0 1 2 3 5 7 10 15 18 19
24	10	15	0 1 2 4 9	0 1 2 3 6 7 9 11 17 20
26	10	15	0 1 2 4 7	0 1 2 3 4 7 10 12 18 22

Other relaxations of the stringent block design conditions can be exploited. A $(g, k; \lambda)$-*difference matrix* over $\mathbb{Z}_g$ is a $k \times \lambda g$ array A with entries from $\mathbb{Z}_g$ with the property that for any $1 \le i < j \le k$, the collection of differences $\{A_{i,\ell} - A_{j,\ell} \bmod g : 1 \le \ell \le \lambda g\}$ contains the g numbers in $\mathbb{Z}_g$ λ times each.

PROPOSITION 3.3. *There is an equireplicate* $(27, 21, 9, 7)$*-bbc and an equireplicate* $(30, 21, 10, 7)$*-bbc.*

Proof. There is a $(3, 9; 3)$-difference matrix; see for example, [2]. Choose any seven of its columns and append the 14 further columns obtained by developing the columns under addition modulo 3. Treat the resulting set of 21 columns as blocks of a packing on the 27 points (i, σ), where i indicates the row and σ the symbol from $\mathbb{Z}_3$. The resulting packing has $\lambda = 3$, and hence is a 2-$(27, 9, 3)$ packing on 21 blocks which is equireplicate. Hence, an equireplicate $(27, 21, 9, 7)$-bbc exists. Now this 2-$(27, 9, 3)$ packing can, by construction, be partitioned into seven sets of three blocks

each, so that each set contains three mutually disjoint blocks. Let $P_1, \ldots, P_7$ be such a partition of the blocks. Add three new elements a, b, c to the packing. Add a to each block in P_1 and P_2 and to the first block in P_7; add b to each block in P_3 and P_4 and to the second block in P_7; add c to the remaining seven blocks. The result is a 2-$(30, 10, 3)$ packing; it is equireplicate with replication number 7, and hence yields an equireplicate $(30, 21, 10, 7)$-bbc. □

4. Dual constructions. Since we are primarily interested in cases in which $b < v$, it is natural to consider the dual set system. The *dual set system* of a set system $(V, \mathcal{B})$ is a set system $(X, \mathcal{D})$ in which $X = \{x_B : B \in \mathcal{B}\}$ and $\mathcal{D} = \{D_y : y \in V\}$, where $D_y = \{x_B : y \in B \in \mathcal{B}\}$. The dual of a (v, b, k)-set system with replication numbers $r_1, \ldots, r_v$ is a (b, v)-set system with v blocks of sizes $r_1, \ldots, r_v$ and having constant replication number k. Indeed, when the (v, b, k)-set system is equireplicate with replication number r, its dual is a (b, v, r)-set system which has constant replication number k. The dual of the set system in Table 1 is given in Table 5.

TABLE 5
Dual set system of $(28, 14, 10, 5)$*-bbc.*

0	3	7	8	9	4	5	7	9	10	1	4	7	8	13
1	3	6	8	12	0	1	2	3	13	3	5	8	10	13
2	4	10	11	13	0	6	7	10	13	2	6	8	9	13
0	1	4	6	9	4	5	6	12	13	5	6	7	8	11
3	6	9	10	11	0	2	4	5	8	0	5	9	11	13
1	2	7	9	11	1	3	4	5	11	4	8	9	11	12
0	1	8	10	11	2	3	5	9	12	0	2	6	11	12
2	3	4	6	7	3	7	11	12	13	0	1	5	7	12
2	7	8	10	12	1	2	5	6	10	1	9	10	12	13
0	3	4	10	12										

The discrimination of the primal is reflected in the dual in a somewhat different manner than in the primal. Two blocks of the dual sharing μ elements result in a discrimination d of the primal satisfying $d \leq 2r - 2\mu$; hence maximizing d amounts to minimizing μ, the intersection size of two blocks, since r is fixed. Translating this into design vernacular, we establish the following.

THEOREM 4.1. *A* t-$(b, r, 1)$ *packing on* v *blocks with replication number* k *yields an equireplicate* $(v, b, k, \min(r, b - r, 2(r - t + 1)))$*-bbc with replication number* r.

Proof. The dual of a t-$(b, r, 1)$ packing on v blocks with replication number k is a (v, b, k)-set system with replication number r in which every pair of elements occurs in at most $t - 1$ blocks together. □

Hence, our goal is to produce t-$(b, r, 1)$ packings with $t \leq r/2 + 1$. One potential benefit of this dual approach when $b < v$ is that we can examine constructions over $\mathbb{Z}_b$ rather than the larger $\mathbb{Z}_v$. We illustrate this by producing a number of 4-equireplicate $(2m, m, 8, 4)$-bbc's.

THEOREM 4.2. *A* 4*-equireplicate* $(2m, m, 8, 4)$*-bbc exists for all* $m \geq 10$.

Proof. The dual set system is constructed with elements in $\mathbb{Z}_m$ and has two base blocks which are developed modulo m. We need only ensure that the result is a 3-$(m, 4, 1)$ packing. When $m = 10$, use the base blocks $\{0, 1, 2, 6\}$ and $\{0, 2, 4, 7\}$; when $m \geq 11$, use the base blocks $\{0, 1, 2, 7\}$ and $\{0, 1, 3, 5\}$. The proof is completed by verifying that no translate of a triple in either base block appears as a translate of a different triple or as a different translate of this triple. □

In a similar vein, other bbc's are easily produced from 3-$(b, 5, 1)$ packings:

v	b	k	d	Dual base blocks in $\mathbb{Z}_b$
30	15	10	5	$\{0,1,4,11,14\}$, $\{0,2,7,8,13\}$
32	16	10	5	$\{0,2,8,14,15\}$, $\{0,3,7,11,14\}$
34	17	10	5	$\{0,3,10,12,14\}$, $\{0,4,12,15,16\}$

The dual solutions thus far presented all have the property that v is an integral multiple of b. We can vary the construction to admit other solutions. Suppose, for example, that we are to produce a $(25, 10, 10, 4)$-bbc. Its dual is a $(10, 25, 4)$-set system which is 10-equireplicate and forms a 3-$(10, 4, 1)$ packing. Two base blocks, $\{0, 1, 2, 6\}$ and $\{0, 2, 4, 7\}$, generate 20 blocks in $\mathbb{Z}_{10}$. A third base block $\{0, 1, 3, 4\}$ is used, but in its development, we include only translates obtained by adding the *even* integers. Since this last base block contains two even and two odd numbers, this development ensures that the resulting packing is 10-equireplicate.

In general, by selecting certain translates out of one orbit of a base block, we can vary k and b in the construction. We give some further examples of constructions of this type next, subscripting one block with the integers to be added in forming its translates. The first three employ packings with $t = 3$, while the last five employ packings with $t = 4$.

v	b	k	d	Dual base blocks in $\mathbb{Z}_b$
27	12	9	4	$\{0,1,3,5\}$, $\{0,1,2,7\}$, $\{0,3,6,9\}_{0,1,2}$
30	12	10	4	$\{0,1,3,5\}$, $\{0,1,2,7\}$, $\{0,2,6,8\}_{0,1,2,3,4,5}$
32	20	8	5	$\{0,1,8,14,17\}$, $\{0,2,11,18,19\}_{0,1,2,5,6,7,10,11,12,15,16,17}$
15	10	9	4	$\{2,3,5,6,7,9\}$, $\{3,4,5,6,8,9\}_{0,2,4,6,8}$
28	21	8	6	$\{0,1,4,9,18,20\}$, $\{0,1,7,8,14,15\}_{0,1,2,3,4,5,6}$
33	22	9	6	$\{0,1,6,7,10,15\}$, $\{0,1,3,11,12,14\}_{0,1,2,3,4,5,6,7,8,9,10}$
24	21	8	7	$\{0,1,2,4,6,7,14\}$, $\{0,3,6,9,12,15,18\}_{0,1,2}$
32	28	8	7	$\{0,1,2,4,7,11,17\}$, $\{0,4,8,12,16,20,24\}_{0,1,2,3}$

In a number of cases, we have not been able to find (dual) solutions which are cyclic modulo b. In some of these situations, we have resorted to using a smaller group.

THEOREM 4.3. *There is a* $(3m, 2m, 9, 6)$*-bbc for all* $m \geq 7$.

Proof. We form the dual of the required bbc on the element set $\mathbb{Z}_m \times \{0, 1\}$. We begin with three base blocks $\{(0,0),(1,0),(3,0),(0,1),(1,1),(3,1)\}$, $\{(2,0),(4,0),(5,0),(6,0),(0,1),(3,1)\}$, and $\{(0,0),(3,0),(2,1),(4,1),(5,1),(6,1)\}$. Each gives m blocks of the dual by adding the nonzero elements of $\mathbb{Z}_m$ in turn to the first coordinates of each element. It is easily verified that the result is a 3-$(2m, 6, 1)$ packing which is 9-equireplicate. □

THEOREM 4.4. *There is a* $(4m, 3m, 8, 6)$*-bbc and a* $(5m, 3m, 10, 6)$*-bbc for all* $m \geq 5$.

Proof. We form the dual of the required bbc on the element set $\mathbb{Z}_m \times \{0, 1, 2\}$. We begin with five base blocks:

$$\{(0,0),(1,0),(2,0),(3,0),(4,1),(4,2)\},$$
$$\{(0,1),(1,1),(2,1),(3,1),(4,0),(4,2)\},$$
$$\{(0,2),(1,2),(2,2),(3,2),(4,1),(4,0)\},$$
$$\{(0,0),(1,0),(0,1),(1,1),(0,2),(1,2)\},$$
$$\{(0,0),(2,0),(0,1),(2,1),(0,2),(2,2)\}.$$

Each gives m blocks of the dual by adding the nonzero elements of $\mathbb{Z}_m$ in turn to the first coordinates of each element. It is easily verified that the result is a 4-$(3m, 6, 1)$ packing and yields a $(5m, 3m, 10, 6)$-bbc. Deleting the last base block and its translates yields a $(4m, 3m, 8, 6)$-bbc. □

PROPOSITION 4.5. *There exists a* $(24, 15, 8, 5)$*-bbc and a* $(27, 15, 9, 5)$*-bbc.*

Proof. The point set for the dual in each case is the 15 points $\mathbb{Z}_{12} \cup \{a, b, c\}$. Start with the blocks obtained by developing $\{0, 1, 2, 4, 9\}$ and $\{0, 1, 5\}$ modulo 12. Then the translates of $\{0, 1, 5\}$ can be partitioned into three parallel classes of four blocks each. For the three parallel classes in turn, add the points $\{a, b\}$, $\{a, c\}$, and $\{b, c\}$, respectively, to each block of the parallel class. The result is the dual of the $(24, 15, 8, 5)$-bbc.

To this dual, add the three distinct translates of $\{0, 3, 6, 9\}$ modulo 12, placing a, b, and c, respectively, in one of the three translates. This is the dual of the $(27, 15, 9, 5)$-bbc. □

For small values of d, a direct construction can be quite simple.

PROPOSITION 4.6. *There are* $(12, 6, 8, 2)$*-,* $(12, 8, 9, 2)$*-,* $(14, 7, 10, 2)$*-, and* $(15, 6, 10, 2)$*-bbc's.*

Proof. Start with a k-regular graph on n vertices for $(k, n) = (4, 6)$, $(3, 8)$, $(4, 7)$, or $(5, 6)$, respectively. The complement of this set system forms the dual of the required bbc. □

Similarly, the complement of the blocks of a 2-$(9, 12, 4, 3, 1)$ design forms the dual of a $(12, 9, 8, 3)$-bbc.

We employ some constructions from Hadamard designs. A *Hadamard* 3*-design* is a 3-$(4n, 2n, n-1)$ design [1]. Such a design has $8n-2$ blocks, and they occur in $4n-1$ complementary pairs. Deleting one point of a 3-$(4n, 2n, n-1)$ design produces a 2-$(4n-1, 2n-1, n-1)$ design which has $4n-1$ blocks and replication number $2n-1$. Hence, the 2-design is *symmetric*, and consequently every two blocks of the 2-design intersect in $n-1$ elements. The 3-design can be recovered from the 2-design by including the complements of the blocks of the 2-design and including the blocks with a single new element which is adjoined to each. From this construction, the 3-design is an $(n+1)$-$(4n, 2n, 1)$ packing. Deleting blocks retains this packing property, but, more importantly, deleting complementary pairs of blocks retains the property that the packing is equireplicate. Indeed, if we select j complementary pairs of blocks, the replication number is j; when $j \geq 2n$, the packing leads to a $(2j, 4n, j, 2n)$-bbc. Using Hadamard designs for $n \in \{3, 4, 5\}$, we obtain the following.

PROPOSITION 4.7. *There exist* $(16, 12, 8, 6)$*-,* $(16, 20, 8, 10)$*-,* $(18, 12, 9, 6)$*-,* $(18, 16, 9, 8)$*-,* $(18, 20, 9, 10)$*-,* $(20, 12, 10, 6)$*-, and* $(20, 16, 10, 8)$*-bbc's.*

We also need one specific construction.

PROPOSITION 4.8. *There exist* $(18, 9, 8, 4)$*- and* $(18, 9, 10, 4)$*-bbc's.*

Proof. The second is the complement of the first. To construct the dual of the first, we begin with nine points $\{(i, j) : i, j \in \mathbb{Z}_3\}$. We include all nine blocks of the form $\{(i, k), (i, \ell), (j, k), (j, \ell)\}$ with $i, j, k, \ell \in \mathbb{Z}_3$, $i \neq j$, and $k \neq \ell$. We then add all nine blocks of the form $\{(i, j), (i, k), (a, \ell), (b, \ell)\}$ when $\{i, a, b\} = \{j, k, \ell\} = \mathbb{Z}_3$. This is a 3-$(9, 4, 1)$ packing with 18 blocks, having constant replication number eight. □

In Table 6, the dual of a $(16, 38, 8, 19)$-bbc is presented. The method used to obtain this solution is of independent interest and is described in [3].

5. Nonexistence results. We have presented a large collection of constructions for optimal equireplicate bbc's, focusing on those with smaller discriminations in order to use addition to produce those with larger discrimination. However, not all bbc's exist; in fact, those with low discrimination appear to be the least likely to exist. We do not restrict to equireplicate bbc's in this section. We establish a preliminary result for small discrimination.

THEOREM 5.1. *An optimal* $(v, b, k, 1)$*-bbc exists only when* $v = k + 1$ *or* $k = 1$.

TABLE 6
Dual of a $(16, 38, 8, 19)$*-bbc.*

0	4	6	7	8	9	10	11	13	14	17	19	21	27	28	29	32	35	37
0	2	4	5	8	9	15	16	17	18	19	20	22	25	27	29	31	32	34
0	1	5	6	14	16	18	19	20	23	24	26	27	28	30	31	34	35	37
0	3	7	8	12	13	16	18	20	21	26	28	29	30	31	32	33	35	36
1	2	5	6	7	9	11	13	20	21	22	24	25	30	31	32	34	35	36
1	2	3	4	5	10	13	14	15	16	20	21	24	27	29	30	32	33	37
0	1	3	4	5	7	11	12	14	17	19	20	22	23	32	33	34	36	37
2	5	6	7	8	9	12	14	15	16	19	23	25	29	30	33	35	36	37
1	2	3	4	6	12	13	15	17	23	25	26	28	29	31	32	34	35	37
2	3	6	7	10	11	15	16	17	18	20	21	22	23	27	28	31	36	37
3	4	5	6	9	10	11	12	18	19	22	24	26	27	29	31	33	35	36
0	1	2	8	10	11	12	13	15	17	19	24	27	28	30	31	33	34	36
9	10	11	13	14	15	18	19	20	21	22	23	25	26	28	29	30	33	34
1	3	5	7	8	9	10	11	12	14	15	16	17	18	24	25	26	28	32
0	2	3	4	7	8	10	12	14	21	22	23	24	25	26	27	30	34	35
0	1	4	6	8	9	13	16	17	18	21	22	23	24	25	26	33	36	37

Proof. If $v \geq 2k$, the dual set system has $b = \lceil v/k \rceil$ points and has at least $v - k + 1$ blocks of size 1. When $k > 1$, some block is repeated, and hence the discrimination is 0. If $v < 2k$, the dual set system has $\lceil v/(v-k) \rceil$ points and has at least $k + 1$ blocks of size $v - 1$. Its complement therefore has $k + 1$ blocks of size 1, and hence contains a repeated block unless $v - k = 1$. □

When the discrimination is two, the analysis is slightly more complex. We describe one concrete example and then give much briefer arguments thereafter. Let us establish that a $(31, 8, 8, 2)$-bbc does not exist. If one were to exist, its dual has eight points. It has 31 blocks, and each must have size at least two (and at most six). There are $64 = 8 \cdot 8$ occurrences of points in blocks. Hence, there are either 30 blocks of size two and one of size four, or there are 29 blocks of size two and two of size three. In this case, since $\binom{8}{2} = 28$, there must be a *repeated* block of size two. However, then the bbc has two identical columns, and its discrimination is zero, a contradiction. In general, the nonexistence results all arise from an analysis of the cases that can arise, showing that each cannot have the required discrimination.

THEOREM 5.2. *An optimal* $(v, b, k, 2)$*-bbc exists only if*

1. $v \leq 13$, $v \in \{29, 30\}$, *or* $v \geq 33$ *when* $k = 8$*;*
2. $v \leq 14$, $v \in \{32, 37, 38\}$, *or* $v \geq 41$ *when* $k = 9$*;*
3. $v \leq 16$, $v \in \{46, 47\}$, *or* $v \geq 51$ *when* $k = 10$.

Proof. First we suppose that $v \geq 2k$. Then $b = \lceil \frac{2v}{k} \rceil$. The dual of the required bbc therefore has bk occurrences of elements distributed across v blocks, each having size at least two. It follows that "most" blocks have size equal to two. If the dual has a block of size three, then no block of size two can share both elements with the block of size three. To maximize the number of blocks in the dual, we therefore construct the dual with the largest possible number of blocks of size four and the remaining blocks of size two.

Consider the case when $k = 8$. Write $v = 4s + \alpha$ with $\alpha \in \{1, 2, 3, 4\}$. Then $b = 2s + 1$, and $bk = 16s + 8$. It follows that the number of blocks of size two in the dual, when no blocks of size three are chosen, is at least $4s - 4 + 2\alpha$. Now requiring that $4s - 4 + 2\alpha \leq \binom{b}{2}$, we obtain that $s(s-7) \geq 4\alpha - 8$. Hence, $s \geq 7$ when $\alpha \in \{1, 2\}$ and $s \geq 8$ when $\alpha \in \{3, 4\}$. When $k = 9$ or $k = 10$, the analysis is similar and is omitted.

When $v < 2k$, we use the fact that a $(v, b, k, 2)$-bbc is equivalent to a $(v, b, v-k, 2)$-

bbc. The remaining cases have $b = 5$ but require more than 10 blocks of size two in the dual of the complementary bbc. □

The restrictions when the discrimination is three are more severe. In this case, the dual has $\lceil \frac{3v}{k} \rceil$ points, and its v blocks are almost all of size three. However, two blocks of size three are permitted to intersect in only one element. This easily establishes that when $k \in \{8, 9, 10\}$ and $2k \leq v \leq 34$, no optimal $(v, b, k, 3)$-bbc exists. When $v < 2k$, a similar argument excludes $v \in \{13, 14, 15\}$ when $k = 8$; $v \in \{15, 16, 17\}$ when $k = 9$; and $v \in \{15, 16, 17, 18, 19\}$ when $k = 10$.

Turning to discrimination four, the blocks of size four in the dual form a packing in which every 3-subset appears in at most one block. Using this fact, we can conclude that no optimal $(v, b, k, 4)$-bbc exists when $v \in \{15, 16, 17\}$ and $k = 8$; $v \in \{17, 18, 19, 20\}$ and $k = 9$; or $v \in \{19, 20, 21, 22\}$ and $k = 10$. For example, when $(v, k) \in \{(16, 8), (18, 9), (20, 10)\}$, the dual is a 3-$(8, 4, 1)$ packing with 16, 18, or 20 blocks; but the maximum packing has only 14 blocks.

For discrimination five, the blocks of size five again form a packing in which every 3-subset appears in at most one block. When $(v, k) \in \{(17, 8), (19, 9), (21, 10), (22, 10)\}$, the dual has 11 points and has at least 14 blocks of size five. Then consider the *derived* design obtained by choosing a point containing the maximum number of blocks of size five, selecting all blocks of size five containing this point, and then deleting the point from each. This is a 2-$(10, 4, 1)$ packing, which must have at least seven blocks by construction. But no 2-$(10, 4, 1)$ packing with seven blocks exists. By complementation, we also eliminate the cases when $(v, k) = (17, 9)$ or $(19, 10)$. A similar argument shows that no $(24, 12, 10, 5)$-bbc or $(26, 13, 10, 5)$-bbc exists. A complete exhaustive search by backtracking established the nonexistence of a $(19, 12, 8, 5)$-bbc.

The astute reader will have observed that fewer negative results arise for even discrimination than for odd, and that as the discrimination increases, the negative results are sparser. Indeed, in Tables 7 and 8 there are very few negative results for $d > 5$. It is, however, possible to prove such results. We give examples in the following two theorems.

THEOREM 5.3. *A $(2k, 2d, k, d)$-bbc does not exist when d is odd and $d < 2k - 1$.*

Proof. Such a bbc is a 2-$(2k, 2d, \lfloor d/2 \rfloor)$ packing. Hence, we require that $\lfloor d/2 \rfloor \cdot \binom{2k}{2} \geq 2d \cdot \binom{k}{2}$. Letting $d = 2s + 1$, we require that $s(2k - 1) \geq (2s + 1)(k - 1)$. Simplifying, $2ks - s \geq 2ks + k - 2s - 1$, i.e., $s \geq k - 1$, or $d \geq 2k - 1$. □

For even values of d, there is also a nonexistence result.

THEOREM 5.4. *A $(2k, 2d, k, d)$-bbc does not exist when $d < k/2$.*

Proof. The columns of such a bbc are $2k$ binary vectors of length $2d$ so that the Hamming distance between any pair is at least d. By the pigeonhole principle, k of them share the same first coordinate, giving a set of k vectors of length $2d - 1$ so that the Hamming distance between every pair is at least d. Since in each coordinate there are at most $\lfloor k^2/4 \rfloor$ pairs of these vectors that differ in this coordinate, and the sum of distances between all pairs of these vectors is at least $\binom{k}{2} d$, it follows that

$$(2d - 1)k^2/4 \geq (2d - 1)\lfloor k^2/4 \rfloor \geq \binom{k}{2} d,$$

implying that $d \geq k/2$, as needed. □

Similar nonexistence results can be derived for other values of v and k, provided $v \geq 2k$ and $v - 2k$ is small, using the Plotkin bound. (See, for example, [6, pp. 41–43].)

TABLE 7
Existence of optimal bbc's I.

v	k	Existence for discrimination d, $1 \leq d \leq 40$			
		0000000001	1111111112	2222222223	3333333334
		1234567890	1234567890	1234567890	1234567890
9	8	+IIIIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
10	8	.++IIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
10	9	+IIIIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
11	8	.Y+YIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
11	9	.+YIIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
11	10	+IIIIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
12	8	.++IIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
12	9	.++IIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
12	10	.++IIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
13	8	.Y.Y+YIYII	IIIIIIIII	IIIIIIIII	IIIIIIIII
13	9	.YY+IIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
13	10	.Y+IIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
14	8	...YY+YY+I	IIIIIIIII	IIIIIIIII	IIIIIIIII
14	9	.YYI+YIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
14	10	.+YIIIIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
15	8	....oY+YYY	YIIIIIIIII	IIIIIIIII	IIIIIIIII
15	9	...+Y+YIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
15	10	.+.I+IIIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
16	8	.....+.+.+	.I.I+I?I+I	IIIIIIIII	IIIIIIIII
16	9	...YYY+YYY	IIIIIIIII	IIIIIIIII	IIIIIIIII
16	10	.Y.YY+YI+I	IIIIIIIII	IIIIIIIII	IIIIIIIII
17	8	.....Yo+YY	YIYIYIIIII	IIIIIIIII	IIIIIIIII
17	9	.....Yo+YY	YIYIYIIIII	IIIIIIIII	IIIIIIIII
17	10	...YYY+IYI	IIIIIIIII	IIIIIIIII	IIIIIIIII
18	8	...+YYoIII	YIIIIIIIII	IIIIIIIII	IIIIIIIII
18	9	.....+.+.+	.I.I.I+I?I	?IIIIIIII	IIIIIIIII
18	10	...+YYoIII	YIIIIIIIII	IIIIIIIII	IIIIIIIII
19	8	...Y.YY+YY	IIYIIIIIII	IIIIIIIII	IIIIIIIII
19	9	.....ooY+Y	YYYYYIIIII	IIIIIIIII	IIIIIIIII
19	10	.....ooY+Y	YYYYYIIIII	IIIIIIIII	IIIIIIIII
20	8	...+Y+YIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
20	9	....oYoY+Y	YYYIIIIIII	IIIIIIIII	IIIIIIIII
20	10	.....+.+.+	.I.I.I.I+I	?I?IIIIIII	IIIIIIIII
21	8	...YYYY+YI	YIIIIIIIII	IIIIIIIII	IIIIIIIII
21	9	...Yo+YY+I	YIIIIIIIII	IIIIIIIII	IIIIIIIII
21	10	.....oooo+	YYYYYYYYYI	IIIIIIIII	IIIIIIIII
22	8	...+YYYIII	IIIIIIIII	IIIIIIIII	IIIIIIIII
22	9	...YYYYI+I	YIIIIIIIII	IIIIIIIII	IIIIIIIII
22	10	.....YoYo+	YYYY+IYIYI	IIIIIIIII	IIIIIIIII

TABLE 8
Existence of optimal bbc's II.

v	k	Existence for discrimination d, $1 \leq d \leq 40$			
		0000000001	1111111112	2222222223	3333333334
		1234567890	1234567890	1234567890	1234567890
23	8	...YYYY+YY	YIIIIIIIII	IIIIIIIIII	IIIIIIIIII
23	9	...YoYYY+Y	IYIIIIIIII	IIIIIIIIII	IIIIIIIIII
23	10	...YoYYYY+	YIYIIIIIII	IIIIIIIIII	IIIIIIIIII
24	8	...++++III	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
24	9	...YY+YI+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
24	10	...Y.YoIY+	YYII+IYIII	IIIIIIIIII	IIIIIIIIII
25	8	...YYYY+II	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
25	9	...YoYYY+Y	YIIYIIIIII	IIIIIIIIII	IIIIIIIIII
25	10	...+o+YIYI	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
26	8	...+YYYIII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
26	9	...YYYYI+Y	YYIIIIIIII	IIIIIIIIII	IIIIIIIIII
26	10	...Y.YYYY+	YIYI+IIIII	IIIIIIIIII	IIIIIIIIII
27	8	...YYYY+II	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
27	9	...++++III	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
27	10	...YYYYII+	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
28	8	...+Y+YIII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
28	9	...YYYYY+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
28	10	...Y+YYYII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
29	8	.Y.YYYY+YI	YIIIIIIIII	IIIIIIIIII	IIIIIIIIII
29	9	...YYYYI+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
29	10	...YYYYII+	YYYIIIIIII	IIIIIIIIII	IIIIIIIIII
30	8	.Y.+YIIIII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
30	9	...YY+YY+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
30	10	...++++III	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
31	8	...YYYY+YY	YIIIIIIIII	IIIIIIIIII	IIIIIIIIII
31	9	...YYYYI+I	YIIIIIIIII	IIIIIIIIII	IIIIIIIIII
31	10	...YYYYYY+	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
32	8	...++++III	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
32	9	.Y.YYYYY+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
32	10	...Y+YYIII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
33	8	.Y.YYYY+II	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
33	9	...YY+YI+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
33	10	...YYYYYY+	IIYIIIIIII	IIIIIIIIII	IIIIIIIIII
34	8	.Y.+YIYIII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
34	9	...YYYYY+I	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII
34	10	...Y+YYIII	IIIIIIIIII	IIIIIIIIII	IIIIIIIIII

Since our focus here is on cases in the range of practical interest, we do not include a detailed study of these results.

6. Existence of optimal bbc's. We summarize the existence results for equireplicate optimal bbc's in the range of primary interest for the oligo array application. We can assume that addition is applied to all of the basic designs produced. Then it is an easy matter to verify that all but a handful of parameter sets are settled. When $k \in \{8, 9, 10\}$ and $k < v \leq 34$, we have established existence or nonexistence in all but five cases, namely, when (v, k, d) is one of $(16, 8, 17)$, $(18, 9, 19)$, $(18, 9, 21)$, $(20, 10, 21)$, or $(20, 10, 23)$.

In [3], we develop a hillclimbing method which is remarkably successful at producing bbc's, even optimal ones. Indeed, when the bbc is not equireplicate, we succeeded in producing a large number of base designs. In Tables 7 and 8, we give a statement of the current result for all parameter sets with $k \in \{8, 9, 10\}$ and $k < v \leq 34$. The encoding is as follows: `+` denotes the existence of an optimal equireplicate bbc, which is described in this paper; `?` denotes an unsettled equireplicate case; `.` denotes a parameter set for which nonexistence of any optimal bbc has been established; `Y` denotes a nonequireplicate optimal bbc, found using the algorithm from [3]; and `o` denotes an unsettled nonequireplicate case. The majority of entries are obtained by addition of bbc's with smaller discrimination; a construction of this type is denoted by `I`, for "implied." Note that sometimes an optimal bbc can be implied by the addition of two nonequireplicate optimal bbc's.

We present the status only for $1 \leq d \leq 40$, but it can easily be established that existence is implied for all $d \geq 40$ for all parameter sets in our range, using addition.

The practical consequence of this is that for large discrimination, the problem appears to become easier. However, only through the direct and computational constructions for small discrimination have we been able to establish such a strong existence result.

7. Concluding remarks. Optimal balanced binary codes appear, at first glance, to require strong balance conditions leading to designs. Indeed, when $v = 2k$, the conditions are quite severe and do require the pair-balance condition of balanced incomplete block designs. However, when v is not near $2k$, the packing conditions that are required appear to be much less restrictive than do the conditions on block sizes and replication numbers. This is the primary reason that the approach here of constructing the required packings directly appears more fruitful than the approach of starting with block designs and applying simple transformations.

One might expect that the nonequireplicate cases would be easier in view of the increased flexibility in choosing replication numbers. In [3], we exploit this flexibility to develop an heuristic search technique that is very successful.

While we have focused in this paper on cases in the range of practical interest, we expect that similar conclusions and techniques arise more generally in the existence of bbc's.

Acknowledgments. Thanks to Jeff Dinitz, Vic Klee, Don Kreher, Esther Lamken, and Rimli Sengupta for helpful suggestions.

REFERENCES

[1] T. Beth, D. Jungnickel, and H. Lenz, *Design Theory*, Cambridge University Press, Cambridge, UK, 1986.

[2] C. J. Colbourn and W. de Launey, *Difference matrices*, in CRC Handbook of Combinatorial Designs, C. J. Colbourn and J. H. Dinitz, eds., CRC Press, Boca Raton, FL, 1996, pp. 279–289.

[3] C. J. Colbourn, A. C. H. Ling, and M. Tompa, *A hillclimbing method for balanced binary codes for oligo arrays*, Bioinformatics, to appear.

[4] E. Hubbell and P. A. Pevzner, *Fidelity probes for DNA arrays*, in Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, Heidelberg, Germany, 1999, pp. 113–117.

[5] R. J. Lipshutz, S. P. A. Fodor, T. R. Gingeras, and D. J. Lockhart, *High density synthetic oligonucleotide arrays*, Nature Genetics Supplement, 21 (1999), pp. 20–24.

[6] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, 1977.

[7] R. A. Mathon and A. Rosa, *Balanced incomplete block designs*, in CRC Handbook of Combinatorial Designs, C. J. Colbourn and J. H. Dinitz, eds., CRC Press, Boca Raton, FL, 1996, pp. 3–41.

[8] W. H. Mills and R. C. Mullin, *Coverings and packings*, in Contemporary Design Theory: A Collection of Surveys, J. H. Dinitz and D. R. Stinson, eds., Wiley, New York, 1992, pp. 371–399.

[9] R. Sengupta and M. Tompa, *Quality control in manufacturing oligo arrays: A combinatorial design approach*, in Pacific Symposium on Biocomputing, Mauna Lani, Hawaii, 2001, pp. 348–359; also available via ftp://ftp.cs.washington.edu/tr/2000/08/UW-CSE-00-08-03.PS.Z.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 498–509

THE OBNOXIOUS CENTER PROBLEM ON A TREE*

RAINER E. BURKARD†, HELIDON DOLLANI†, YIXUN LIN‡, AND GÜNTER ROTE§

Abstract. The obnoxious center problem in a graph G asks for a location on an edge of the graph such that the minimum weighted distance from this point to a vertex of the graph is as large as possible. We derive algorithms with linear running time for the cases when G is a path or a star, thus improving previous results of Tamir [*SIAM J. Discrete Math*, 1 (1988), pp. 377–396]. For subdivided stars we present an algorithm of running time $O(n \log n)$. For general trees, we improve an algorithm of Tamir [*SIAM J. Discrete Math*, 1 (1988), pp. 377–396] by a factor of $\log n$. Moreover, a linear algorithm for the unweighted center problem on an arbitrary tree with neutral and obnoxious vertices is described.

Key words. location problems, center problem, obnoxious facilities, linear-time algorithm

AMS subject classifications. 90B80, 90B85, 90C35, 90C27

PII. S0895480198340967

1. Introduction.

In the center problem, a set of clients on certain locations (sites) is given. The center problem asks for finding a location for a new facility from which the farthest client (site) can be reached in minimum time. It occurs if a fast service in the case of an emergency is needed. This problem has received strong interest since Hakimi (1964) published the first paper on this topic; see also Kariv and Hakimi (1979). For a survey on center problems, see Handler (1990); for a study of algorithms with respect to a good complexity, see Megiddo and Tamir (1983). In particular, the case has been considered that the clients are modeled as vertices of a tree. In this case it has been shown that the objective function is convex on each path, which implies that a local optimum solution is also a global optimum. Exploiting this, Megiddo (1983) gave a linear algorithm for the center problem in trees.

Recently, *obnoxious* location problems have found an increasing interest. In contrast to the usual problem, the center should be *as far away as possible* from the given sites. Thus we have to maximize the minimum (weighted) distance from the center to the sites. A formal definition of the problem is given in section 2. Such a problem occurs, for example, if the center is a facility which produces toxic agents which should be as far away as possible from the given locations of cities.

Another case where this model is applicable is a facility which is to be located as far away as possible from *obnoxious sites*. The center in this case might be some sensitive facility like an observatory or a radio station, which is affected by moisture from lakes, pollution from cities, traffic from airports, or the like. In such models, it is unnatural to use the metaphor of "clients" for the given locations; thus we prefer the neutral term *sites*.

*Received by the editors June 19, 1998; accepted for publication (in revised form) August 6, 2001; published electronically October 4, 2001. This research has been supported by the Spezialforschungsbereich F 003 "Optimierung und Kontrolle," Projektbereich Diskrete Optimierung.
http://www.siam.org/journals/sidma/14-4/34096.html
†Technische Universität Graz, Institut für Mathematik, Steyrergasse 30, A-8010 Graz, Austria (burkard@opt.math.tu-graz.ac.at, dollani@opt.math.tu-graz.ac.at).
‡Department of Mathematics, Zhengzhou University, Zhengzhou 450052, People's Republic of China (linyixun@mail.zzu.edu.cn)
§Freie Universität Berlin, Institut für Informatik, Takustraße 9, D-14195 Berlin, Germany (rote@inf.fu-berlin.de).

Complexity issues regarding the placement of several facilities in an obnoxious setting were considered by Tamir (1991). If we again consider the model where the sites are vertices of a tree, the objective function is no longer convex or concave. Drezner and Wesolowsky (1985) solve the obnoxious center problem on a tree with n vertices in $O(n^3)$ time. Tamir (1991, 1988) gives two algorithms of $O(n \log^2 n)$ and $O(kn \log^2 n)$ time, respectively, where k is a parameter that depends on the structure of the tree. In section 6 we will make a simple observation which reduces the time bound of the second algorithm to $O(kn \log n)$ time by the use of different data structures. Tamir (1988) shows that the obnoxious center problem on a path or a star with n vertices can be solved in $O(n \log n)$ time. In this paper we show that the center problem on a path or a star can even be solved in linear time (in sections 3 and 4, respectively). In section 5, we treat extended stars, which can be obtained from stars by subdividing edges and introducing additional vertices on them. We show that an obnoxious center in an extended star graph with b branches can be found in $O(n + b \log n)$ time.

In section 6 we design as well a linear algorithm for the obnoxious center problem on a tree where all sites have the same weight. Even in this case the objective function does not have any useful convexity properties. Obnoxious center problems for locating a center in the *plane* have also been considered. Melachrinoudis and MacGregor Smith (1995) used weighted Voronoi diagrams to find a weighted obnoxious center inside a convex m-gon with n sites in $O(mn^2)$ time.

The counterpart to the center problem is the median problem, where a location should be found such that the *sum* of weighted distances from the median to the sites is minimized. A version including obnoxious sites has recently been treated in Burkard and Krarup (1998). They showed that the median problem, where the friendly and/or obnoxious sites correspond to the vertices of a cactus, can be solved in linear time. (A cactus is a graph where any two cycles have at most one vertex in common.) Some further questions for future research will be mentioned in section 7.

2. Obnoxious center problems. Let $G = (V, E)$ be a simple graph with a set of n vertices $V = \{v_1, \ldots, v_n\}$ and a set E of m edges. Each edge $(v_i, v_j) \in E$ has a positive length c_{ij}. Thus we can interpret each edge as the image of a closed real interval $[0, c_{ij}]$ of length c_{ij}. For any point z in this interval we have a corresponding point P, and we define its distance to v_i as $|z|$ and its distance to v_j as $|c_{ij} - z|$. This enables us to define the shortest distance between any point P on an edge of G and a vertex v of G. The distance between P and v, denoted by $d(P, v)$, is thus the length of a shortest path from P to v. Moreover, we consider each vertex v_i of G as a site (client) and attach a positive weight w_i to it. The *center problem* on a graph G is to minimize

$$f(P) = \max_{v_i \in V} \; w_i d(P, v_i)$$

over all points P on the edges of G. This objective function reflects the goal to locate a facility (center) P as close to the clients v_i as possible, so that the clients can quickly get services from the center in case of an emergency.

In the case of an *obnoxious* facility one wants to maximize

$$g(P) = \min_{v_i \in V} \; w_i d(P, v_i).$$

This objective function places the new location as far away as possible from the sites (vertices) v_i of G. Note that the significance of weights is contrary to the usual case.

Important and highly sensitive sites (or highly obnoxious sites) receive small weights. Vertices which contain no sites should get weight ∞.

3. Obnoxious center problems on paths. If the input graph is a path, we may put the n vertices on the real line and identify them with real numbers such that

$$0 = x_1 < x_2 < \cdots < x_n$$

and $d(x_i, x_j) = |x_i - x_j|$. Then the objective function is

$$g(z) = \min\{\, w_i|z - x_i| : i = 1, \ldots, n \,\} = \min\{g^+(z), g^-(z)\}$$

with

$$g^+(z) = \min\{\, w_i(z - x_i) : x_i \le z \,\},$$
$$g^-(z) = \min\{\, w_i(x_i - z) : x_i \ge z \,\}.$$

In this section we give a linear-time algorithm which finds the maximum of $g(z)$ along a path. We first describe how to compute $g^+(z)$ from left to right. We can compute $g^-(z)$ in an analogous ways from right to left, and then it is easy to compute $g(z)$ and to find the maximum.

We incrementally compute the functions g_1^+, g_2^+, ..., g_{n-1}^+, which are defined as follows:

$$g_j^+ \colon [x_j, x_n] \to \mathbb{R}_{\ge 0} \text{ with } g_j^+(z) := \min\{\, w_i(z - x_i) : i = 1, \ldots, j \,\}.$$

The following properties are straightforward consequences of the definition, except for the statement about the number of breakpoints in part (e), which we shall prove later.

LEMMA 3.1.

(a) *For $x_j \le z < x_{j+1}$, we have $g^+(z) = g_j^+(z)$.*

(b) *g_j^+ is a piecewise linear concave and increasing function.*

(c) *$g_1^+(z) = w_1(z - x_1)$.*

(d) *For $j = 2, \ldots, n-1$, $g_j^+(z) = \min\{w_j(z - x_j), g_{j-1}^+(z)\}$ in the domain of g_j^+ (i.e., for $z \ge x_j$).*

(e) *The function $g^+(z)$ is piecewise linear and has at most $2n - 3$ linear pieces.*

Lemma 3.1(b) and (e) suggest a way to represent the functions g_j^+ and g^+: as a list of adjacent *intervals*, together with the coefficients a, b of a linear function $az + b$ for each interval. Actually, the list for g_j^+ can be conveniently organized as a stack because we will modify only the list at its left end. In what follows, when we speak of an interval, we will include the coefficient of a linear function defined on that interval without mentioning it.

Lemma 3.1(c)–(d) opens the way for an incremental construction of these lists; see Figure 1.

Incremental step: Construction of g_j^+ from g_{j-1}^+. First scan. We scan the list of intervals of g_{j-1}^+ from x_{j-1} until we reach x_j. This list of intervals is removed and copied to the list of intervals that define the function g^+, according to Lemma 3.1(a). In general, the point x_j lies in the middle of an interval. That interval is then split into a left part, which is contributed to g^+, and the right part, which remains as part of the definition of g_{j-1}^+.

Second scan. After this transformation, the domain of the function has been reduced to $[x_j, x_n]$, which is the domain of the function g_j^+ that we wish to compute.

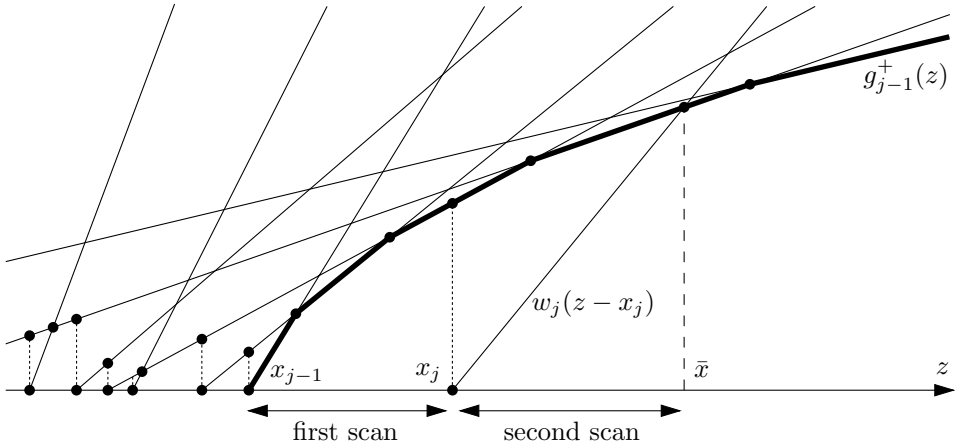


FIG. 1. *The construction of g^+_j from g^+_{j-1}.*

We now apply Lemma 3.1(d). The function g^+_j will start with an interval where $g^+_j(z) = w_j(z - x_j)$. This interval extends until this linear function intersects the graph of $g^+_{j-1}(z)$; from then on, $g^+_j(z)$ coincides with $g^+_{j-1}(z)$. More precisely, this is done as follows.

We scan the list of (remaining) intervals of g^+_{j-1} from the left and compare each interval to the function $w_j(z - x_j)$; as long as this function is smaller than g^+_{j-1} in the whole range of the interval, we remove that interval from the list. When the two functions intersect, we split the interval at the intersection point $\bar{x}$, and we replace the left half by an interval $[x_j, \bar{x}]$ where the function $w_j(z - x_j)$ is used.

This concludes the construction of g^+_j.

We can now prove the bound of $2n - 3$ on the total number of linear pieces of g^+. Initially, g^+_1 has one piece. In the first scan, each piece that is contributed to the final function g^+ is removed from g^+_{j-1}, except for one additional piece that results from splitting one interval. In the second scan, the function g^+_j gets one additional piece (and it may lose other pieces). This gives a total of $1 + (n-2)(1+1) = 2n - 3$ pieces altogether, taking into account that the last iteration terminates by copying everything from g^+_{n-1} to g^+ in the first scan.

It is easy to see that the number of $2n - 3$ pieces can actually be attained.

The computation of g^- proceeds in the same way from right to left. Finally, we vary z from x_1 to x_n and compute $g(z) = \max\{g^+(z), g^-(z)\}$, simultaneously scanning the two lists of intervals for g^+ and g^-, and we return the solution z attaining the maximum of this function.

THEOREM 3.2. *The above algorithm solves the weighted obnoxious center problem on a path in linear time.*

Proof. We have to show only that the computation of g^+ (and g^-) takes linear time: each interval that is looked at in the first scan contributes one piece to the function g^+. Therefore the total time for the first scan is $O(n)$, by Lemma 3.1(e). The time for the second scan is $O(1 + k)$, where k is the number of intervals that are removed from the list of intervals. Since the total number of removed intervals cannot be bigger than the total number of intervals that were ever added to the list, the total time for the second scan is also $O(n)$. Note that the comparison of two linear

functions and determining the intersection point in each interval can be executed in constant time.

The final scan of the algorithm is easily done in $O(n)$ time. □

We remark that the essence of the above algorithm for computing g^+ is the same as an incremental algorithm for computing the intersection of half-spaces $H_1 \cap \cdots \cap H_i$, for $i = 1, \ldots, n$, if the half-spaces are inserted *in the order of their intersection with a fixed line* (the x-axis in our case). In our case, we have the half-spaces $H_j := \{ (z, y) : y \le w_j(z - x_j) \}$, and we are actually only interested in the part lying above the x-axis. The algorithm is geometrically dual to (and algebraically identical to) an incremental algorithm for computing the convex hull for points in the plane which are *sorted by x-coordinate*; cf., for example, Preparata and Shamos (1985) for a description of this duality and for the linear-time convex hull algorithm for sorted points.

The unweighted version of this problem is essentially the MAX-GAP problem of finding the longest edge (or the *maximum gap*) between two successive numbers which can be solved in linear time even if the numbers x_i are not given in sorted order; see Gonzalez (1975).

4. Obnoxious centers in star graphs. A *star* is a complete bipartite graph $K_{1,n}$. It is a tree $T = (V, E)$ consisting of a central vertex v_0 which has edges to n other vertices $\{v_1, \ldots, v_n\}$. Denote $x_i := c_{0i}$, for $i = 1, \ldots, n$, and $x_0 = 0$. The subproblem of determining a locally optimum solution on the edge (v_0, v_i) will be denoted by S_i. It is equivalent to the problem on a path as follows: we put all vertices on the real line such that $v_0 = 0$, v_i is to the right of v_0 with distance $|v_i - v_0| = x_i$, and all other vertices v_j $(j \ne i)$ are to the left of v_0 with distance $|v_j - v_0| = x_j$. In problem S_i we have to maximize the function

$$g_i(z) = \min \left\{ \min_{j \ne i} w_j(x_j + z),\ w_i(x_i - z) \right\} \tag{4.1}$$

for $0 \le z \le x_i$ (see Figure 2). We can omit the condition $j \ne i$ from (4.1) without changing the problem because, for $z \ge 0$, $w_i(x_i + z)$ is always larger than the second term of the expression, $w_i(x_i - z)$:

$$g_i(z) = \min \left\{ \min_{j=0,\ldots,n} w_j(x_j + z),\ w_i(x_i - z) \right\}. \tag{4.2}$$

Let the maximum be attained in $z^{(i)}$. Obviously, the point $z^{(i)}$ is a solution of the following linear program in two variables y and z (see Figure 2):

$$\min\{ z : y \le w_j(x_j + z) \text{ for } j = 0, \ldots, n,\ y \ge w_i(x_i - z) \}. \tag{4.3}$$

The constraints which are common to all problems S_i can be written as $y \le h(z)$, where

$$h(z) := \min_{j=0,\ldots,n} w_j(x_j + z).$$

Obviously, $h(z)$ is a piecewise linear and increasing function. The point $z^{(i)}$ is the intersection point of $h(z)$ with $w_i(x_i - z)$. Due to the monotonicity of $h(z)$, the obnoxious center problem asks for $z^* := \max z^{(i)}$. However, this z^* can now be obtained as optimal solution of the linear program (see Figure 3)

$$\min\{ z :\ y \le w_j(x_j + z) \text{ for } j = 0, \ldots, n;\ y \ge w_j(x_j - z) \text{ for } j = 1, \ldots, n \}. \tag{4.4}$$

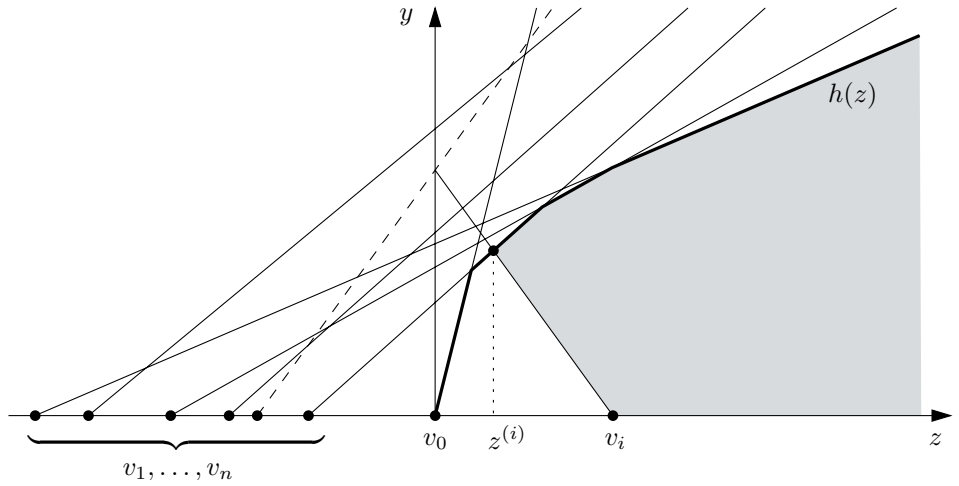


FIG. 2. *Graphical representation of subproblem S_i. The shaded region is the feasible region of* (4.3). *Adding the constraint indicated by the dotted line does not change the problem.*

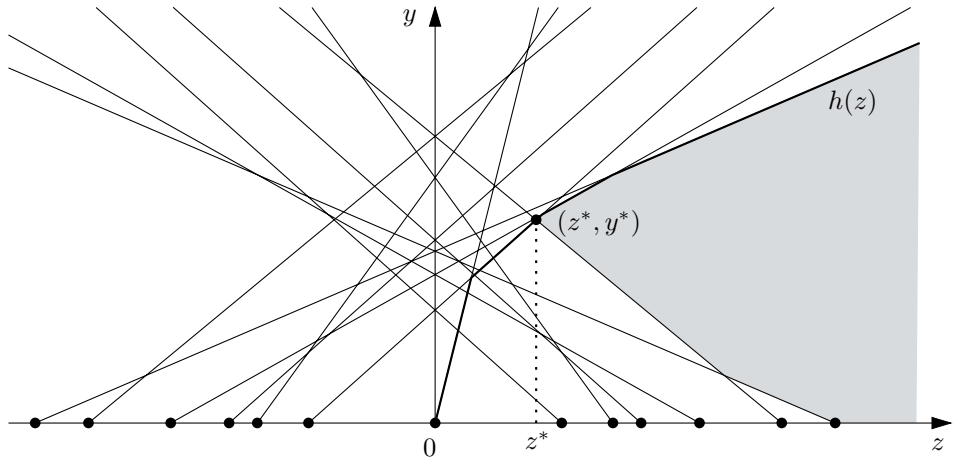


FIG. 3. *The feasible region of the linear program* (4.4).

THEOREM 4.1. *Consider an optimal solution (z^*, y^*) of the linear program* (4.4). *Then the optimal objective function value of the obnoxious center problem is y^*, and the optimal locations problem are the points at distance z^* from the central vertex v_0 on all edges (v_0, v_i) for which $y^* = w_i(x_i - z^*)$ holds.*

Proof. First note that $h(z^*) = y^*$, and $y^* = w_i(x_i - z^*)$ must hold for at least one index i, because otherwise there would be a solution of (4.4) with $z < z^*$. It is easy to check that the locations which are constructed in the theorem have the claimed objective function value.

We still have to show that there is no other solution. Consider a point P on edge (v_0, v_i) at distance z from v_0. If $z < z^*$, then $h(z) < h(z^*) = y^*$, and hence there is a site v_j whose distance $d(P, v_j)$ from P is $h(z)/w_j$, which means that $w_j d(P, v_j) < y^*$. If $z > z^*$, or if $z = z^*$ and i is not one of the indices j for which $y^* = w_j(x_j - z^*)$ holds, then $y^* > w_i(x_i - z^*)$ and $d(P, v_i) = x_i - z < y^*/w_i$, and thus

$w_i d(P, v_i) < y^*$. □

The linear program (4.4) has $2n$ constraints and two variables. By the algorithm of Megiddo (1983) it can be solved in linear time. Thus the obnoxious center in a star graph can be found in linear time.

5. Obnoxious centers in extended star graphs. An *extended* star graph is a tree which has a single vertex v_0 with degree greater than 2. The remaining vertices form paths from v_0 to the leaves of the graph. We call these paths the *branches* of the tree. This class of graphs is a mixture of paths and stars, which were considered in the previous two sections. We will show that an obnoxious center in an extended star with n vertices and b branches can be found in $O(n + b \log n)$ time. When b is relatively small, the algorithm runs in linear time. However, when, for example, all branches contain two edges and $b \approx n/2$, the time complexity is $O(n \log n)$. We do not see how to solve the problem in linear time even in this special case.

As in section 4, we denote $x_i := c_{0i}$. First we consider a local subproblem for each branch separately: we move the center P on the ith branch using $z := d(P, v_0)$ as a parameter. We construct the "local" objective function $g_i(z)$, considering only sites on the ith branch (starting at v_0) and ignoring all vertices on other branches. This function is defined on some interval $0 \le z \le Z_i$, where Z_i is the length of the branch. The vertex v_0 corresponds to $z = 0$. By the methods of section 3, all piecewise linear functions g_i can be constructed in linear time.

We also consider the function

$$h(z) := \min_{j=0,\dots,n} w_j(z + x_j).$$

As in the previous section, the optimum value is now given by

$$y^* = \max_{i=1,\dots,b} \max_{0 \le z \le Z_i} \min\{h(z), g_i(z)\}. \tag{5.1}$$

It is clear that this optimum is located either at a local maximum of some function g_i or at an intersection point of the graph of g_i with the graph of h.

The overall approach for solving this problem can roughly be described as follows. The function h is a piecewise linear concave and increasing function. We perform a binary search among its breakpoints $(\hat{z}, \hat{y})$ to find the range of y values in which the optimum value lies. To do this, we need to test whether $y^* \ge \hat{y}$ for a given point $\hat{y} = h(\hat{z})$ on the function h.

This test is carried out as follows. We successively look at each function g_i, decreasing z from the maximum permitted value Z_i down to $\hat{z}$. We stop this scan as soon as some value $g_i(z) \ge \hat{y}$ with $z \ge \hat{z}$ is found. We have found a feasible solution with value $\hat{y}$, and hence $y^* \ge \hat{y}$. On the other hand, if we have scanned all domains $z \ge \hat{z}$ for all branches i without finding a value $g_i(z) \ge \hat{y}$, we know that $y^* < \hat{y}$.

As we scan the functions g_i, we remember the highest value $\tilde{y}$ that we have encountered. If we later get another query with a different point $(\hat{z}, \hat{y})$, we may be able to answer immediately because $\tilde{y} \ge \hat{y}$. Otherwise, we continue the right-to-left scan of each function g_i at the value z where we left off during the last previous scan of this function.

Note that we scan only intervals where we know that $g_i(z) \le h(z)$. This means that linear pieces of g_i which were examined need not be examined again because the feasible value $\tilde{y}$ is an upper bound of $g_i(z)$ over all intervals that were examined. Therefore the time complexity for answering a sequence of tests of the condition $y^* \ge \hat{y}$

is bounded by the total number of pieces of all functions g_i, which is $O(n)$, plus an overhead of $O(b)$ for each test. The overhead comes from the fact that we may have to spend constant time for each branch i just to "look at" this branch. For example, if we stopped the previous scan of g_i because the point $\hat{z}$ was reached, we may have to repeatedly examine the single linear piece to which this point belongs.

We will now describe more precisely how the binary search among the breakpoints of h is carried out. The function h is the lower envelope of $n+1$ increasing linear functions, whose slopes are given by the weights w_i. We start by finding the median $\hat{w}$ of the $n+1$ slopes and identifying the leftmost point $\hat{y} = h(\hat{z})$ on the graph where the slope becomes $\leq \hat{w}$. This point can be identified by solving the linear program

$$\max\{\, y - \hat{w}z : z \geq 0,\ y \leq w_j(z + x_j) \text{ for } j = 0, \ldots, n \,\} \tag{5.2}$$

in the two variables y and z. Actually, this linear program may yield *any* point with slope $\hat{w}$, not necessarily the *leftmost* point with slope $\hat{w}$ or smaller. The leftmost point can be found by perturbing the objective function or by solving the following auxiliary linear program:

$$\min\{\, z : y - \hat{w}z = K^*,\ z \geq 0,\ y \leq w_j(z + x_j) \text{ for } j = 0, \ldots, n \,\},$$

where K^* is the optimum value of (5.2). (This is a linear program in only one variable, after using the equation to eliminate y.)

Now we test the condition $y^* \geq \hat{y}$ as described above. If we find that $y^* \geq \hat{y}$, we can discard the first half of the linear functions, with slopes $> \hat{w}$, from consideration in $h(z)$ because we know that the optimum cannot lie in the range $z < \hat{z}$. Otherwise, if $y^* < \hat{y}$, we can discard the other half of the linear functions from the definition of $h(z)$ because they play no role for restricting the optimum of (5.1). (In fact, in this case, we have actually scanned the part with $z \geq \hat{z}$ of all branches, and we can restrict the remaining search to the range $z < \hat{z}$.)

We continue this process by finding the median of the remaining pieces of h, and so on, until only one linear piece of h is left. It is then easy to find the optimum value of (5.1) directly in linear time.

Since the median of n numbers can be found in linear time by the algorithm of Blum et al. (1973) (see also Aho, Hopcroft, and Ullman (1983)), the overall effort for the binary search is

$$O(n) + O(n/2) + O(n/4) + \cdots = O(n).$$

To this we must add the effort for the $O(\log n)$ queries, which is

$$O(\log n) \cdot O(b) + O(n),$$

as discussed above. Summarizing, we have the following theorem.

THEOREM 5.1. *The weighted obnoxious center problem on an extended star tree with n vertices and b branches can be found in $O(n + b \log n)$ time and $O(n)$ space.*

6. The obnoxious center problem in general trees. In this section we consider the obnoxious center problem in weighted and unweighted trees.

6.1. Finding an obnoxious center in weighted trees. Tamir (1991, 1988) gives two algorithms of $O(n \log^2 n)$ and $O(kn \log^2 n)$ time complexity, respectively, for solving the obnoxious center problem on an arbitrary tree with n vertices. The

parameter k depends on the structure of the tree. For paths and stars $k = O(1)$, and for balanced trees $k = O(\log n)$, but there exist trees such that $k = \Theta(n)$. By an easy observation, Tamir's algorithm of 1988 can be improved by a factor of $\log n$. Tamir notes that, if the center is restricted to a single edge, the objective function is a lower envelope of n linear functions. When one goes from an edge to an adjacent edge, not all of these linear functions have to be changed. We can obtain the lower envelope of the functions for the adjacent edge by removing some linear functions and adding new ones. Tamir (1988) showed that one can successively obtain the objective function for all edges with a total of $O(kn)$ insertions and deletions of linear functions.

Tamir (1988) used the data structure of Overmars and van Leeuwen (1981) for maintaining a lower envelope of n linear functions under deletions and insertions. This data structure takes linear space and $O(\log^2 n)$ time for a deletion or insertion. The maximum of the current lower envelope over some given interval can be found in $O(\log n)$ time.

However, the sequence of $O(kn)$ insertions and deletions of linear functions can be computed beforehand, and thus we can use the algorithm of Hershberger and Suri (1996) for an *off-line* maintenance of the lower envelope of linear functions. This data structure needs only $O(n \log n)$ time to process a sequence of n insertions, deletions, and queries for the maximum, i.e., only $O(\log n)$ time per operation on the average. In total, this reduces the complexity to $O(kn \log n)$. Note that for small k ($k = O(\log n)$), this time complexity bound is lower than the $O(n \log^2 n)$ bound of Tamir (1991).

6.2. A linear algorithm for finding an obnoxious center in unweighted trees. Let us determine an obnoxious center in the tree $T = (V, E)$ with edge lengths c_{xy}. The center can be placed in any vertex or on any edge of the tree but should be as far away as possible from the vertices of some given set $V_0 \subseteq V$ of obnoxious sites. Let us call the vertices in V_0 *black vertices* and the vertices outside of V_0 *white vertices*. Thus our problem is to maximize

$$g(z) = \min_{v \in V_0} d(z, v).$$

The objective function $g(z)$ is not necessarily concave along a path. For example, in the tree shown in Figure 4 (V_0 consists of the 4 black vertices), $g(z)$ is neither convex nor concave along the path (a, b, c, d).

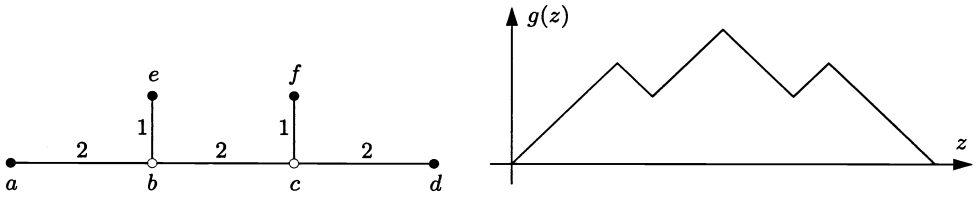


FIG. 4. *Nonconvexity of objective function $g(z)$.*

Since $g(z)$ is concave on each edge, an $O(n^2)$ algorithm is easy to realize by examining every edge. In the following we describe a linear algorithm.

We first select an arbitrary vertex r as the root of the tree. Then we perform a sweep from the leaves to the root, and for each vertex u, we compute the minimum distance $g^+(u)$ from u to a black vertex in the subtree below u (including u itself).

Finally, during a root-to-leaf sweep, for each vertex u, we compute the minimum distance $g(u)$ from u to any black vertex, and we also locate the optimal point on each edge.

Phase I. We denote the set of children of the vertex u by $S(u)$. We can set $g^+(u) := 0$ for all black vertices. For white vertices we have

$$g^+(u) = \min_{v \in S(u)} (c_{uv} + g^+(v)).$$

This includes the case of white leaves ($S(u) = \emptyset$) for which we can initialize $g^+(u) = \infty$. We then inductively compute $g^+(u)$ for all other white vertices, proceeding from the leaves towards the root.

Phase II. We proceed from the root to the leaves. At the root r, we have $g(r) := g^+(r)$.

Now consider an edge (u, v) from a vertex u to its child v. We assume that $g(u)$ has already been determined correctly. For a point P at distance z from u on this edge, we claim that the minimum distance to a black vertex is

$$\min\{g(u) + z,\ g^+(v) + (c_{uv} - z)\} \text{ for } 0 \le z \le c_{uv}. \tag{6.1}$$

The second expression is clearly equal to the minimum distance from P to the nearest black vertex in the subtree of v. On the other hand, if the path from P to the nearest black vertex goes through u, its length is represented by the first expression in the above formula. It follows that the expression (6.1) is certainly not bigger than the minimum distance from P to a black vertex.

It is possible that the first expression $g(u) + z$ does not correspond to a path from P to a black vertex. It may represent a walk that starts by going from P to u, returns to v, and continues into the subtree of v. This happens precisely when the closest black vertex of u lies in this subtree. However, then we must have $g(u) = c_{uv} + g^+(v)$, and the second expression is smaller than the first. We conclude that there is a black vertex whose distance from P equals (6.1), and hence the claim is true.

We can now determine the optimal location for the center on the edge (u, v) in constant time by maximizing (6.1) over all z. For $z = c_{uv}$, we get $P = v$, and hence

$$g(v) := \min\{g(u) + c_{uv}, g^+(v)\}.$$

This formula allows us to determine $g(v)$ for every vertex v from the value $g(u)$ of its parent u, and we can inductively find $g(v)$ for all vertices.

It is obvious that the above procedure takes linear time. Thus we have shown the following theorem.

THEOREM 6.1. *The unweighted obnoxious center problem on a tree can be solved in linear time.*

7. Concluding remarks. In the study of obnoxious center problems, Tamir (1988, 1991) presented $O(n \log n)$ algorithms for path trees and star trees, and an $O(n \log^2 n)$ algorithm for general trees. For the extremal cases, i.e., for paths (the trees with largest diameter) and for stars (the trees with smallest diameter), as well as for unweighted trees, we have obtained $O(n)$ algorithms in this note. The question whether one can get linear-time algorithms for general trees remains open.

For the multifacility obnoxious center problem on a path, an approach based on the $O(n \log n)$ algorithm of Tamir (1988) significantly improved the $O(n^3)$ bound of

Drezner and Wesolowsky (1985). Now, by using our linear algorithm of section 3, the bound can be further improved to $O(n)$.

A natural generalization of the center problem and the obnoxious center problem is to combine the two objective functions $f(x)$ and $g(x)$ for locating a center x. A similar approach was proposed for the generalized median problem by Burkard and Krarup (1998). On one hand, we may view

$$f(z) = \max_{v_i \in V_+} w_i d(z, v_i)$$

as the service cost for friendly sites in $V_+ \subseteq V$ in case of emergency, where $w_i > 0$ for $v_i \in V_+$. On the other hand, we may view

$$g(z) = M + \max_{v_i \in V_-} w_i d(z, v_i)$$

as the damage cost of the obnoxious sites in $V_- \subseteq V$ in case of an emergency, where $w_i < 0$ for $v_i \in V_-$ and $M > 0$ is a constant. Let p, q be the probabilities of these two kinds of emergency events. Then the expected cost will be

$$E(z) = p \cdot f(z) + q \cdot g(z).$$

The model of minimizing $E(z)$ would be an analogue of the median problem with positive and negative weights (see Burkard and Krarup (1998)), and could be an interesting problem for further study. First results in this respect concerning paths, stars, and trees can be found in the recent report by Burkard and Dollani (2001).

Acknowledgment. We thank Gerhard Woeginger for useful discussions.

REFERENCES

A. V. Aho, J. E. Hopcroft, and J. D. Ullman (1983), *Data Structures and Algorithms*, Addison-Wesley, Reading, MA.

M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan (1973), *Time bounds for selection*, J. Comput. System Sci., 7, pp. 448–461.

R. E. Burkard and H. Dollani (2001), *Center Problems with Pos/Neg Weights on Trees*, SFB-Report 215, Institute of Mathematics, Graz University of Technology, Graz, Austria.

R. E. Burkard and J. Krarup (1998), *A linear algorithm for the pos/neg-weighted* 1*-median problem on a cactus*, Computing, 60, pp. 193–215.

Z. Drezner and G. O. Wesolowsky (1985), *Location of multiple obnoxious facilities*, Transportation Sci., 19, pp. 193–202.

T. Gonzalez (1975), *Algorithms on Sets and Related Problems*, Technical report, Department of Computer Science, University of Oklahoma, Norman, OK.

S. L. Hakimi (1964), *Optimum locations of switching centers and the absolute centers and medians of a graph*, Operations Res., 12, pp. 450–459.

G. Y. Handler (1990), *p-center problems*, in Discrete Location Theory, P. B. Mirchandani and R. L. Francis, eds., Wiley, New York, pp. 305–347.

J. Hershberger and S. Suri (1996), *Off-line maintenance of planar configurations*, J. Algorithms, 21, pp. 453–475.

O. Kariv and S. L. Hakimi (1979), *An algorithmic approach to network location problems* I: *The p-centers*, SIAM J. Appl. Math., 37, pp. 513–538.

O. Kariv and S. L. Hakimi (1979), *An algorithmic approach to network location problems* II: *The p-medians*, SIAM J. Appl. Math., 37, pp. 539–560.

N. Megiddo (1983), *Linear-time algorithms for linear programming in* R^3 *and related problems*, SIAM J. Comput., 12, pp. 759–776.

N. Megiddo and A. Tamir (1983), *New results on the complexity of p-center problems*, SIAM J. Comput., 12, pp. 751–758.

E. Melachrinoudis and J. MacGregor Smith (1995), *An* $O(mn^2)$ *algorithm for the maximin problem in* E^2, Oper. Res. Lett., 18, pp. 25–30.

M. H. Overmars and J. van Leeuwen (1981), *Maintenance of configurations in the plane*, J. Comput. System Sci., 23, pp. 166–204.
F. Preparata and M. I. Shamos (1985), *Computational Geometry: An Introduction*, Springer-Verlag, New York.
A. Tamir (1988), *Improved complexity bounds for center location problems on networks by using dynamic data structures*, SIAM J. Discrete Math., 1, pp. 377–396.
A. Tamir (1991), *Obnoxious facility location on graphs*, SIAM J. Discrete Math., 4, pp. 550–567.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 510–523

STRUCTURAL DIAGNOSIS OF WIRING NETWORKS: FINDING CONNECTED COMPONENTS OF UNKNOWN SUBGRAPHS*

WEIPING SHI† AND DOUGLAS B. WEST‡

Abstract. Given a graph $G = (V, \mathcal{E})$, we want to find the vertex sets of the components of an unknown subgraph $F = (V, E)$ of G such that $E \subseteq \mathcal{E}$. We learn about F by sending an oracle a query set $S \subseteq V$, and the oracle tells us the vertices connected to S in F. The objective is to use the minimum number of queries to partition the vertex set V into components of F. In electronic circuit design, the problem is also known as structural diagnosis of wiring networks.

Key words. graph theory, graph algorithm, lower bound, fault diagnosis, component, connection class

AMS subject classifications. 68Q25, 68R10, 05C85, 05C40, 94C12

PII. S0895480100371286

1. Introduction.

1.1. Problem formulation.

Diagnosis of wiring networks is an important problem in the design and production of very large scale integration, multichip module, and printed circuit board systems [1, 4, 5, 7, 8, 12]. A wiring network consists of a set of nets. Each net contains a driver, a set of receivers, and electric conductors that connect the driver and the receivers. The logic value (1 or 0) of a good net is set by its driver and observed by its receivers. When two or more nets are involved in a short fault, their receivers all receive the logical OR of the values of their drivers. To diagnose a wiring network, we send test vectors of 0's and 1's from the drivers, and observe the outputs from the receivers, to find all the short faults.

In this paper, we study *structural diagnosis*, that is, the detection and location of all short faults between nets using the information regarding the particular routing of the nets. In contrast, *behavioral diagnosis* does not use any structural information and assumes all faults are possible. From the circuit layout information, we can find all places where a direct short fault may occur and represent the information as an undirected graph $G = (V, \mathcal{E})$, which we call the *adjacency graph*. Each vertex $v \in V$ represents a net and each edge $v_iv_j \in \mathcal{E}$ represents a potential direct short fault between nets v_i and v_j. Note that although G records only potential direct short faults between pairs of nets, we may have multiple-net short faults through sequences of direct shorts involving two nets each.

In any graph G, vertices v_i and v_j are *connected* if G contains a path from v_i to v_j. The *components* of a graph G are its maximal connected subgraphs. The vertex sets of the components are the equivalence classes of the connection relation, which we call the *connection classes* of G.

The actual presence of direct short faults among the nets can be viewed as a *fault graph* $F = (V, E)$, which is a subgraph of G. The vertex set of F is the same as the

*Received by the editors April 24, 2000; accepted for publication (in revised form) May 9, 2001; published electronically October 23, 2001.

http://www.siam.org/journals/sidma/14-4/37128.html

†Department of Electrical Engineering, Texas A&M University, College Station, TX 77843 (wshi@ee.tamu.edu). The research of this author was supported in part by NSF grant MIP-9309120.

‡Department of Mathematics, University of Illinois at Urbana-Champaign, Urbana, IL 61801 (west@math.uiuc.edu). The research of this author was supported in part by NSA/MSP grant MDA904-93-H-3040.

vertex set of G. The edge set of F is a subset of the edge set of G, but it is not given to us. Each edge v_iv_j of F represents the presence of an actual direct short fault between nets v_i and v_j. Figure 1 shows an adjacency graph G, a fault graph F, and the connection classes of F. In general, it is not possible to uniquely determine F. For example, in Figure 1, we cannot distinguish whether F has two edges or three edges among $\{v_1, v_2, v_3\}$.

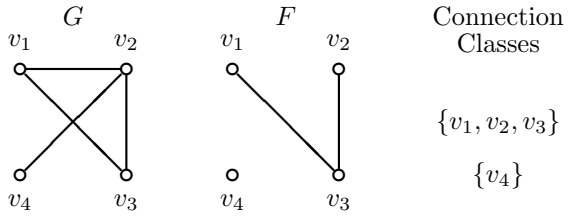


FIG. 1. *Adjacency graph G, fault graph F, and connection classes of F.*

Formally, the problem of structural diagnosis can be described as follows. Given an adjacency graph G, we want to find the connection classes of the unknown fault graph $F \subseteq G$. We obtain information about F only through queries to an oracle. For any query set $S \subseteq V$, the oracle tells us $Q(S)$, the set of vertices connected to vertices of S in the fault graph F. In other words, $Q(S)$ is the union of the connection classes that intersect S. In the example of Figure 1, $Q(\{v_1\}) = \{v_1, v_2, v_3\}$ and $Q(\{v_2, v_4\}) = \{v_1, v_2, v_3, v_4\}$. The objective is to find the connection classes of F using the minimum number of queries. Diagnosing a wiring network corresponds to finding the connection classes of the fault graph F, and applying tests corresponds to querying the oracle.

1.2. Previous results. Diagnosis algorithms may be adaptive or nonadaptive [1]. In an *adaptive* algorithm, each query is computed using the responses to previous queries. In a *nonadaptive* algorithm, all query inputs are decided before asking any queries. Adaptive algorithms can be used in applications where test vectors and results are sent and received though a external device. Nonadaptive diagnosis can be used in applications where tests are hardwired inside of the system.

There are three levels of diagnostic resolution. The highest level is *full diagnosis*, finding all the connection classes. The second level is *faulty net identification*, identifying all the nets involved in short faults. The lowest level is *fault detection*, detecting whether there is any fault at all. Faulty net identification and fault detection have been well understood [4, 8, 12]. In this paper, we study full diagnosis.

Tables 1 and 2 summarize the previous best results. In the tables, lg denotes the base 2 logarithm, n is the number of nets, G is the adjacency graph, and $\chi(G)$ is the chromatic number of G.

For behavioral diagnosis of n nets, which is the special case of structural diagnosis where $G = K_n$, Kautz [8] showed that $\lceil \lg n \rceil$ tests are necessary and sufficient to detect whether some short exists. He used the *counting sequence* method, which is optimal for fault detection but does not provide faulty net identification. To identify all the nets involved in short faults, Cheng, Lewandowski, and Wu [4] proposed the *maximum antichain* method that uses $\lg n + \frac{1}{2} \lg \lg n + O(1)$ tests. In this method, the bit strings sent by the nets all contain an equal number of 0's and 1's. The maximum antichain algorithm is proved to be optimal, or within 1 from optimal, under various restrictions on how the results are analyzed [4, 12]. For nonadaptive full diagnosis, a number of

TABLE 1
Best algorithms for behavioral diagnosis ($G = K_n$).

Diagnostic resolution	Adaptive	Best algorithm	Number of tests	Optimal
fault detection	no	counting seq [8]	$\lceil \lg n \rceil$	yes
faulty net identification	no	max antichain [4]	$(1 + o(1)) \lg n$	almost
full diagnosis	no	walking 1 skip 1 [12]	$n - 1$	yes
	yes	divide conquer [12]	$\lceil \lg n \rceil$	yes

TABLE 2
Best algorithms for structural diagnosis (arbitrary G).

Diagnostic resolution	Adaptive	Best algorithm	Number of tests	Optimal
fault detection	no	graph coloring [4, 7]	$\lceil \lg \chi(G) \rceil$	yes
faulty net identification	no	max antichain [4]	$(1 + o(1)) \cdot \lg \chi(G)$	almost
full diagnosis	no	decomposition [5]	?	?
	yes	?	?	?

researchers proposed the *walking ones* method. In this method, each query contains a single vertex. After n queries, we can look at each output $Q(\{v_i\})$ to tell which vertices are shorted to v_i. Shi and Fuchs [12] proved that if we skip any one test from the walking ones algorithm, then it is optimal. For adaptive full diagnosis, Shi and Fuchs [12] presented a divide-and-conquer algorithm that uses $\lceil \lg n \rceil$ tests, which is optimal. Shi and West [13] also gave an optimal randomized algorithm that uses expected $\lg \lg n + \lg k$ queries, where k is the number of connection classes.

For structural diagnosis, where the adjacency graph G is an arbitrary graph, several researchers [4, 7] observed that $\lceil \lg \chi(G) \rceil$ tests are sufficient for fault detection, where $\chi(G)$ is the chromatic number of G. It can be shown that $\lceil \lg \chi(G) \rceil$ tests are also necessary: Consider an adversary that responds $Q(S) = S$ while potential edges remain; this result is equivalent to showing that the minimum number of bipartite subgraphs needed to cover G is $\lceil \lg \chi(G) \rceil$. Although these observations imply that finding the minimum of tests for fault detection is NP-hard, it relates the fault detection problem to the well-known graph theory problem, where approximation algorithms are available. The maximum antichain algorithm is also applied to structural diagnosis [4], using $\lg \chi(G) + \frac{1}{2} \lg \lg \chi(G) + O(1)$ tests for faulty net identification. For full diagnosis, Feng, Huang, and Lombardi [5] proposed a graph decomposition algorithm.

1.3. Our results. We present a variety of results for full diagnosis using graph theoretic techniques. In section 2, we propose adaptive algorithms for structural diagnosis. Theorem 2.7 leads to a method of approximating the minimum number of adaptive tests for arbitrary adjacency graphs. In section 3, we consider the nonadaptive problem. We prove that deciding whether a set of queries can perform full diagnosis is NP-hard, even if $G = K_n$. Theorem 3.7 gives a method of approximating the minimum number of nonadaptive queries for arbitrary adjacency graphs. In sections 2 and 3, we also present optimal or near optimal adaptive and nonadaptive algorithms when the adjacency graphs are complete graphs, complete bipartite graphs,

paths, trees, planar graphs, or random graphs.

2. Adaptive algorithms. In this section, we study adaptive algorithms for finding the connection classes of an unknown subgraph F of an arbitrary graph G. An adaptive algorithm A implicitly defines a decision tree. At each node of the decision tree, the algorithm selects a set of vertices S and makes a query. Upon receiving the set $Q(S)$ from the oracle, the algorithm selects a subtree. Each leaf of the decision tree corresponds to a partition of the vertex set $V(G)$ into connection classes. The decision tree cannot be represented explicitly because the number of leaves is at least the number of ways to partition $V(G)$, which is exponential.

DEFINITION 2.1. *Let $\mathbf{A}_G$ be the set of all adaptive algorithms that find the connection classes of an unknown subgraph of G. Let $q(A, F)$ be the number of queries used by algorithm A when the unknown subgraph is F. We define $q(G)$, the adaptive test number of a graph G, to be*

$$q(G) = \min_{A \in \mathbf{A}_G} \max_{F \subseteq G} q(A, F).$$

In other words, $q(G)$ is the minimum number of tests necessary and sufficient for adaptive full diagnosis when the adjacency graph is G.

Graph G_1 is a *minor* of graph G_2 if G_1 can be obtained from G_2 by a sequence of edge deletions and edge contractions. For example, Figure 2 shows that the complete graph K_4 is a minor of the complete bipartite graph $K_{3,3}$.

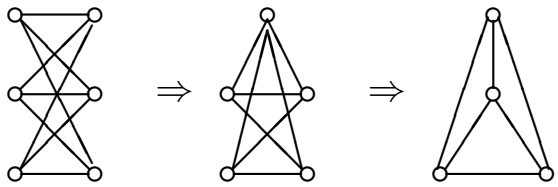


FIG. 2. *K_4 is a minor of $K_{3,3}$.*

LEMMA 2.2. *If G_1 is a minor of G_2, then $q(G_1) \leq q(G_2)$.*

Proof. If G_1 is obtained from G_2 by deleting (or contracting) an edge e, then the problem of finding connection classes for an unknown subgraph F_1 of G_1 is the problem of finding the connection classes of an unknown subgraph F_2 of G_2 with the additional information that F_2 does not contain (or does contain) edge e. With more information about F_2 within G_2, we do not need more queries.

To see how to construct an algorithm for G_1 given an algorithm for G_2, assume G_1 is obtained from G_2 by contracting an edge v_iv_j to v_i', where $v_i, v_j \in V_2$ and $v_i' \in V_1$. Any algorithm A_2 that finds the connection classes for G_2 can be modified to become an algorithm A_1 for G_1 as follows. Whenever A_2 makes a query S, A_1 makes a query S', where

$$S' = \begin{cases} S & \text{if } v_i \notin S \text{ and } v_j \notin S, \\ S \cup \{v_i'\} - \{v_i, v_j\} & \text{otherwise.} \end{cases}$$

□

LEMMA 2.3. *If the vertex sets of graphs G_1 and G_2 are disjoint, and $G_1 + G_2$ denotes the disjoint union of G_1 and G_2, then $q(G_1 + G_2) = \max\{q(G_1), q(G_2)\}$.*

Proof. Let V_1, V_2 be the vertex sets for G_1, G_2, and let A_1, A_2 be optimal algorithms for finding connection classes on these graphs. We define an algorithm on

$G_1 + G_2$. Whenever A_1 wants to query $S_1 \subseteq V_1$ and A_2 wants to query $S_2 \subseteq V_2$, we query $S_1 \cup S_2$. Since $Q(S_1) = Q(S_1 \cup S_2) \cap V_1$ and $Q(S_2) = Q(S_1 \cup S_2) \cap V_2$, both A_1 and A_2 can proceed. □

DEFINITION 2.4. *For any graph $G = (V, \mathcal{E})$ and $S \subseteq V$, the S-connection graph G_S of G is the graph with vertex set S and the edge set consisting of all pairs v_iv_j such that $v_i, v_j \in S$ and G has a path from v_i to v_j that intersects S only at its endpoints.*

For example, Figure 3 shows a graph G and its S-connection graph G_S. The concept of an S-connection graph allows us to concentrate on the set of vertices S and simplify the rest of the graph to keep only the connection information.

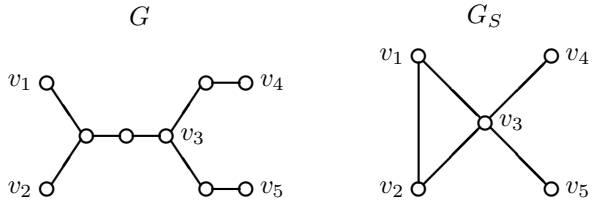


FIG. 3. *G_S is an S-connection graph of G for $S = \{v_1, v_2, v_3, v_4, v_5\}$.*

DEFINITION 2.5. *For any adjacency graph $G = (V, \mathcal{E})$ and $S \subseteq V$, the restricted connection class problem (G, S) is the problem of finding the connection classes of an unknown subgraph F of G, where F is restricted to subgraphs of G such that $Q(S) = V$. The number of queries required to solve this problem is*

$$q'(G, S) = \min_A \max_F q(A, F),$$

where F satisfies the restriction described above, A has the knowledge that F is restricted, and $q(A, F)$ is the number of queries used by A when the fault graph is F.

For example, consider the adjacency graph G in Figure 1 and the restricted connection class problem $(G, \{v_2, v_4\})$. Since the restriction requires that $Q(\{v_2, v_4\}) = \{v_1, v_2, v_3, v_4\}$, we know the fault graph F must contain at least two edges among $\{v_1, v_2, v_3\}$. Therefore, the restricted problem can be solved using only one query, $Q(\{v_2\})$.

Arguing as in the proof of Lemma 2.2, it follows that if G_1 is a minor of G_2 and both contain the vertex set S, then $q'(G_1, S) \le q'(G_2, S)$.

LEMMA 2.6. *For any graph $G = (V, \mathcal{E})$ and a set of vertices $S \subseteq V$, $q'(G, S) \le q(G_S)$.*

Proof. We first show that, for every fault graph F of G, there exists a fault graph H of G_S such that the connection classes of H are the intersections of the connection classes of F with S. H consists of edges v_iv_j such that F has a path from v_i to v_j intersecting S only at its endpoints. The graph H incorporates the information of which pairs of vertices in F are in the same connection classes.

Let A be an optimal algorithm that solves the connection class problem for G_S. We use A to solve the restricted connection class problem (G, S). At each step, if A wants to make a query R on G_S, we make a query R on G and use $Q(R) \cap S$ as a simulated response on G_S. This is the correct response for an actual subgraph H of G_S whose connection classes are the intersections with S of the connection classes of the unknown subgraph F of G. Algorithm A uses the response $Q(R) \cap S$ to choose the next query to make on G_S. When A completes its work, it declares the connection classes for H. We claim that this partition of S permits us to determine the connection

classes of F without further queries. If this claim is true, then we have used at most $q(G_S)$ queries to solve the restricted connection class problem.

Let C_i, C_j be any two connection classes of F. Since $Q(S) = V$, $C_i \cap S \neq \emptyset$ and $C_j \cap S \neq \emptyset$. Also, $C_i \cap S$ and $C_j \cap S$ are connection classes in H. Thus we know the correct distribution of S among connection classes of F. Since A determines that C_i and C_j are distinct classes, some response contains one of them but not the other. Therefore, we learn that each vertex $v \in C_i$ is not connected to any vertex in C_j for any $j \neq i$. □

For any graph $G = (V, \mathcal{E})$ and set $S \subseteq V$, the *induced subgraph* of G by S is the graph $G[S]$ whose vertex set is S and the edge set is $\{v_iv_j\colon\ v_iv_j \in \mathcal{E} \text{ and } v_i, v_j \in S\}$. The graph obtained by deleting the vertices in S is $G - S$; thus $G[V - S] = G - S$.

THEOREM 2.7. *For any graph G with vertex set V,*

$$q(G) \leq 1 + \min_{S \subseteq V} \max\{q(G_S), q(G - S)\}.$$

Proof. For any algorithm that solves the connection class problem on G, let S be the set of vertices chosen by the algorithm to make the first query. The response $Q(S)$ partitions G into disjoint subgraphs $G[Q(S)]$ and $G - Q(S)$ such that the fault graph F has no edge between $Q(S)$ and $G - Q(S)$. Therefore,

$$\begin{aligned} q(G) &\leq 1 + \min_{S \subseteq V} \max_{Q(S)} \max\{q'(G[Q(S)], S), q(G - Q(S))\} \\ &= 1 + \min_{S \subseteq V} \max \left\{ \max_{Q(S)} q'(G[Q(S)], S), \max_{Q(S)} q(G - Q(S)) \right\} \\ &\leq 1 + \min_{S \subseteq V} \max\{q'(G, S), q(G - S)\}. \end{aligned}$$

In the last step, we used the fact that $G[Q(S)]$ is a minor of G and that $G - Q(S)$ is a minor of $G - S$. From Lemma 2.6, the theorem is proved. □

COROLLARY 2.8. *Let G be a graph with vertex set V. For any $S \subseteq V$, if $G_1, \ldots, G_k$ are components of $G - S$, then $q(G) \leq 1 + \max\{\lceil \lg |S| \rceil, q(G_1), \ldots, q(G_k)\}$.*

Proof. Since every S-connection graph is a minor of $K_{|S|}$, $q(G_S) \leq q(K_{|S|}) = \lceil \lg |S| \rceil$. From Lemma 2.3, $q(G - S) = \max\{q(G_1), q(G_2), \ldots, q(G_k)\}$. □

THEOREM 2.9. *For any graph G with vertex set V, if, for every $S \subseteq V$, G_S is obtained by deleting or contracting each edge not in $G[S]$, then*

$$q(G) = 1 + \min_{S \subseteq V} \max\{q(G_S), q(G - S)\}.$$

Proof. Every optimal algorithm begins by making a query on some set S, after which it must solve the restricted problem $(G[Q(S)], S)$ and the usual connection class problem on $G - Q(S)$. It chooses S to minimize the worst-case subsequent number of queries. This yields the first equality

$$\begin{aligned} q(G) &= 1 + \min_{S \subseteq V} \max_{Q(S)} \{\max\{q'(G[Q(S)], S), q(G - Q(S))\}\} \\ &= 1 + \min_{S \subseteq V} \max \left\{ \max_{Q(S)} q'(G[Q(S)], S), \max_{Q(S)} q(G - Q(S)) \right\} \\ &\geq 1 + \min_{S \subseteq V} \max\{q'(G, S), q(G - S)\}. \end{aligned}$$

In the last step, we choose $Q(S) = V$ for the first term, and we choose $Q(S) = S$ for the second term. Then, since G_S is a minor of G, we have $q'(G, S) \geq q'(G_S, S)$.

Finally, we observe $q'(G_S, S) = q(G_S)$ since the restricted problem (G_S, S) is actually the unrestricted problem on G_S. Therefore,

$$q(G) \geq 1 + \min_{S \subseteq V} \max\{q(G_S), q(G - S)\}.$$

The other direction of the inequality is Theorem 2.7. □

Theorem 2.7 and Corollary 2.8 can be used to design an approximation algorithm for general graphs, such as the one at the end of this section. Theorem 2.9 can be used to obtain exact expressions for the adaptive test number on some classes of graphs.

COROLLARY 2.10. *For the complete graph K_n on n vertices, $q(K_n) = \lceil \lg n \rceil$.*

Proof. This was first proved by Shi and Fuchs [12]. Here we obtain it from Theorem 2.9. Each S-connection graph of K_n is $K_{|S|}$, which is a minor of K_n. Also, $K_n - S = K_{n-|S|}$. Therefore, $q(K_n) = 1 + q(K_{\lceil n/2 \rceil}) = \lceil \lg n \rceil$. □

COROLLARY 2.11. *For the n-vertex path P_n, $q(P_n) = \lceil \lg \lg(n+1) \rceil$.*

Proof. Each S-connection graph is $P_{|S|}$, which is a minor of P_n. The graph $G - S$ has $n - |S|$ vertices. From the pigeonhole principle, at least one component of $G - S$ is a path of at least $(n - |S|)/(|S| + 1)$ vertices. On the other hand, if the vertices of S are evenly spaced among the n vertices, then every component in $G - S$ contains at most $(n - |S|)/(|S| + 1)$ vertices. Therefore,

$$q(P_n) = 1 + \min_{S \subseteq V} \max\{q(P_{|S|}), q(P_{\lceil (n-|S|)/(|S|+1) \rceil})\}.$$

Solving the equation $|S| = (n - |S|)/(|S| + 1)$ gives $|S| = \sqrt{n+1} - 1$. Therefore, $q(P_n) = 1 + q(P_{\lceil \sqrt{n+1}-1 \rceil})$, which yields $q(P_n) = \lceil \lg \lg(n+1) \rceil$. □

THEOREM 2.12. *If G is a tree of n vertices, then $q(G) \leq \lg \lg n + 3$, and the queries can be constructed in polynomial time.*

Proof. By removing a single vertex, an n-vertex tree can be partitioned into components having at most $n/2$ vertices each. (Pick any vertex v in the tree; if some component in $G - \{v\}$ has more than $n/2$ vertices, move to the neighbor of v in that component and repeat until all components have size at most $n/2$.)

We iteratively place such splitting vertices into S until each remaining component has at most $2\sqrt{n}$ vertices. This process is modeled by a decomposition tree T. The parents of leaves in T correspond to connected subgraphs of G with at least $2\sqrt{n}$ vertices, so there are at most $\sqrt{n}/2$ of them. When the leaves of T are deleted, we have a tree with at most $\sqrt{n}/2$ leaves and thus fewer than $\sqrt{n}$ vertices, each corresponding to a vertex of S.

If G_S at this point is not a tree, as in Figure 3, we add additional vertices of G to S. Let G' denote the subgraph of G that is the union of all paths in G joining vertices of G. We add to S all vertices that have degree at least 3 in G'. (In Figure 3, one vertex is added.) Since G' has fewer than $\sqrt{n}$ leaves, we add fewer than $\sqrt{n}$ vertices to S. The final set S has fewer than $2\sqrt{n}$ vertices. The graph G_S is the graph obtained from G' by contracting an edge incident to a vertex of degree 2 (unless both endpoints are in S) until no further such operations are available.

Let $f(n) = \max q(G)$, where the maximum is taken over all n-vertex trees. By Theorem 2.7, $f(n) \leq 1 + f(2\sqrt{n})$. This recurrence yields $f(n) \leq \lg \lg n + f(8) = \lg \lg n + 3$. □

Theorem 2.12 is essentially best possible because the test number of the path is within three of this bound.

THEOREM 2.13. *If G is the complete bipartite graph $K_{m,n}$, then*

$$q(G) = \lceil \lg(\min\{m, n\} + 1) \rceil.$$

Proof. Assume without loss of generality that $m \le n$. If we pick any $\lceil m/2 \rceil$ vertices in the partite set of size m to use as S in Theorem 2.7, then

$$\begin{aligned} q(K_{m,n}) &\le 1 + \max\{q(K_{\lceil m/2 \rceil}), q(K_{\lfloor m/2, n})\} \\ &\le 1 + q(K_{\lfloor m/2 \rfloor \rfloor, n}). \end{aligned}$$

With $q(K_{1,n}) = 1$, the recurrence yields $q(K_{m,n}) \le \lceil \lg(m+1) \rceil$.

On the other hand, contracting $m-1$ edges of a matching in $K_{m,n}$ yields a minor K_{m+1}, as illustrated in Figure 2. From Lemma 2.2, $q(K_{m,n}) \ge q(K_{m+1}) = \lceil \lg(m+1) \rceil$. □

Next consider that G is a planar graph. Planar graphs arise naturally when the routing is planar, and direct short faults occur only between wires that are close together [7].

THEOREM 2.14 (planar separator theorem; see Lipton and Tarjan [10]). *Let G be an n-vertex planar graph. In $O(n)$ time we can partition $V(G)$ into three sets, A, B, and C, such that* (1) *no edge has one endpoint in A and the other endpoint in B,* (2) *$|A|, |B| \le 2n/3$, and* (3) *$|C| \le \sqrt{8n}$.*

COROLLARY 2.15. *Let G be an n-vertex planar graph. In $O(n)$ time we can find a set $S \subset V(G)$ such that $|S| \le (12 + 6\sqrt{2})\sqrt{n}$, and each component of $G - S$ has at most $n/4$ vertices.*

Proof. From Theorem 2.14, $V(G)$ can be partitioned into A, B, and C such that there is no edge between A and B, $|A|, |B| \le 2n/3$, and $|C| \le \sqrt{8n}$. We call such a set C a *separator*. The sizes of A and B are bounded by αn and $(1-\alpha n)$ for some $1/3 \le \alpha \le 2/3$. Recursively finding separators C_A for $G[A]$ and C_B for $G[B]$, we have $|C_A \cup C_B| \le \sqrt{8\alpha n} + \sqrt{8(1-\alpha)n} < \sqrt{8n}\sqrt{2}$. We apply Theorem 2.14 recursively for 4 levels, reducing all components among the remaining vertices to order at most $(2/3)^4 n = 16n/81 < n/4$. Let S be the union of all the separators found in this tree of separations. We have

$$|S| \le \sqrt{8n}(1 + 2^{1/2} + 2^{2/2} + 2^{3/2}) = \sqrt{n}(12 + 6\sqrt{2}).$$

Since each C can be found in time linear in the number of vertices, the total time to find S is $O(n)$. □

THEOREM 2.16. *If G is a planar graph of n vertices, then $q(G) \le \frac{1}{2}\lg n + O(1)$, and each query can be constructed in $O(n)$ time.*

Proof. Let $f(n) = \max q(G)$, where the maximum is taken over all n-vertex planar graphs. By Corollaries 2.8 and 2.15,

$$f(n) \le 1 + \max\{\lg(\sqrt{n}(12 + 6\sqrt{2})), f(n/4)\}.$$

By induction it can be shown that $f(n) \le \frac{1}{2}\lg n + c + 1$, where $c = \lg(12 + 6\sqrt{2})$. □

We do not know whether Theorem 2.16 is best possible. The best lower bound is $\lg \lg n$ when G is a path of n vertices. Note that the S-connection graph of a planar graph need not be planar. We leave it as an open problem to close the gap. Please note that since no planar graph contains K_5 or $K_{3,3}$ as a minor, it is not possible to prove any nontrivial lower bound using Lemma 2.2.

Random graphs [2] are generated by letting each edge occur with probability 1/2. When we say *almost every* graph has property X, it means the probability for a random graph on n vertices to have property X intends to 1 as n goes to infinity. We next show that solving the connection class problem for random graphs is almost as hard as that for complete graphs.

THEOREM 2.17. *For almost every graph G, $q(G) \geq \lg n - \frac{1}{2}\lg\lg n + O(1)$. This bound also holds for every graph with at least $n^2/4$ edges.*

Proof. Bollobás, Catlin, and Erdős [3] proved that almost every n-vertex graph has K_m as a minor, where $m = (1+o(1))n/\sqrt{\lg n}$. Lemma 2.2 then yields $q(G) \geq \lg m = \lg n - \frac{1}{2}\lg\lg n + O(1)$ almost always. In fact, *every* graph of n vertices and about $n^2/4$ edges has K_m as a minor, where $m = (1+o(1))n/\sqrt{\lg n}$ (see Bollobás [2, p. 279]). □

In general, computing $q(G)$ appears to be NP-hard, but we have not proved this. Observe that the minimum number of queries to determine whether $F = \overline{K}_n$ equals $\lceil \lg \chi(G) \rceil$. Thus $q(G) \geq \lceil \lg \chi(G) \rceil$, but it is still possible that $q(G)$ is easier to compute.

It is important to clarify that the arguments of all theorems in this section give procedures for generating the first query for the algorithms whose number of queries satisfies the resulting bounds, but the arguments are not recursive algorithms for finding the connection classes. For example, in the proof of Theorem 2.7, we assumed the worst case that $Q(S) = V$ and $Q(S) = G - S$, but the actual response is not necessarily the worst case. Thus we must take $Q(S)$ into account to decide each query to solve a particular instance.

The proof in Theorem 2.7 can be used to design a heuristic for general graphs: Find a vertex separator S, construct G_S and the components in $G - S$, and then recurse for G_S and $G - S$ in parallel. The key is to find a vertex separator S so that $\max\{q(G_S), q(G-S)\}$ is minimized. In general, when S is small (large), $q(G_S)$ is small (large) while $q(G-S)$ is large (small). Therefore, we may experiment on the size of S until we balance $q(G_S)$ and $q(G-S)$.

Algorithm 1 is the adaptive diagnosis algorithm. It iteratively maintains a *component structure* $P = \{(G_i, R_i) : i = 1, 2, \ldots, t\}$, where $\{G_1, G_2, \ldots, G_t\}$ is a collection of adjacency graphs whose vertex sets form a partition of V, and $\{R_1, R_2, \ldots, R_t\}$ is a collection of "representative" subsets of vertices such that $R_i \subseteq V(G_i)$ and $Q(R_i) = V(G_i)$. In other words, each (G_i, R_i) is a restricted connection class problem. This property of R_i's implies that each G_i is a union of components. Algorithm 1 initializes with the component structure $P = \{(G, V(G))\}$ and then refines the partition to reduce the size of each R_i. When the algorithm terminates, every R_i is reduced to a single vertex, and therefore every $V(G_i)$ is one connection class. The hardest step is step 4, where we have to use the theorems in this section to find the set S_i.

For example, Figure 4 considers $G = P_{15}$ and uses Corollary 2.11 to generate the queries. The edges in the adjacency graph G are shown, and the edges in the fault graph F are marked with "×." The dark vertices are vertices in the query set S. After the first query, we have five subgraphs containing potential components, including two in $G[Q(S)]$ and three in $G - Q(S)$. Each subgraph is a path. We use Corollary 2.11 to generate the next query for each subgraph. Because the 7-vertex component in $G[Q(S)]$ was generated by two vertices of S, we know that all but one of its edges are faults, and one additional query at v_8 finishes the restricted problem. After the second query, we have all the connection classes of F.

3. Nonadaptive algorithms. In this section, we study nonadaptive algorithms for solving the connection class problem. A nonadaptive algorithm T is a sequence of queries $S_1, S_2, \ldots, S_t$ and a subroutine that analyzes the responses $Q(S_1), Q(S_2), \ldots, Q(S_t)$ to derive the connection classes. The query sets $S_1, S_2, \ldots, S_t$ are decided before asking any queries. We say that a nonadaptive algorithm T solves the connection class problem for G if, for every $F \subseteq G$, T finds the connection classes of F.

Algorithm 1. ADAPTIVE FULL DIAGNOSIS.
Input: Adjacency graph G.
Output: All connection classes.
1: $P \leftarrow \{(G, V(G))\}$.
2: **Repeat**
3: **For** each $(G_i, R_i) \in P$ **do**
4: Find $S_i \subset R_i$.
5: Query $\cup S_i$, with result $Q \leftarrow Q(\cup S_i)$.
6: $P' \leftarrow \emptyset$.
7: **For** each $(G_i, R_i) \in P$ **do**
8: $U_i \leftarrow Q \cap V(G_i)$.
9: **For** each component C of $G_i[U_i]$ **do**
10: $P' \leftarrow P' \cup \{(C, S_i \cap V(C))\}$
11: **For** each component C of $G_i - U_i$ **do**
12: $P' \leftarrow P' \cup \{(C, R_i \cap V(C))\}$.
13: $P \leftarrow P'$.
14: **Until** $|R_i| = 1$ for all $(G_i, R_i) \in P$.
15: Report each $V(G_i)$ as one connection class.

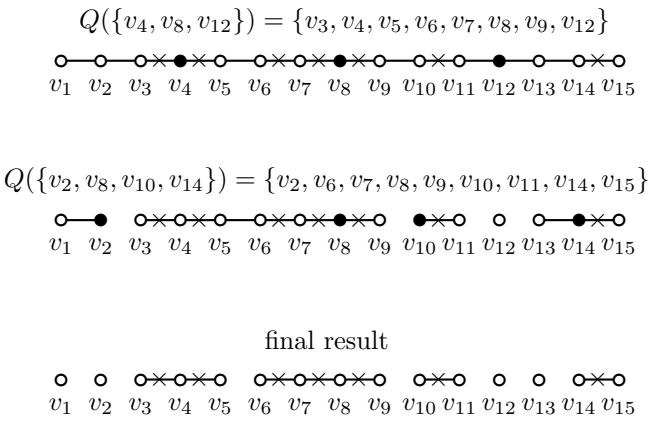


FIG. 4. *Find connection classes of $F \subset P_{15}$ in 2 queries.*

DEFINITION 3.1. *Let $\mathbf{T}_G$ be the set of all nonadaptive algorithms that solve the connection class problem for G. Let the number of queries used by an algorithm T be $t(T)$. The nonadaptive test number $t(G)$ of a graph G is the minimum number of nonadaptive queries that always suffices to solve the following connection class problem of G:*

$$t(G) = \min_{T \in \mathbf{T}_G} t(T).$$

A sequence of queries $S_1, S_2, \ldots, S_t$ defines a *sequential test vector* $X(v_i)$ for each vertex v_i. $X(v_i)$ is a t-bit binary vector $x_{i1}x_{i2}\cdots x_{it}$, where $x_{ij} = 1$ if $v_i \in S_j$, and $x_{ij} = 0$ otherwise. Conversely, sequential test vectors $X(v_1), \ldots, X(v_n)$, where $X(v_i) = x_{i1}x_{i2}\cdots x_{it}$, define a set of queries $S_1, S_2, \ldots, S_t$ by $S_j = \{v_i : x_{ij} = 1\}$. We

will also use sequential test vectors to describe the queries in this section because the vector language underscores the history of queries on each vertex. In Figure 5, each row is a sequential test vector, and each column corresponds to one query.

	test vectors				queries
$X(v_1)$	0	0	1	1	$S_1 = \{v_2, v_5\}$
$X(v_2)$	1	0	0	1	$S_2 = \{v_3\}$
$X(v_3)$	0	1	1	0	$S_3 = \{v_1, v_3, v_5\}$
$X(v_4)$	0	0	0	0	$S_4 = \{v_1, v_2\}$
$X(v_5)$	1	0	1	0	

FIG. 5. *Sequential test vectors and queries.*

It is well known that the sequence of queries $S_1, S_2, \ldots, S_n$, where $S_i = \{v_i\}$ (the so-called walking ones sequence), is sufficient to solve the connection class problem. It is also known that a sequence of queries is sufficient if it contains the walking ones sequence as a subsequence or is diagonally independent. It is less obvious that the queries in Figure 5 can also perform full diagnosis for K_5.

LEMMA 3.2 (see Shi and Fuchs [12]). *A necessary and sufficient condition for sequential test vectors $X(v_1), \ldots, X(v_n)$ to solve the connection class problem on K_n is that for any disjoint nonempty vertex sets U, V, $\vee_{u\in U} X(u) \neq \vee_{v\in V} X(v)$, where the operation $\vee$ is the bit-wise Boolean OR.*

Lemma 3.2 does not give an efficient method to check whether the set of queries can perform full diagnosis because the number of such V_1 and V_2 subsets is exponential in terms of n. Next, we prove that it is unlikely that there exists any polynomial time verifiable characterization.

THEOREM 3.3. *It is NP-hard to tell whether a set of sequential test vectors can solve the connection class problem for G, even if $G = K_n$.*

Proof. By Lemma 3.2, it is sufficient to show the following problem is NP-hard.

Problem Π: Given a set of t-bit binary vectors $T = \{X_1, \ldots, X_n\}$, are there disjoint nonempty subsets of indices I and J such that $\vee_{i\in I} X_i = \vee_{j\in J} X_j$?

We use a reduction from the *not-all-equal* $3SAT$ problem, which is a known NP-complete problem [6], to prove Π is NP-hard. Due to the page limit, the details of the reduction are omitted and can be found in [11]. □

THEOREM 3.4. *A necessary and sufficient condition for sequential test vectors $X(v_1), \ldots, X(v_n)$ to perform full diagnosis for G is that $\vee_{v\in U} X(v) \neq \vee_{v\in W} X(v)$ whenever U, W are disjoint nonempty subsets of $V(G)$ such that $G[U]$, $G[W]$, and $G[U \cup W]$ are connected graphs.*

Proof. If the condition fails and there are U and W as described, then consider fault graphs F_1 and F_2 of G. Graph F_1 has components $G[U]$, $G[W]$, and the rest are isolated vertices. Graph F_2 has component $G[U \cup W]$, and the rest are isolated vertices. Clearly, F_1 and F_2 have different connection classes. However, the responses from the oracle for F_1 and F_2 are the same.

Conversely, suppose that the condition holds. First partition the set of vertices into disjoint subsets according the response of each query as follows. Before the ith step, suppose that we have partitioned $V(G)$ into $V_1, \ldots, V_m$. The response of the next query $Q(S_i)$ further partitions each V_j into $V_j \cap Q(S_i)$ and $V_j - Q(S_i)$. When we finish all the queries, report each maximal vertex set inducing a connected subgraph of G that lies in a single V_i as one connection class of the fault graph F. We show

that this algorithm works correctly.

If C is a connection class of F, then $C \subseteq Q(S)$ or $C \cap Q(S) = \emptyset$ for every $S \subseteq V(G)$. Thus all of C remains in the same V_k, and all of C will be reported to be in the same connection class.

If connection classes $C_1, \ldots, C_k$ of F are reported as being a single connection class D, then $G[D]$ is connected because each set reported as a connection class induces a connected subgraph of G. Also, at each iteration the algorithm leaves the entire set D unpartitioned. Thus each $Q(S_i)$ contains all or none of D. Thus S_i contains a vertex of some C_j if and only if it contains a vertex of each C_j. In particular, $\vee_{v \in C_j} X(v)$ is the same for all j. Letting $U = C_1$ and $W = C_2 \cup \cdots \cup C_k$ yields a violation of the condition. □

Ideas like those in section 2 allow us to compute nonadaptive test numbers of some graphs.

LEMMA 3.5. *If G_1 is a minor of G_2, then $t(G_1) \leq t(G_2)$.*

Proof. The proof is similar to Lemma 2.2. □

LEMMA 3.6. *If graphs G_1 and G_2 are disjoint, then*

$$t(G_1 + G_2) = \max\{t(G_1), t(G_2)\}.$$

Proof. The proof is similar to Lemma 2.3. □

THEOREM 3.7. *For a graph G with vertex set V,*

$$t(G) \leq 1 + \min_{S \subseteq V}\{t(G_S) + t(G - S)\}.$$

Proof. For the set S achieving the minimum, we make the queries consisting of S, a minimum set of queries for G_S, and a minimum set of queries for $G - S$. Each resulting sequential test vector $X(v)$ is the concatenation of one truth bit for $v \in S$, the sequential test vector for v in the query set for G_S, and the sequential test vector for v in the query set for $G - S$.

Let U, W be disjoint nonempty subsets of $V(G)$ such that $G[U]$, $G[W]$, and $G[U \cup W]$ are connected. It suffices to show that $\vee_{v \in U} X(v) \neq \vee_{v \in W} X(v)$.

If U, W both lie outside S, then U, W and $U \cup W$ induce connected subgraphs in $G - S$. Thus the last $t(G - S)$ bits of the sequential test vectors yield the desired nonequality.

If exactly one of U, W intersects S, then query S yields the desired nonequality, since $\vee_{v \in U} X(v)$ and $\vee_{v \in W} X(v)$ differ in the first bit.

Finally, suppose that both U and W intersect S. Let $U' = U \cap S$ and $W' = W \cap S$. By construction, $U', W', U' \cup W'$ all induce connected subgraphs of G_S. Thus the $t(G_S)$ bits of the sequential test vectors corresponding to G_S yield the desired nonequality. □

The algorithm of Feng, Huang, and Lombardi [5] is a special case of Theorem 3.7 by letting G_S to be $K_{|S|}$.

Shi and Fuchs [12] proved $t(K_n) = n - 1$, using general consequences of Lindström [9] and Tverberg [14]. We now consider other families of graphs.

THEOREM 3.8. *If G is the complete bipartite graph $K_{m,n}$, then $t(G) = \min\{m, n\}$.*

Proof. The proof is similar to Theorem 2.13. □

THEOREM 3.9. *For the n-vertex path P_n, $t(P_n) = \lceil \log n \rceil$.*

Proof. For the upper bound, pick the center vertex as S in Theorem 3.7. Then $t(P_n) \leq 1 + t(P_{\lfloor n/2 \rfloor})$. Solving the recurrence relation with $t(P_1) = 0$ gives $t(P_n) \leq \lfloor \lg n \rfloor$.

For the lower bound, let U be the set of $\lceil n/2 \rceil$ vertices closest to one end of the path, and let W be the set of the remaining $\lfloor n/2 \rfloor$ vertices closest to the other end. Note that U, W, and $U \cup W$ all induce connected subgraphs. For sequential test vectors solving the connection class problem, we must have $\vee_{v \in U} X(v) \neq \vee_{v \in W} X(v)$. Let j be a coordinate where they differ. Then elements of U (or W) have test vectors that are all 0 in coordinate j. That means we can solve the connection class problem for U (or W) without using query S_j. Therefore, $t(P_{\lfloor n/2 \rfloor}) \leq t(P_n) - 1$ and $t(P_1) = 0$. Solving the recurrence relation yields $t(P_n) \geq \lfloor \lg n \rfloor$. □

THEOREM 3.10. *If G is a tree of n vertices, then $t(G) \leq \lfloor \lg n \rfloor$, and the queries can be constructed in $O(n \log n)$ time.*

Proof. As in Theorem 2.12, each n-vertex tree can be divided into subtrees of order at most $\lfloor n/2 \rfloor$ by removing one vertex. Let $f(n) = \max t(G)$, where the maximum is taken over all n-vertex trees. From Theorem 3.7, $f(n) \leq 1 + f(\lfloor n/2 \rfloor)$. Solving the recurrence relation with $f(1) = 0$ yields $f(n) \leq \lfloor \lg n \rfloor$. The time to find each separator is linear in the number of vertices. The depth of the recursion is $O(\log n)$. Therefore, the total time to generate the queries is $O(n \log n)$. □

The upper bound in Theorem 3.10 holds with equality for paths by Theorem 3.9.

THEOREM 3.11. *If G is an n-vertex planar graph, then $t(G) = O(\sqrt{n})$, and the set of queries can be computed in polynomial time.*

Proof. Let $f(n) = \max t(G)$, where the maximum is taken over all n-vertex planar graphs G. From Theorems 2.14 and 3.7, $f(n) \leq \sqrt{8n} + f(2n/3)$, which yields $f(n) = O(\sqrt{n})$. The time to find each separator is $O(n)$. The time to find all separators is $O(n \log n)$ since the depth of the recursion is $O(\log n)$. □

THEOREM 3.12. *For almost every graph G, $t(G) \geq (1 + o(1))n/\sqrt{\lg n}$.*

Proof. The proof is similar to Theorem 2.17. □

Let $\chi(G)$ be the chromatic number of G. Since $\lg \chi(G)$ queries are necessary to tell whether $F = \overline{K}_n$, at least this many queries are needed in the worst case to find all the connection classes. Therefore, $t(G) \geq \lg \chi(G)$. If the conjecture of Hadwiger [3] is true, then $t(G) \geq \chi(G) - 1$. Hadwiger's conjecture states that each graph G has $K_{\chi(G)}$ as a minor.

4. Discussion. We presented new adaptive and nonadaptive algorithms for interconnect diagnosis. The adaptive algorithms reduce the number of tests exponentially compared with traditional nonadaptive algorithms. We also show that structural information can further reduce the number of tests drastically for certain sparse graphs such as planar graphs. For dense graphs, there is not much gain using structural diagnosis.

Our results for special families of graphs are summarized in Table 3. In the table, n is the number of vertices of G except for $K_{m,n}$, and all fractions are rounded to the ceiling unless otherwise specified. The *lower bound* for a family is the maximum number of tests necessary for any graph in that family. The *upper bound* for a family is the number of tests sufficient for all graphs in that family. The computation time for generating the queries is low-order polynomial for all the algorithms in Table 3. The results for K_n were first given by Shi and Fuchs [12], but we include them here for completeness.

To use Theorems 2.7 and 3.7 for general graphs, it is crucial that we find good multiway vertex separators. Unfortunately, there are few practical algorithms for finding good vertex separators of general graphs.

TABLE 3
Number of tests for full diagnosis of special families of adjacency graphs.

Adaptive		K_n	$K_{m,n}$ $m \le n$	Path	Tree	Planar graph	Random graph
yes	lower bound	$\lg n$	$\lg(m+1)$	$\lg\lg(n+1)$	$\lg\lg(n+1)$	$\lg\lg(n+1)$	$\lg n - \frac{1}{2}\lg\lg n$
	upper bound	$\lg n$	$\lg(m+1)$	$\lg\lg(n+1)$	$\lg\lg n + 3$	$\frac{1}{2}\lg n$	$\lg n$
no	lower bound	$n-1$	m	$\lfloor \lg n \rfloor$	$\lfloor \lg n \rfloor$	$\lfloor \lg n \rfloor$	$n/\sqrt{\lg n}$
	upper bound	$n-1$	m	$\lfloor \lg n \rfloor$	$\lfloor \lg n \rfloor$	$O(\sqrt{n})$	$n-1$

REFERENCES

[1] M. ABRAMOVIC, M. A. BREUER, AND A. D. FRIEDMAN, *Digital System Testing and Testable Design*, Computer Science Press, Woodland Hills, CA, 1990.

[2] B. BOLLOBÁS, *Random Graphs*, Academic Press, New York, 1985.

[3] B. BOLLOBÁS, P. CATLIN, AND P. ERDŐS, *Hadwiger's conjecture is true for almost every graph*, European J. Combin., 1 (1980), pp. 195–199.

[4] W.-T. CHENG, J. L. LEWANDOWSKI, AND E. WU, *Optimal diagnostic methods for wiring interconnects*, IEEE Trans. Computer-Aided Design, 11 (1992), pp. 1161–1166.

[5] C. FENG, W. HUANG, AND F. LOMBARDI, *A new diagnosis approach for short faults in interconnects*, in Proceedings of the 1995 Fault Tolerant Computing Symposium, pp. 331–339.

[6] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability—A Guide to the Theory of NP-Completeness*, Freeman, New York, 1979.

[7] M. GAREY, D. JOHNSON, AND H. SO, *An application of graph coloring to printed circuit testing*, IEEE Trans. Circuits and Systems, 23 (1976), pp. 591–599.

[8] W. H. KAUTZ, *Testing for faults in wiring networks*, IEEE Trans. Comput., 23 (1973), pp. 358–363.

[9] B. LINDSTRÖM, *A theorem on families of sets*, J. Combin. Theory Ser. A, 13 (1972), pp. 274–277.

[10] R. J. LIPTON AND R. E. TARJAN, *A separator theorem for planar graphs*, SIAM J. Appl. Math., 36 (1979), pp. 177–189.

[11] W. SHI, *Complexity of Finding Two Disjoint Subsets That Have the Same Union*, Technical Report TAMU-ECE-2001-03, Department of Electrical Engineering, Texas A&M University, College Station, TX, 2001.

[12] W. SHI AND W. K. FUCHS, *Optimal interconnect diagnosis of wiring networks*, IEEE Trans. on VLSI, 3 (1995), pp. 430–436.

[13] W. SHI AND D. B. WEST, *Diagnosis of wiring networks: An optimal randomized algorithm for finding connected components of unknown graphs*, SIAM J. Comput., 28 (1999), pp. 1541–1551.

[14] H. TVERBERG, *On equal unions of sets*, in Studies in Pure Mathematics, L. Mirsky, ed., Academic Press, London, 1971, pp. 249–250.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 524–539

RESERVING RESILIENT CAPACITY IN A NETWORK*

G. BRIGHTWELL†, G. ORIOLO‡, AND F. B. SHEPHERD§

Dedicated to the memory of Ewart Lowe

Abstract. We examine various problems concerning the reservation of capacity in a given network, where each arc has a per-unit cost, so as to be "resilient" against one or more arc failures. For a given pair (s,t) of nodes and demand T, we require that, on the failure of any k arcs of the network, there is sufficient reserved capacity in the remainder of the network to support an (s,t) flow of value T. This problem can be solved in polynomial time for any fixed k, but we show that it is NP-hard if we are required to reserve an integer capacity on each arc.

We concentrate on the case where the reservation has to consist of a collection of arc-disjoint paths: here we give a very simple algorithm to find a minimum cost fractional solution, based on finding successive shortest paths in the network. Unlike traditional network flow problems, the integral version is NP-hard: we do, however, give a polynomial time $\frac{15}{14}$-approximation algorithm in the case $k = 1$ and show that this bound is best possible unless P = NP.

Key words. network flows, resilience, capacity reservation

AMS subject classifications. 90B10, 90B25

PII. S0895480100368189

1. Introduction.

A commonly encountered network design problem is that of reserving capacities in a network so as to support a given set of pairwise traffic demands. Algorithms for this *network capacity allocation problem* have been developed by a number of groups; see, for example, [6, 8, 9, 19, 21, 22, 23]. One significant drawback to this "vanilla" capacity reservation model is that it does not account for the failure of certain network elements. For instance, if we simply reserve capacity for a commodity along a single path, we make ourselves totally vulnerable to the failure of any arc (or node) along this path. In many practical settings, this is not acceptable, and we thus wish to reserve our capacities so as to be resilient to certain failure states in the network.

Several groups have recently addressed this issue of "resilience" or "survivability" in network design problems; see, e.g., [2, 3, 5, 10, 14, 15, 20, 24, 25, 26, 27]. Their solution techniques are based primarily on polyhedral or branch and cut methods, and hence produce exact optimal solutions if they terminate, and usually give some guarantee of optimality even before terminating. Such techniques are not always the right selection in a given scenario. On one hand, the need for exact solutions must be balanced with the degree to which the input costs and data are known or certain. These methods also do not exhibit polynomially bounded running time, and hence performance may not scale well as network sizes grow. This may prove to be an even

*Received by the editors February 22, 2000; accepted for publication (in revised form) July 25, 2001; published electronically October 23, 2001. A longer version of this paper appears as [11].
http://www.siam.org/journals/sidma/14-4/36818.html

†Centre for Discrete and Applicable Mathematics, London School of Economics, Houghton Street, London WC2A 2AE, UK (graham@tutte.lse.ac.uk). The research of this author was supported by EU-HCM grant CMRX-CT98 0202 DONET and was carried out in part while visiting the University of Memphis.

‡Dept. of Computer Science, Systems and Production, University of "Tor Vergata", 110-00133 Rome, Italy (oriolo@disp.unirom2.it). The research of this author was supported by EU-HCM grant CMRX-CT98 0202 DONET.

§Bell Laboratories, Lucent Technologies, 600 Mountain Avenue, Murray Hill, NJ 07974-0636 (bshep@research.bell-labs.com).

larger issue for resilient capacity reservation, which initially appears to be a much less tractable problem computationally (cf. [10]). In addition, many applications require solutions in time scales which force the adoption of heuristics with fast run-time properties.

Many existing network planning tools for the (vanilla) capacity allocation problem resort to some form of repeated single-source-destination heuristics. That is, for each demand pair (s_i, t_i) a shortest path is found and then as much flow as possible is pushed on this path. The process is then repeated until all demands are met. In its most general form, the cost of an edge is updated as a function of its remaining capacity. This approach is fast and allows for trivial implementation in software. In its simpler forms, however, one easily concocts examples where it produces solutions arbitrarily far from the optimum. Of course, there are many examples where the exact methods do not even find feasible solutions in a comparable running time. Moreover, they can require substantial mathematical sophistication on the part of its implementors.

Another situation where the single source-destination pair model applies is in an on-line setting. Here, the demands are being given sequentially as they arise in the network. This area is becoming increasingly important as network management becomes a concern of network operators. This is heightened by the changing nature of demands from customers. In particular, bursty or short-term data transfers are becoming increasingly common. As a result, larger amounts of point to point bandwidth are being traded on shorter time frames.

The present paper is dedicated to adapting the shortest path (or mincost flow) model to a *minimum cost resilient capacity reservation* model. We are restricting ourselves to the study of resilient capacity allocation for traffic generated by a single source-destination pair of nodes; we show that even this case presents some surprising difficulties. Overviews of previous computational and theoretical work on related survivability and augmentation problems can be found in [18] and [17].

We adopt the viewpoint that the network, with specified nodes s and t, is given to us, along with a per-unit cost c_a for each arc a, and that we are free to reserve, once and for all, as much capacity as we like on whatever arcs we choose. Our objective is to find a "reservation vector" x minimizing the *total cost*, $\text{cost}(x) = \sum_a c_a x_a$, subject to supporting a given target amount T of traffic from s to t, even if any one, or more generally any k, of the arcs in the network fails.

This rough description of the problem admits many different versions, depending on the type of network we are dealing with, the way we are required to recover from arc failures, and especially on any structure imposed on the vector of reserved capacities itself. In this paper, we consider two types of constraints on the capacity reservation vector.

1. *Integrality.* We may be forced to reserve capacity in discrete amounts, so that our reservation vector must be integral.

2. *Structural.* Specifically, it may be required that our reservation vector be formed by selecting a collection of disjoint (s,t)-paths (i.e., directed paths from s to t in the network) and assigning a capacity to each path. (We call such a reservation a *diverse-path reservation.*)

Diverse-path solutions have several features which are attractive to network planners. For a start, a diverse-path routing may be "hardwired" at the terminating nodes, thus decreasing routing complexity. If traffic flow control is centralized, then this allows load balancing of traffic over the collection of diverse paths. Even if this is

not the case, as in a noncooperative network, the restoration phase is much simpler since an arc failure may be treated as a path failure, and traffic routed along the path may be shifted to the remaining nonfailed paths. One final advantage is that they are conceptually simple to visualize and work with in any operational setting.

2. Summary of results. In this paper, we consider various versions of the resilience problem; we summarize these and our results in this section.

Throughout, we suppose (sometimes implicitly) that we are given a directed graph (network) $D = (V, A)$ with *node set* V and *arc set* A. We always assume that D comes with two nodes permanently fixed as the *source* s and the *destination* t. We also suppose that we are given a rational number T (usually an integer) representing the required traffic flow from s to t through the network D in the case of failure and an integer k representing the maximum number of arcs that may fail. Finally, we are also given a vector (c_a) of nonnegative rational (again, usually integer) costs on the arcs a of D. We are seeking a *reservation vector*, which is a nonnegative vector (x_a) on the arcs a, representing the amount of capacity reserved on the arcs of D.

A reservation vector $x = (x_a)$ is (T, k)-*resilient* if, for each set K of at most k arcs in A, the reserved capacities on the arcs in $A - K$ are sufficient to admit an (s, t) flow of value T. The *cost* of a reservation vector x is $\text{cost}(x) = \sum_{a \in A} c_a x_a$. Our aim in all versions considered is to find a minimum cost (T, k)-resilient reservation vector x in D.

The problem stated above is the General Resilience problem. We do not normally treat k as part of the input, so an instance of this problem, for resilience against k failures, consists of a network D, with specified source and destination, a demand T, and a cost vector c on the arcs of D. If the reservation vector x is required to be an integer, then the demand T and all the costs must also be integers, and we have the Integer Resilience problem.

As explained in section 3, for each fixed k, General Resilience can be solved in polynomial time using linear programming or the ellipsoid algorithm. However, we show in section 7 that Integer Resilience is strongly NP-hard even for $k = 1$, although we do give a simple $(k + 1)$-approximation algorithm.

For most of the paper, we concentrate on the case where the reservation vector is also required to be a *diverse-paths reservation*; i.e., it is derived from a set $P_1, \ldots, P_m$ of arc-disjoint (s, t)-paths with capacity x_j reserved on each arc of path P_j $(j = 1, \ldots, m)$.

In section 4, we consider the case where the (s, t)-paths $P_1, \ldots, P_m$ to be used are prespecified. In this case, we may as well consider each path as a single arc from s to t, whose cost is the sum of the individual arc-costs, and we have the k-Failure Allocation problem, an instance of which consists simply of a demand T and a sequence of costs $c_1, \ldots, c_m$, where we assume that $c_1 \leq \cdots \leq c_m$. The problem is to find a nonnegative real vector $x = (x_1, \ldots, x_m)$ to minimize $\text{cost}(x) = \sum c_j x_j$ subject to the (T, k)-resilience constraint that the sum of any $m - k$ of the x_j is at least T. If we additionally impose the constraint that the x_j be integers, then we have the Integer k-Failure Allocation problem.

We give an extremely simple algorithm for solving both k-Failure Allocation and its integer counterpart. For the case $k = 1$, a polynomial time algorithm (for the integer version) based on convex optimization is already explained in the work of Bartholdi, Orlin, and Ratliff [7], which addresses certain integer programming problems arising from matrices with the circular ones properties in each row. Our results provide additional information on the structure of the optimal fractional and

integral solutions, which is needed later in the paper when we study the INTEGER DIVERSE-PATH RESILIENCE problem without the paths being prefixed. Some of these structural properties were independently observed by Bienstock and Muratore [10], who gave a complete linear description for an associated polyhedron.

In sections 5 and 6, we turn to the case when our diverse-path reservation x may use any set of diverse (s,t)-paths. The problems we consider are the DIVERSE-PATH RESILIENCE problem and INTEGER DIVERSE-PATH RESILIENCE problem, which are exactly the same as GENERAL RESILIENCE and INTEGER RESILIENCE except that the reservation vector x is required to be a diverse-path reservation.

Using the information from section 4 about the nature of any optimal diverse-path reservation, we give a simple combinatorial algorithm, based on finding successive shortest paths in the network, to solve DIVERSE-PATH RESILIENCE.

However, INTEGER DIVERSE-PATH RESILIENCE turns out to be NP-hard, even for $k = 1$. Here instead we give a polynomial time $\frac{15}{14}$-approximate algorithm in the case $k = 1$ and show this bound is the best possible (if P $\neq$ NP). Similar results hold if k takes other values or is unrestricted.

We conclude the paper with a discussion of a possible application to the case of more than one source-destination pair and a few remarks about other types of resilience problems.

3. Polyhedral formulation. Given a directed graph $D = (V, A)$, and any $S \subseteq V$, let $\delta^+(S)$ denote the set of arcs with tail in S and head in $V - S$ and set $\delta^-(S) = \delta^+(V - S)$. We call $S \subseteq V$ an (s,t)-*set* if $s \in S$ and $t \in V - S$.

Let $\mathbb{Q}_+$ denote the set of nonnegative rational numbers, so that $\mathbb{Q}_+^A$ is the set of all assignments of nonnegative rationals to each member of the arc-set A, which we frequently view as a vector. For any vector $x \in \mathbb{Q}_+^A$ and $A' \subseteq A$, we denote by $x(A')$ the sum $\sum_{a \in A'} x_a$.

The problem of finding a minimum cost (T,k)-resilient reservation vector can be expressed as an optimization problem over a certain polyhedron, which we now describe.

For any rational T, (s,t)-set S, and set $K \subseteq \delta^+(S)$ of at most k arcs, the *partial T-cut constraint* associated with the pair (S,K) is the constraint $x(\delta^+(S) - K) \geq T$. The *resilience polyhedron* is defined by the following system of all partial cut constraints:

(3.1)
$$\mathcal{R}(T,k,D) = \left\{ x \in \mathbb{Q}_+^A \quad : \quad x(\delta^+(S) - K) \geq T \quad \begin{array}{l} \text{for each } (s,t)\text{-set } S \\ \text{and } K \subseteq \delta^+(S) \text{ with } |K| \leq k \end{array} \right\}.$$

Note that $\mathcal{R}(T,k,D)$ is empty if there is an (s,t)-set S with $\delta^+(S)$ of size at most k, and otherwise $\mathcal{R}(T,k,D)$ is full-dimensional. It is straightforward to verify that $\mathcal{R}(T,k,D)$ consists exactly of the (T,k)-resilient vectors, and so the GENERAL RESILIENCE problem—finding a minimum cost (T,k)-resilient reservation—is that of minimizing the linear function $\text{cost}(x) = \sum_{a \in A} c_a x_a$ over $\mathcal{R}(T,k,D)$.

A consequence of this formulation is that, for each fixed k, there is a polynomial time algorithm to solve GENERAL RESILIENCE. Indeed, it is easily seen that the separation problem for $\mathcal{R}(T,k,D)$ amounts to solving at most $|A|^k$ maximum flow problems. Moreover, the problem can be rephrased as that of finding a single edge capacity vector x, together with an (s,t) flow vector y^K of value T for each failing set K of size k (i.e., with $y_a^K = 0$ for $a \in K$), subject to the constraint $y_a^K \leq x_a$ for each arc a and each failing set K. This formulation constitutes a linear program with a polynomially bounded number of variables and constraints. This is not, however,

offered as a practical approach, even for $k = 1$, and the remainder of the paper addresses the task of finding more direct combinatorial algorithms.

4. Reservations on a fixed set of paths. A version of the material in this section, written for a nontechnical audience, appears as [13]. A more thorough handling of the polyhedron considered implicitly herein (including a complete linear description of the integer hull) has been given by Bienstock and Muratore [10]. The case $k = 1$ can also be regarded as a special case of a problem treated by Bartholdi, Orlin, and Ratliff [7]; our methods give a somewhat simpler solution in this special case.

We start by considering k-Failure Allocation and its integer version. Recall that these problems can be formulated as follows.

k-Failure Allocation. Given a demand T, and a sequence of nonnegative costs $c_1 \le c_2 \le \cdots \le c_m$, find a nonnegative real vector $x = (x_1, x_2, \ldots, x_m)$ minimizing $\text{cost}(x) = \sum_{i=1}^{m} c_i x_i$ subject to the constraint that $\sum_{i \notin K} x_i \ge T$ for every set K of size k.

Integer k-Failure Allocation. As above, with x required to be integer.

We start with a result that is fundamental in much of the rest of the paper. For $k < j \le m$, let $z^{j,k}$ be the reservation vector defined by

$$z_i^{j,k} = \begin{cases} T/(j-k), & i \le j, \\ 0, & i > j. \end{cases}$$

Theorem 4.1. *An optimal solution to* k-Failure Allocation *is obtained at one of the solutions* $z^{j,k}$.

Proof. Because of the symmetry of the situation and the ordering of the costs c_i, it is clear that there is an optimal solution such that $x_1 \ge x_2 \ge \cdots \ge x_m$. Thus we lose nothing by including these inequalities as constraints. Once we do this, we see that, if the constraint $x_{k+1} + x_{k+2} + \cdots + x_m \ge T$ is satisfied, all the other resilience constraints given by the removal of k of the paths are automatically satisfied. Thus we may reformulate the problem as follows.

Given a demand T*, and a sequence of costs* $c_1 \le c_2 \le \cdots \le c_m$*, find nonnegative real numbers* $x_1, \ldots, x_m$ *to minimize* $\sum_i c_i x_i$ *subject to the constraints* $x_1 \ge x_2 \ge \cdots \ge x_m \ge 0$ *and* $x_{k+1} + \cdots + x_m \ge T$.

We note for future reference that the same reformulation goes through if the x_i are all constrained to be integers.

Consider a basic optimal solution for the resulting linear program which necessarily satisfies m linearly independent inequalities with equality. If there are j nonzero variables at the optimum, then the only possibility is that all of the inequalities $x_1 \ge x_2$, ... , $x_{j-1} \ge x_j$, $x_{j+1} \ge \cdots \ge x_m \ge 0$, and $x_{k+1} + \cdots + x_m \ge T$ are satisfied with equality, i.e., that $x_1 = x_2 = \cdots = x_j = T/(j-k)$ and $x_{j+1} = \cdots = x_m = 0$. This is just the solution $z^{j,k}$, and the result follows. □

Solving k-Failure Allocation thus amounts to choosing amongst the solutions $z^{j,k}$. In fact, the structure of the problem allows a particularly simple procedure for doing this. Note that $\frac{1}{T}\text{cost}(z^{j,k})$ is equal to $A_{j,k} = (c_1 + \cdots + c_j)/(j-k)$, which is the average of $c_1 + c_2 + \cdots + c_{k+1}$, c_{k+2}, ... , and c_j. The $A_{j,k}$ are decreasing in j up to the minimum, which is attained for the last j where $c_j < A_{j-1,k}$, and increasing thereafter: this unimodality property will be a recurring theme. Thus we may terminate our search for the minimum value of $\text{cost}(z^{j,k})$ if ever we find that $c_{j+1} \ge A_{j,k}$.

For Integer k-Failure Allocation, we show that the following procedure suffices.

First find the optimal fractional solution $z^{j,k}$, i.e., the optimal solution of the corresponding instance of k-Failure Allocation. Then (if $z^{j,k}$ is not already an integer vector) consider the two integer solutions "nearest" to $z^{j,k}$, as follows.

(a) Set r equal to either $\lfloor T/(j-k) \rfloor$ or $\lceil T/(j-k) \rceil$. (Here $\lceil a \rceil$ denotes the next integer above the real number a and $\lfloor a \rfloor$ the next integer below.) Note that r may be zero if $T < j-k$.

(b) If r is one of the two chosen values and r is nonzero, we attempt to construct a solution x with all the nonzero x_i, except possibly one, equal to r. To do this, we set $\ell = \lceil T/r \rceil + 1$; if $\ell \le m$, set $x_1 = x_2 = \cdots = x_{\ell-1} = r$, $x_\ell = T - (\ell - k - 1)r$, and $x_{\ell+1} = \cdots = x_m = 0$. (The choice of ℓ ensures that $x_\ell \le x_1 = r$. If $r = \lfloor T/(j-k) \rfloor$, we could have $\ell > m$, but this is not possible with $r = \lceil T/(j-k) \rceil$.)

Note that this is a feasible solution, since removing any k of the x_i leaves capacity at least $(\ell - k - 1)r + x_\ell$, which is constructed to be at least T.

(c) We now have either one or two candidate integral solutions corresponding to the two choices of r in (a). We denote by $z^{j,k,+}$ the solution with $r = \lfloor T/(j-k) \rfloor$ (if it is feasible) and by $z^{j,k,-}$ the solution with $r = \lceil T/(j-k) \rceil$ (which is always feasible).[1] To finish, just calculate the costs of the two solutions and choose the lower.

Theorem 4.2. *Suppose we have an instance of* k-Failure Allocation *in which the optimal solution is* $z^{j,k}$. *Then the optimal solution* x *to the corresponding instance of* Integer k-Failure Allocation *is either* $x = z^{j,k,+}$ *or* $x = z^{j,k,-}$.

Proof. We work with the reformulation of the problem as in the beginning of the proof of Theorem 4.1, which, as we noted, is also valid for the integer case. Suppose that x is an optimal integer solution.

Clearly, we have $x_1 = x_2 = \cdots = x_{k+1}$ at the optimum. Now suppose that some x_j is nonzero but that not all of $x_1, \ldots, x_{j-1}$ are equal. Then let i be the minimum index with $x_i < x_1$; note that $i \ge k+2$ by the previous observation. Also let x_ℓ be the last nonzero variable, so $k+2 \le i < j \le \ell$. Increasing x_i by one and decreasing x_ℓ by one keeps the solution feasible, and $x_{k+1} + \cdots + x_m$ is unaltered. Also, this operation does not increase the cost.

We have thus shown that one may restrict attention to integral solutions where there is some j, with $k+2 \le j \le m$, such that all of $x_1, \ldots, x_{j-1}$ are equal, and all of $x_{j+1}, \ldots, x_m$ are equal to 0. If the value of x_j is q, then the common value of the earlier x_i is $(T-q)/(j-k-1)$, which is an integer, at least q.

At this point, there is still potentially one solution for each integer value $x_1 \ge T/m$, namely, to set $j = k + \lceil T/x_1 \rceil$, and $q = T - (j-k-1)x_1$; observe that this solution is equal to

$$\frac{q(j-k)}{T} z^{j,k} + \left(1 - \frac{q(j-k)}{T}\right) z^{j-1,k}.$$

Thus each of our candidate integral solutions is a convex combination of two consecutive $z^{i,k}$'s. Let $\mathcal{A}$ be the set of all such convex combinations; we think of $\mathcal{A}$ as a "path" with vertices corresponding to $z^{k+1,k}, \ldots, z^{m,k}$; any vector on this path gives a feasible solution. Note that the first coordinate value x_1 decreases along $\mathcal{A}$, and the solution cost is unimodal along $\mathcal{A}$, since it is linear between the vertices. If the fractional optimum is attained at a vertex with x_1 equal to r, then the lowest cost integer solution on $\mathcal{A}$, and hence the overall integer optimum, is obtained by taking x_1 to be either $\lceil r \rceil$ or $\lfloor r \rfloor$. This amounts to taking either $z^{j,k,+}$ or $z^{j,k,-}$, as required. ∎

[1] The notation $z^{j,k,+}$, $z^{j,k,-}$ indicates moving from $z^{j,k}$ towards $z^{j+1,k}$ or $z^{j-1,k}$.

We close this section with a bound relating the costs of the optimal integral and fractional solutions to k-Failure Allocation.

Proposition 4.3. *For any $j > k \geq 1$, $\mathrm{cost}(z^{j,k,-})/\mathrm{cost}(z^{j,k}) < 1 + \frac{k(j-k)}{jT} < 1 + \frac{k}{T}$.*

Proof. Recall that, in the solution $z^{j,k}$, the j cheapest paths are chosen, each with capacity $T/(j-k)$. The "rounded" solution $z^{j,k,-}$ is obtained from this by taking the $\ell - 1$ cheapest paths with capacity $x = \lceil T/(j-k) \rceil$, and the next cheapest path with capacity $T - (\ell - k - 1)x$, where $\ell = \lceil T/x \rceil + k \leq j$.

The first observation is that the average cost per unit of reservation in $z^{j,k,-}$ is no greater than that in $z^{j,k}$. Thus $\mathrm{cost}(z^{j,k,-})/\mathrm{cost}(z^{j,k})$ is at most the ratio of the total numbers of units of capacity reserved in the two allocations.

In $z^{j,k}$, a total of $jT/(j-k)$ units of capacity are allocated, while in $z^{j,k,-}$, the total is $(\ell - 1)x + T - (\ell - k - 1)x = T + kx$. Hence we have

$$\frac{\mathrm{cost}(z^{j,k,-})}{\mathrm{cost}(z^{j,k})} \leq \frac{T + kx}{jT/(j-k)} = 1 + \frac{k}{jT}\left((j-k)x - T\right).$$

Now $(j-k)x - T < j - k$, by definition of x, so that we have the estimate as claimed. ∎

Corollary 4.4. *If O_I is the cost of the optimal solution to an instance of* Integer k-Failure Allocation *with target flow T, and O_F the cost of the optimal solution to the corresponding instance of* k-Failure Allocation*, then $O_I < (1 + k/T)O_F$.*

Consideration of the proof of Proposition 4.3 shows that the ratio $(1+k/T)$ cannot in general be improved. Indeed, if our network consists of a large number M of paths of cost 1, then it is easy to see that $O_I = T + k$, whereas $O_F = MT/(M-k)$; as $M \to \infty$, $O_I/O_F \to (1 + k/T)$.

5. Diverse-path reservations without specified paths. We now turn to the Diverse-Path Resilience problem, when we are still required to find a resilient reservation consisting of a set of diverse paths in a given network D, but we are not restricted as to what paths we may use. Our aim here is to give a fast algorithm for Diverse-Path Resilience, whatever number k of failures is to be allowed for.

Theorem 4.1 implies that the optimal solution has as support the arcs of some $j > k$ diverse paths with each arc in the support given capacity $T/(j-k)$. Of course, j can take only integer values up to the (s,t)-connectivity $\kappa = \kappa(D)$ of D. We may take advantage of this structure and apply the *successive shortest path method* (cf. [1]) for minimum cost flow problems, thus needing only to solve $\kappa(D)$ shortest path problems.

For an arc $a = (u,v) \in A$, we let a^- denote an "artificial" arc (v,u) not present in D. In the course of the following algorithm, we construct a series of auxiliary digraphs D_j, each of which contains exactly one from each pair a, a^-. We assume that we are given a digraph D with $\kappa(D) > k$.

In the algorithm Paths below, we find a succession of arc-sets $\mathcal{P}_1, \mathcal{P}_2, \ldots$, where each $\mathcal{P}_j$ is the arc-set of a set of j diverse (s,t)-paths of minimum cost. Each $\mathcal{P}_{j+1}$ is derived from $\mathcal{P}_j$ by adding a cheapest path in the network D_j with costs c^j. Adding the arc a^- corresponds to removing the arc a. In line with our earlier notation, $z^{j,k}$ denotes the diverse-path reservation using the paths of $\mathcal{P}_j$. The algorithm terminates if $z^{j+1,k}$ is at least as expensive as $z^{j,k}$.

PATHS(D,k)
{
  $j=0; D_0=D; c^0=c; \mathcal{P}_0=\emptyset;$
  While (D_j contains a directed (s,t)-path)
    Let Q_j be the arc-set of a minimum c^j-cost
      directed (s,t)-path in D_j
    Set $\mathcal{P}_{j+1}=(\mathcal{P}_j-R)\cup F$
      where $R=\{a\in A: a^-\in Q_j\}$
      and $F=A\cap Q_j$
    If $j\geq k$, let $z^{j+1,k}$ be the vector obtained by assigning
      $T/(j+1-k)$ to each arc in $\mathcal{P}_{j+1}$
    If $j\geq k$ and $\text{cost}(z^{j+1,k})\geq \text{cost}(z^{j,k})$
      then Output($z^{j,k}$) and Quit
    D_{j+1}, c^{j+1} are the same as D_j, c^j except
      if $a\in R$
        remove a^-, and include a with cost $c_a^{j+1}=c_a$
      if $a\in F$
        remove a, and include a^- with cost $c_{a^-}^{j+1}=-c_a$
    Set $j=j+1$;
  EndWhile
  Output($z^{j,k}$)
}

We also refer to the version of the algorithm which does not terminate early and thus generates a reservation vector $z^{j,k}$ for every $j=k+1,\ldots,\kappa(D)$.

PROPOSITION 5.1 (cf. [1]). *Let c be a nonnegative vector of arc costs in a network $D=(V,A)$. The algorithm* PATHS *finds a minimum cost (T,k)-resilient diverse-path reservation.*

To establish correctness, we need two facts. First, for each j, the collection $\mathcal{P}_j$ induces a minimum cost collection of j diverse (s,t)-paths; this follows from the correctness of the successive shortest path method. This implies that each solution $z^{j,k}$ is the minimum cost solution using j paths, and hence the minimum cost (T,k)-resilient vector is among these vectors $z^{j,k}$. Moreover, traditional flow theory implies that for each $j\geq k+1$, $z^{j,k}$ is a minimum cost flow of value $jT/(j-k)$ subject to the capacities $T/(j-k)$ on each arc. Second, as we now show, the sequence $\text{cost}(z^{j,k})$ is unimodal for $j\geq k+1$, and so early termination is justified.

PROPOSITION 5.2. *Let h, i, and j be such that $k<h<i<j\leq\kappa(D)$. If $\text{cost}(z^{h,k})\leq\text{cost}(z^{i,k})$, then $\text{cost}(z^{i,k})\leq\text{cost}(z^{j,k})$.*

Proof. Suppose the contrary: there exists h, i, j, with $h<i<j$, such that $\text{cost}(z^{h,k})\leq\text{cost}(z^{i,k})$ and $\text{cost}(z^{i,k})>\text{cost}(z^{j,k})$. Let $M=T\frac{i}{i-k}$ and choose $\lambda\in(0,1)$ such that $\frac{i}{i-k}=\lambda\frac{h}{h-k}+(1-\lambda)\frac{j}{j-k}$. Then $z'=\lambda z^{h,k}+(1-\lambda)z^{j,k}$ is a flow of value $iT/(i-1)$ and does not exceed $T/(i-1)$ on any arc. Thus by the remarks preceding the proposition, $\text{cost}(z')\geq\text{cost}(z^{i,k})$. However, of course $\text{cost}(z')=\lambda\text{cost}(z^{h,k})+(1-\lambda)\text{cost}(z^{j,k})<\text{cost}(z^{i,k})$, a contradiction. □

6. Integer diverse-path reservations. We now turn to INTEGER DIVERSE-PATH RESILIENCE, where we are required to find a minimum-cost diverse-path reservation taking integer values. We assume throughout this section that the demand T is also an integer.

Again the results of section 4 give us information about the structure of an optimal

solution: Theorem 4.2 shows that the support of an optimal solution consists of a collection of diverse (s,t)-paths $P_1, P_2, \ldots, P_j$, where the arcs of the first $j-1$ paths will reserve a common amount, r, of capacity, and the last path's arcs will reserve capacity $T-(j-1-k)r \leq r$.[2] We now show that the subproblem with $k=1$ and $T=3$, denoted by 3-IDP, is NP-hard. Let 2DIV-PATHS denote the problem of determining whether a given digraph D, with four distinct nodes s_1, t_1, s_2, t_2, contains a pair of arc-disjoint paths P_1, P_2, where P_i joins s_i and t_i $(i=1,2)$. Fortune, Hopcroft, and Wyllie [16] show that this problem is NP-complete.

THEOREM 6.1. *The problem* 3-IDP *is NP-hard. Furthermore, unless* $P = NP$, *there is no polynomial time* $(1+\varepsilon)$*-approximation algorithm with* $\varepsilon < \frac{1}{14}$.

Proof. Suppose that we are given an instance of 2DIV-PATHS as above. Construct a digraph obtained from D by adding new nodes s, t as well as the arcs (s,t), (s,s_1), (s,s_2), (t_1,t), and (t_2,t) with costs 3, 1, 2, 1, and 2, respectively. All remaining arcs have cost zero. This is our instance of 3-IDP. From Theorem 4.2, we deduce that an optimal 3-resilient reservation on diverse paths will have either support on (i) 2 diverse paths, in which case capacity 3 is reserved on each of the arcs of these paths, or (ii) 3 diverse paths in which case two of the paths will have reserved capacity 2 and the third capacity 1.

Note that the cheapest collection of 2 diverse paths has cost 5, and hence any solution of the form (i) will have cost at least 15. Next note that if there exists a positive solution to the instance of 2DIV-PATHS, with P_i a path between s_i, t_i $(i=1,2)$, then by assigning capacity 2 to the arc (s,t) and the arcs of P_1, and capacity 1 to the arcs of P_2, we obtain a solution to 3-IDP of cost 14. Conversely, if the instance of 2DIV-PATHS has no solution, then any "3-path" solution to 3-IDP will use only paths of cost 3, from which we deduce that the reservation will cost at least 15. Thus the optimal solution to the instance of 3-IDP is 14 if and only if the instance of 2DIV-PATHS has a positive solution and is otherwise at least 15. The result follows. □

We continue to concentrate on the case $k=1$ but allow T to take arbitrary integer values. We have already seen in Proposition 4.3 that applying the rounding procedure to an optimal fractional solution yields a $(1+\frac{1}{T})$-approximation to the optimal fractional solution and hence to the optimal integral solution. It is clear that the optimal fractional solution is integral in the case $T=2$, so in fact we have a polynomial time $\frac{4}{3}$-approximation algorithm for arbitrary T: we now improve this to a $\frac{15}{14}$-approximation algorithm which, in view of Theorem 6.1, is best possible. Note that, by Proposition 4.3, we may assume that $T \leq 13$.

Consider the polynomial time algorithm $\mathcal{A}$ (based on PATHS) that finds, for each value of j, the fractional solution $z^{j,1}$ based on *some* cheapest set of j diverse paths, and the two "rounded" integer solutions $z^{j,1,-}$ and $z^{j,1,+}$, and chooses the best among all of the integer solutions. The algorithm $\mathcal{A}$ can fail to find the optimal integer solution because it may use a minimum cost set of ℓ paths in which the costs are distributed "more evenly" between the paths (in particular, the most expensive path is cheaper) than in some other (not necessarily even minimum cost) set of ℓ paths. All we know is that an optimal solution for an instance of INTEGER DIVERSE-PATH RESILIENCE has the *same form* as either $z^{\ell,1,-}$ or $z^{\ell,1,+}$ for some ℓ, since it arises from a similar rounding process applied to *some* collection of diverse paths.

Let O_F be the cost of a fractional optimum solution, O_I the cost of an integer

[2] In essence, we thus need to solve an integer 2-multicommodity flow problem where both commodities have the same source and destination.

optimum solution, and O_A the best solution among those considered by the algorithm, i.e., the value returned by $\mathcal{A}$. Clearly, we have $O_F \le O_I \le O_A$.

THEOREM 6.2. *The algorithm $\mathcal{A}$ is a $\frac{15}{14}$-approximate algorithm for* INTEGER DIVERSE-PATH RESILIENCE *with $k = 1$, that is, for each instance $O_A \le \frac{15}{14} O_I$.*

We note that the quality of approximation by the algorithm depends greatly on the input T. If we view $\mathcal{A}$ as an infinite collection of algorithms $\{\mathcal{A}_T\}_{T=1}^{\infty}$, each restricted to instances with a fixed value of the demand T, then many of these—in particular, those with large values of T—are $(1+\epsilon)$-approximate algorithms with $\epsilon < \frac{1}{14}$. Indeed, Proposition 4.3 tells us that, for each T, $O_A < (1 + 1/T)O_F \le (1 + 1/T)O_I$.

Furthermore, we note in the course of the proof that, for $T = 1, 2, 4, 6$, and 12, $\mathcal{A}$ solves the problem exactly.

Proof. Take any instance of the problem, and let z^* be an optimum solution, say, using paths $P_1, \ldots, P_\ell, P_{\ell+1}$. We certainly know that there is no better solution using *these* paths, so by Theorem 4.2 we know that $P_1, \ldots, P_\ell$ all have the same reserved capacity r under z^*, and path $P_{\ell+1}$, which has the greatest cost among these paths, has reserved capacity $y \le r$. Furthermore, r is either $\lfloor T/(\ell-1) \rfloor$ or $\lceil T/\ell \rceil$ with $T = (\ell-1)r + y$.

Clearly, we can assume that $O_I \ne O_A$. In particular, this means that $0 < y < r$; otherwise one of $z^{\ell+1,1}$ or $z^{\ell,1}$ would be an integral solution found by $\mathcal{A}$ whose cost was at most that of z^*. We may thus assume that $\ell \ge 2$ and that $2 \le r \le (T+1)/2$.

The values of T and r determine y and ℓ. Also, as noted above, Proposition 4.3 implies the result for $T \ge 14$, so we may assume that $T \le 13$ and that $r \le \lceil T/2 \rceil \le 7$. There are thus only a finite number of possible forms of z^* (in fact, just 23 pairs (T, r) satisfy all the restrictions mentioned so far), and we rule all of these out using the same basic method. At this point, let us observe that there are no cases with $T = 1, 2, 4, 6$, or 12; for these values of T, any r not dividing T exactly is not of the form $\lfloor T/(\ell-1) \rfloor$ or $\lceil T/\ell \rceil$ for any integer ℓ. In particular, we may assume that we have $T \ge 3$.

We require a lower bound on $\text{cost}(z^*) = O_I$. Notice that z^* can be written as y times the characteristic vector of some set of $\ell + 1$ diverse paths, plus $(r - y)$ times the characteristic vector of some set of ℓ diverse paths. Let C_i be the cost of reserving one unit of capacity on the arcs in a cheapest collection of i diverse paths. Therefore we have $O_I = \text{cost}(z^*) \ge yC_{\ell+1} + (r-y)C_\ell$.

Our algorithm $\mathcal{A}$ considers some integer solution $z^\dagger$ of the same form as z^* (i.e., the same values of T, r, y, ℓ), using some set of $\ell+1$ paths of cost $C_{\ell+1}$. The cost of $z^\dagger$ is at most what it would be if all the $\ell+1$ paths had the same cost $(\ell r+y)C_{\ell+1}/(\ell+1)$. Therefore O_A is at most this quantity, i.e.,

$$C_{\ell+1} \ge \frac{\ell+1}{\ell r + y} O_A = \frac{\ell+1}{T+r} O_A.$$

We aim for a similar bound on C_ℓ, and to get this we need to look at a solution produced by $\mathcal{A}$ on at most ℓ paths. Accordingly, let $r' = \lceil T/(\ell-1) \rceil$; there is some integer solution with reserved capacity r' on the first $m < \ell$ paths from C_ℓ, and $v \le r'$ on one further path, with total of reserved capacities on all the paths equal to $T + r'$. Our algorithm will have looked at an integer solution with a cost at least as low as some solution of this form, and the average cost of a path in any solution of this form is at most C_ℓ/ℓ, as in the proof of Proposition 4.3.

Therefore we have $O_A \le (T+r')C_\ell/\ell$, i.e., $C_\ell \ge \frac{\ell}{T+r'}O_A$. We conclude that

$$O_I \ge \left(\frac{y(\ell+1)}{T+r} + \frac{(r-y)\ell}{T+r'}\right) O_A.$$

After a little manipulation, this becomes

$$\frac{O_I}{O_A} \ge 1 - \frac{(r'-r)(r-y)\ell}{(T+r)(T+r')} \equiv 1 - G.$$

We could now run through all the 23 cases separately and show that $G \le 1/15$ in each case, but we can save a little effort.

First, we consider all cases with $\ell = 2$. In this case, we have $r' = T$, and $r = \lceil T/2 \rceil$, which implies that $r' = T = 2y+1$ and $r = y+1$. Now $G = 2y/(3y+2)(4y+2) \le 1/15$ for $y \ge 1$.

Next, if $r = 2$, we have $y = 1$, $T = 2\ell - 1$, and $r' = 3$. This gives $G = \ell/(2\ell+1)(2\ell+2) \le 1/15$ for $\ell \ge 2$. Assume from now on that $\ell, r \ge 3$. This implies that $T \ge 7$ and that $r < T/2$.

If $r' = r+1$, then, since also $(r-y)\ell < r\ell < T+r$, we have $G < 1/(T+r+1)$, and we are done if $T+r \ge 14$. Therefore we may assume $T \le 10$, whence, since $\ell \ge 3$, we have $r \le 4$. On the other hand, if $r' = \lceil T/(\ell-1) \rceil \ge r+2 \ge (T+1)/\ell + 2$, then we have $T \ge \ell^2 + \ell - 1$; thus the only two cases with $T \le 13$ are $\ell = 3$ and $T = 11, 13$.

From hereon in, there seems to be no great saving on dealing with all the cases individually. Here are all the cases not so far ruled out.

T	r	ℓ	y	r'	G
7	3	3	1	4	6/110
8	3	3	2	4	3/132
9	4	3	1	5	9/182
10	3	4	1	4	8/182
10	4	3	2	5	6/210
11	4	3	3	6	6/255
13	5	3	3	7	12/360

All the values for G above are less than $1/15$, so the theorem is proved. □

It is clear that this technique can be used to prove similar results for other values of k. If k is allowed to take any value, we have the following result.

THEOREM 6.3. *There is a polynomial time $\frac{6}{5}$-approximation algorithm for* INTEGER DIVERSE-PATH RESILIENCE *with k as part of the input. Furthermore, there is no polynomial time $(1+\epsilon)$-approximation algorithm for the problem with $\epsilon < \frac{1}{5}$ unless $P = NP$.*

Proof (sketch). Our algorithm is the obvious adaptation of algorithm $\mathcal{A}$: find optimal fractional solutions $z^{j,k}$ for each possible j, round these solutions, and choose the best. We define O_F, O_I, and O_A as before.

An argument exactly as in Theorem 6.2 shows that, if O_I and O_A are not equal, then

$$\frac{O_I}{O_A} \ge 1 - \frac{(r'-r)(r-y)\ell k}{(T+rk)(T+r'k)} \equiv 1 - G,$$

where $r' = \lceil T/(\ell-k) \rceil$, r is equal to either $\lfloor T/(\ell-k) \rfloor$ or $\lceil T/(\ell-k+1) \rceil$, and $T + rk = \ell r + y$ with $0 < y < r$. Therefore we need to show that the quantity G is at most $1/6$ in all cases.

Note that $T + rk > \ell r$ and $T + r'k > r'k$, so we have

$$G \leq \frac{(r' - r)(r - y)}{rr'}.$$

For both possible values of r, one can show that the integers r', r, and $(r'-r)(r-y)$ satisfy $r' > r > (r' - r)(r - y)$, so G is at most $n/(n+1)(n+2)$ for some nonnegative integer n, and this quantity is always at most $1/6$, as required.

To see that this approximation ratio is best possible, take an instance I of 2Div-Paths and a large value of k; set $T = 3$ and construct an instance of Integer Diverse-Path Resilience as follows. There are nodes a_i, b_i, c_i, d_i, for $i = 1, \ldots, k+1$, as well as s and t. There are arcs of cost 1 from s to each a_i, from c_i to b_{i+1} $(i = 1, \ldots, k)$, and from each d_i to t. There are also arcs of cost 2 from s to b_1 and from c_{k+1} to t. We also take a copy of our instance I of 2Div-Paths for each i with initial nodes a_i and b_i and corresponding terminal nodes d_i and c_i: all arcs involved cost 0. If there are arc-disjoint paths linking each a_i and d_i, and each b_i and c_i, then we can use these to make $k+1$ diverse paths of cost 2, and another path (through all the b_i and c_i), of cost $k+4$. Reserving 2 units on the first $k+1$ paths and 1 unit on the last gives a $(3, k)$-resilient reservation of cost $5k + 8$. If there are no such linking paths, then there is no reservation costing less than $6k + 6$. (We can either reserve capacity 3 on each of the $k + 1$ diverse paths of cost 2 or use the only set of $k + 2$ diverse paths, in which each path has cost 3.) ∎

7. Integer resilience. We next show that the problem of finding a minimum cost integral (T, k)-resilient vector is NP-hard, even in the single-failure case $k = 1$. We couch the problem as a decision problem.

Integer Resilience.

Instance: a digraph D, with integer costs c_{ij} on the arcs, with a single source s and destination t, a demand T (integer), and a target cost C (integer).

Question: is there an integer reservation vector x on the arcs of D such that $c \cdot x \leq C$, and such that x is $(T, 1)$-resilient?

Theorem 7.1. Integer Resilience *is strongly NP-complete.*

Proof. (We omit some of the details: a full proof can be found in [11].) Certainly the problem is in NP, since checking $(T, 1)$-resilience simply involves finding flows of value T in the networks obtained by removing individual edges.

To prove the problem is NP-complete, we give a reduction from 3D-Matching. Recall that an instance of 3D-Matching consists of three sets A, B, C of size n and a collection $\mathcal{T}$ of m "triangles," each containing exactly one element from each of A, B, and C; the question is whether $A \cup B \cup C$ can be written as the disjoint union of n triangles from $\mathcal{T}$.

Suppose that we are given an instance of 3D-Matching as above. We show how to construct an instance of Integer Resilience with $m + 3n + 2$ nodes, $11m + 3n$ arcs, each of cost 1, n, or $2n$, such that there is a $(4m + 3n - 1)$-resilient integer reservation of cost at most $(2n + 1)(4m + 3n) + n$ if and only if the original instance did possess a 3D-matching.

We take one node of D for each triangle $abc \in \mathcal{T}$, one node for each element of $A \cup B \cup C$, and also nodes s and t. We take four parallel arcs, each of cost 1, from s to each node corresponding to an element of $\mathcal{T}$. Each node abc has seven arcs leaving it: four, of cost $2n$, go directly to t and one, of cost n, to each of the constituent elements a, b, and c. Finally, there is a single arc of cost n from each element of $A \cup B \cup C$ to t.

Given a 3D-matching $\mathcal{U}$, we can find a $(T,1)$-resilient reservation, with $T = 4m+3n-1$, by reserving capacity 1 on all arcs entering t and all arcs leaving elements of $\mathcal{U}$; we further reserve capacity 1 on arcs from s to elements of $\mathcal{T}\backslash\mathcal{U}$ and capacity 2 on arcs from s to elements of $\mathcal{U}$. It is easy to check that this reservation is $(T,1)$-resilient and has the required cost $(2n+1)(4m+3n)+n$.

Conversely, if there is a $(T,1)$-resilient reservation with this cost, one may easily check that it must involve reserving capacity 1 on all arcs into t and also on one arc entering each node of $A \cup B \cup C$. Such a set of arcs has total cost $2n(4m+3n)$ and includes between 4 and 7 arcs from each node of $\mathcal{T}$. For $v \in \mathcal{T}$, let $d(v)$ be the number of reserved arcs leaving v.

For $v \in \mathcal{T}$, the reservations on the four arcs between s and v must total at least $d(v)$, and the sum of any three of them must be at least $d(v)-1$ (since after deleting an arc there still exists a T-flow). The minimum cost of such a reservation between s and v is just 4 if $d(v) = 4$ but $d(v)+1$ if $d(v) \in \{5,6,7\}$. Therefore the total cost of a $(T,1)$-resilient reservation consistent with the values $d(v)$ is $2n(4m+3n)+\sum_v d(v)$ plus the number N of elements v for which $d(v) > 4$. We know that $\sum_v d(v) = T+1 = 4m+3n$, and one may check that $N \geq n$, with equality if and only if $d(v) = 7$ for just n elements of $\mathcal{T}$, and $d(v) = 4$ for the remainder. The arcs out of $\mathcal{T}$ can be distributed in such a manner only if those elements v with $d(v) = 7$ constitute a 3D-matching in the original instance. □

On the positive side, there is a simple $(k+1)$-approximate algorithm for Integer Resilience, namely, to find a cheapest set of $k+1$ edge-disjoint paths and reserve capacity T on each arc. The following result states that this is indeed a $(k+1)$-approximate algorithm.

Proposition 7.2. *If x is a (T,k)-resilient reservation vector, then* $\mathrm{cost}(x) \geq \frac{1}{k+1}\mathrm{cost}(z^{k+1,k})$.

Proof. Let x be a minimum cost (T,k)-resilient vector. Define x' by setting, for each arc a, $x'_a = \min((k+1)x_a, T)$; it follows that $x' \geq x$.

Let S be any (s,t)-set. We claim that $x'(\delta^+(S)) \geq (k+1)T$. Suppose first that there is a set K of $k+1$ arcs in $\delta^+(S)$ such that, for $a \in K$, $x_a \geq \frac{T}{k+1}$, and so $x'_a = T$. Then $x'(\delta^+(S)) \geq \sum_{a\in K} x'_a = (k+1)T$.

If instead $x_a < \frac{T}{k+1}$ for all arcs $a \in \delta^+(S)$ except for those in a set K of k arcs. Then $x(\delta^+(S)-K) \geq T$, and so $x'(\delta^+(S)-K) = (k+1)x(\delta^+(S)-K) \geq (k+1)T$. This proves our claim.

Now, since $x'(\delta^+(S)) \geq (k+1)T$ for each (s,t)-set S, there exists an (s,t) flow x'' of value $(k+1)T$ such that $x'' \leq x'$, so in particular no arc has reserved capacity more than T; thus x'' is a (T,k)-resilient vector.

As we remarked earlier, the diverse-path reservation $z^{k+1,k}$ is a minimum cost (s,t) flow of value $(k+1)T$ subject to an upper bound T on the flow through any given arc. It follows that $\mathrm{cost}(z^{k+1,k}) \leq \mathrm{cost}(x'') \leq \mathrm{cost}(x') \leq (k+1)\mathrm{cost}(x)$. □

Thus we have that the minimum cost diverse-path reservation has at most $k+1$ times the cost of an optimal (T,k)-resilient reservation. Of course, this is of greatest interest in the case $k = 1$. The ratio $k+1$ between the two optimal costs is best possible, as can be seen by considering a network D with three nodes s, u, and t, $k+1$ arcs of cost 1 from s to u, and many arcs of large cost c from u to t.

8. Conclusions.

8.1. Future directions. Although General Resilience can be solved in polynomial time for fixed k, we have been unable to find a truly practical algorithm for

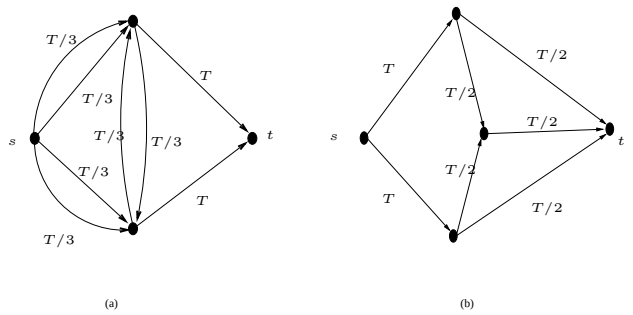


FIG. 8.1. (a) *Vertex of* $\mathcal{R}(T, 1, D)$ *with a cycle;* (b) *basic solution for both arc-failure and node-failure resilience.*

the problem even in the case $k = 1$. It is natural to believe that there might be an algorithm which uses some generalization of cycle augmentation for standard minimum cost flows. In order to explain why it is likely to be difficult to find such an algorithm, we give in Figures 8.1(a) and 8.1(b) two examples of vertices of polyhedra $\mathcal{R}(T, 1, D)$, which of course give the unique optimal solution to instances of GENERAL RESILIENCE. Indeed, the polyhedral structure for general resilience appears to be quite rich; further examples are provided in [11].

There are other versions of resilience questions which we did not investigate. For instance, one may wish to guard against node failures instead of or as well as arc failures: this problem can be formulated using our previous models and then applying standard splitting operations on nodes other than s and t. Figure 8.1(b) also represents a basic solution for such a fractional node-failure resilience problem.

In a further paper [12], we consider the effect of imposing upper bounds on the capacities that can be reserved on each arc.

Finally, another critical concern is how we may recover from failures. In the case of diverse-path reservations, we have the best possible scenario. The node s can be programmed so that if a communication path fails, it simply shunts its traffic onto the remaining paths. For general reservation vectors, the rerouting of traffic may be more complex. Indeed, unless the reservation vector was carefully constructed, traffic on the nonfailed communication paths may need also to be rerouted from scratch. The distinction between whether or not we may disturb nonfailed traffic flows leads to two types (*strong* and *weak*) of resilience problems; these are discussed in some detail in the technical report [11].

8.2. Applications to more than one source-destination pair. We now consider the problem where we are given a collection of node pairs $(s_1, t_1), \ldots, (s_q, t_q)$ as well as a collection of demands T_i, $i = 1, 2, \ldots, q$, and (possibly) a collection of integers $k_1, \ldots, k_q$. Each *commodity* i must reserve capacity in a network D which is (T_i, k_i)-resilient for the source-destination pair (s_i, t_i).

In developing solution techniques for a general multicommodity instance, we feel there is significant computational benefit in insisting that each commodity is handled by diverse-path reservation vectors. This is despite the fact that such reservations may cost more, even in the 1-commodity case (e.g., Figures 8.1(a) and 8.1(b)). This approach replaces the complexity of general resilience constraints with the simpler subproblem of deciding, for each commodity i, the number n_i of diverse paths to be included in the support of its reservation vector. This subproblem may be handled by

some branching scheme and, for any fixed choice of the n_i's, the optimization problem can be formulated as a much simplified multicommodity network design problem.

For instance, suppose that for each arc a we may purchase up to M_a units of capacity, each at cost c_a. We then formulate the diverse-path multicommodity resilience problem, where each commodity i must use a diverse-path reservation on (exactly) n_i paths, as a mixed integer program.

$$\begin{array}{ll}
\min \sum_a c_a y_a, & \\
\sum_{i=1}^{q} \frac{T_i}{n_i - k_i} r_a^i = y_a \le M_a & \text{for each arc } a, \\
r^i(\delta^+(v)) = r^i(\delta^-(v)) & \text{for each } i \text{ and } v \ne s_i, t_i, \\
r^i(\delta^+(s_i)) - r^i(\delta^-(s_i)) = n_i & \text{for each } i, \\
r_a^i \in \{0,1\}, y_a \in \mathbb{Z} & \text{for each } a \text{ and } i.
\end{array}$$

A solution r^i is thus an (s_i, t_i) 0-1 flow vector of value n_i, and the first family of constraints state that the total capacity reserved on an arc a is at most M_a.

Acknowledgments. The first and second authors acknowledge support from DIMACS during extended visits to Bell Labs. The authors are grateful for insightful remarks and encouragement from Gautam Appa, Dan Bienstock, Fan Chung, Michele Conforti, Bharat Doshi, Susan Powell, Paul Seymour, and Mihalis Yannakakis, as well as two anonymous referees.

A major inspiration for this work was Dr. Ewart Lowe, of British Telecom, who tragically died in a diving accident on May 22, 1998, off the coast of Normandy. Ewart introduced the authors to many mathematical problems in telecommunications. He also acted as mentor to the final author during his projects for British Telecom. We dedicate this paper to the memory of his inspiration, generosity, and his unbounded enthusiasm, which are greatly missed by all who knew him.

REFERENCES

[1] A.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows—Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, NJ, 1993.

[2] D. Alevras, M. Grötschel, and R. Wessäly, *Cost-efficient network synthesis from leased lines*, Ann. Oper. Res., 76 (1998), pp. 1–20.

[3] D. Alevras, M. Grötschel, and R. Wessäly, *Capacity and Survivability Models for Telecommunication Networks*, Technical Report 97-24, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Berlin, Germany, 1997.

[4] N. Ascheuer, M. Grötschel, and J. Rambau, *Combinatorial online optimization in practice*, Optima, 57 (1998), pp. 1–6.

[5] A. Balakrishnan, T. Magnanti, J. Sokol, and Y. Wang, *Modeling and Solving the Single Facility Line Restoration Problem*, Working Paper OR 327-98, Operations Research Center, MIT, Cambridge, MA, 1998.

[6] F. Barahona, *Network design using cut inequalities*, SIAM J. Optim., 6 (1996), pp. 823–837.

[7] J.J. Bartholdi, J.B. Orlin, and H.D. Ratliff, *Cyclic scheduling via integer programs with circular ones*, Oper. Res., 28 (1980), pp. 1074–1085.

[8] D. Bienstock, S. Chopra, O. Günlük, and C. Tsai, *Minimum cost capacity installation for multicommodity network flows*, Math. Programming, 81 (1998), pp. 177–199.

[9] D. Bienstock and O. Günlük, *Capacitated network design—polyhedral structure and computation*, INFORMS J. Comput., 8 (1996), pp. 243–260.

[10] D. Bienstock and G. Muratore, *Strong inequalities for capacitated survivable network design problems*, Math. Program., 89 (2000), pp. 127–147.

[11] G. Brightwell, G. Oriolo, and F.B. Shepherd, *Some Strategies for Reserving Resilient Capacity*, LSE CDAM Report 98-04, London School of Economics, London, UK, 1998.

[12] G. Brightwell, G. Oriolo, and F.B. Shepherd, *Reserving Resilient Capacity with Upper Bound Constraints*, LSE CDAM Report 2000-03, London School of Economics, London, UK, 2000.

[13] G. Brightwell and F.B. Shepherd, *Consultancy Report: Resilience Strategy for a Single Source-Destination Pair*, LSE CDAM Report 96-22, London School of Economics, London, UK, 1996.

[14] S.J. Bye and M. Herzberg, *An optimal spare-capacity assignment model for survivable networks with hop limits*, in Proceedings of GLOBECOM, San Francisco, 1994, pp. 1601–1606.

[15] S.J. Bye and M. Herzberg, *Spare-capacity assignment in survivable networks for multi-link and node failures with hop limits*, in Proceedings of Networks'94, 1994, pp. 381–386.

[16] S. Fortune, J. Hopcroft, and J. Wyllie, *The directed subgraph homeomorphism problem*, Theoret. Comput. Sci., 10 (1980), pp. 111–121.

[17] A. Frank, *Connectivity and network flows*, in Handbook of Combinatorics, Vol. 1, R.L. Graham, M. Grötschel, and L. Lovász, eds., North-Holland, Amsterdam, 1995, pp. 111–177.

[18] M. Grötschel, C.L. Monma, and M. Stoer, *Design of survivable networks*, in Network Models, Handbooks in Operations Research and Management Science, North-Holland, Amsterdam, 1995, pp. 617–672.

[19] O. Günlük, *Branch-and-cut algorithm for capacitated network design problems*, Math. Programming, 86 (1999), pp. 17–39.

[20] R.R. Irashko, M.H. MacGregor, and W.D. Grover, *Optimal capacity placement for path restoration in mesh survivable networks*, IEEE International Conference on Communications, Dallas, 1996, pp. 1568–1574.

[21] T. Magnanti, P. Mirchandani, and R. Vachani, *Modelling and Solving the Capacitated Network Loading Problem*, MIT OR Working Paper, OR 239-91, MIT, Cambridge, MA, 1991.

[22] T. Magnanti, P. Mirchandani, and R. Vachani, *The convex hull of two core capacitated network design problems*, Math. Programming, 60 (1993), pp. 233–250.

[23] T. Magnanti, P. Mirchandani, and R. Vachani, *Modelling and solving the two facility capacitated network loading problem*, Oper. Res., 43 (1995), pp. 142–157.

[24] T. Magnanti and Y. Wang, *Polyhedral Properties of the Network Restoration Problem—With the Convex Hull of a Special Case*, Working Paper OR 323-97, Operations Research Center, MIT, Cambridge, MA, 1997.

[25] H. Sakauchi, Y. Nishimura, and S. Hasegawa, *A self-healing network with an economical spare-channel assignment*, in Proceedings of GLOBECOM, San Diego, CA, 1990, pp. 438–443.

[26] I. Saniee, *Optimal routing designs in self-healing communications networks*, Int. Trans. Oper. Res., 3 (1996), pp. 187–195.

[27] B.D. Venables, W.D. Grover, and M.H. MacGregor, *Two strategies for spare capacity placement in mesh restorable networks*, IEEE International Conference on Communications, Geneva, 1993, pp. 267–271.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 540–548

WHEN IS INDIVIDUAL TESTING OPTIMAL FOR NONADAPTIVE GROUP TESTING?*

S. H. HUANG† AND F. K. HWANG†

Abstract. The combinatorial group testing problem is, assuming the existence of up to d defectives among n items, to identify the defectives by as few tests as possible. In this paper, we study the problem for what values of n, given d, individual testing is optimal in nonadaptive group testing. Let $N(d)$ denote the largest n for fixed d for which individual testing is optimal. We will show that $N(d) = (d+1)^2$ under a prevalent constraint in practical nonadaptive algorithms and prove that $N(d) = (d+1)^2$ for $d = 1, 2, 3, 4$ without any constraint.

Key words. nonadaptive group testing, disjunct matrix, union-free matrix

AMS subject classifications. 05D05, 05B20

PII. S0895480199359247

1. Introduction.

In combinatorial group testing, a prototype problem called the $(\bar{d}, n)$ problem is to assume that there are up to d defectives among n given items, and the problem is to separate the good items from the defective ones by group tests. A (group) test is administered on an arbitrary subset S of the items with two possible outcomes; a *negative* outcome means S contains no defectives and a *positive* outcome means S contains at least one defective, not knowing exactly how many or which ones. A group testing algorithm is *optimal* if it minimizes the worst-case number of tests required.

A group testing algorithm is *sequential* if the tests can be done sequentially and the outcomes of previous tests are known at the time to determine the current test. A group testing algorithm is *nonadaptive* if all tests must be specified at once. A nonadaptive algorithm can be represented by a 0-1 matrix where columns are items, rows are tests, and a 1-entry in cell (i, j) means item j is contained in test i. Note that a column can be viewed as a subset whose elements are indices of the rows incident to the column. Thus we can talk about the union of columns. Group testing has applications to blood testing, electrical and chemical testing, coding, multiaccess channel conflict resolution, etc. Recently, nonadaptive group testing has been shown to play a crucial role in the clone library screening problem.

Kautz and Singleton [8] introduced the notions of "$\bar{d}$-*separable*" and "*d-disjunct*" of 0-1 matrix $M_{t\times n}$; the former requires that no two unions of up to d columns are identical, while the latter requires that no union of d columns contains a column not in the union. They showed that both properties guarantee $M_{t\times n}$ to be a nonadaptive $(\bar{d}, n)$ algorithm, while the d-disjunct property has an extra feature of simplifying the process of identifying defectives. These two properties were also called r-union-free and r-cover-free [3, 4] in extremal set theory.

A trivial d-disjunct algorithm not using the idea of group testing would test the n items individually, which requires n tests. Thus it is of interest to know for what values of n, given d, individual testing is optimal.

*Received by the editors July 6, 1999; accepted for publication (in revised form) June 7, 2001; published electronically October 23, 2001.

http://www.siam.org/journals/sidma/14-4/35924.html

†Department of Applied Mathematics, National Chiao-Tung University, Hsinchu 30050, Taiwan, Republic of China (u8622522@math.nctu.edu.tw, fhwang@math.nctu.edu.tw). This research was partially supported by Republic of China National Science Council grant NSC 90-2115-M-009-007.

A similar question as posed in the title has been asked on sequential group testing for exactly d defectives. (Thus the state of the last item can be deduced without testing.) Hu, Hwang, and Wang [5] conjectured that individual testing is optimal if and only if $n \leq 3d$. They proved the "necessary" part, but the sufficient condition is proved only for $n \leq 2.5d$, improving an earlier sufficient condition $n \leq 2d$ of Hwang [7]. Du and Hwang [1] further improved the sufficient condition to $n \leq 2.625d$, but the conjecture remains open.

Back to the nonadaptive case, let $N(d)$ denote the largest n for fixed d for which individual testing is optimal. Bassalygo (see [2]) first gave a lower bound.

LEMMA 1.1. $N(d) \geq \binom{d+2}{2}$.

Erdös, Frankl, and Füredi [3] conjectured that

$$\begin{aligned} \lim \quad & N(d)/d^2 = 1 \quad \text{(weaker version)}, \\ & N(d) \leq (d+1)^2 \quad \text{(stronger version)} \end{aligned}$$

and stated without giving details that they can prove the stronger version for $d \leq 3$. In this paper, we will prove $N(d) \leq (d+1)^2$ for $d+1$ a prime power. Thus

$$\binom{d+2}{2} \leq N(d) \leq (d+1)^2.$$

This establishes $N(d) = O(d^2)$, as opposed to $N(d) = O(d)$ in the sequential case. We will also show that under a prevalent constraint in practical nonadaptive algorithms, $N(d) = (d+1)^2$. Finally, we prove $N(d) = (d+1)^2$ for d=1, 2, 3, 4 without any constraint.

2. A necessary condition. Let *t(d, n)* denote the minimum number of tests a nonadaptive algorithm requires, given d and n. We first make an observation.

LEMMA 2.1. *The existence of a d-disjunct matrix $M_{t\times n}$ with $t < n$ implies $N(d) \leq t$.*

Proof. *t(d, n)* is clearly nondecreasing in n. Hence, the existence of a d-disjunct $M_{t\times n}$ with $n > t$ implies $t+1$ items can be done in t tests, or, equivalently, $N(d) \leq t$. □

Let $\lambda_{cc'}$ denote the inner product of two columns c and c'. Define $\bar{\lambda} = \max\lambda_{cc'}$ over all pairs of columns. A 0-1 matrix is called a *weight-w matrix* if each column is a w-set. The following lemma is well known [10].

LEMMA 2.2. *A weight-w matrix is $(\lceil w/\bar{\lambda}\rceil - 1)$-disjunct.*

COROLLARY 2.3. *If $\bar{\lambda} = 1$, then a weight-w matrix is $(w-1)$-disjunct.*

The main result in this section is the following theorem.

THEOREM 2.4. *$N(d) \leq (d+1)^2$ for $d+1$, a prime power.*

Proof. By Lemma 2.1 and Corollary 2.3, it suffices to construct a weight-$(d+1)$ matrix $M_{(d+1)^2\times n}$ with $\bar{\lambda} = 1$ and $(d+1)^2 < n$.

It is well known [9] that if $d+1$ is a prime power, then there exists a set of d mutually orthogonal latin squares (MOLS). Each such square will generate $d+1$ columns (as $(d+1)$-subsets of $\{0, 1, 2, \ldots, (d+1)^2 - 1\}$) of M, one for each set of cells (i, j) having the same entry k, $0 \leq k \leq d$, and cell (i, j) is translated to the number $i(d+1) + j$. Clearly, columns generated from the same square have $\lambda_{cc'} = 0$. Due to orthogonality, columns generated from different latin squares satisfy $\lambda_{cc'} = 1$. Thus the d MOLS generate a total of $(d+1)d$ columns of M with $\bar{\lambda} = 1$. Finally, consider the $(d+1) \times (d+1)$ matrix S, where the entry in cell (i, j) is just (i, j). Clearly, the columns (rows) of S have $\lambda_{cc'} = 0$, and each row-column pair has $\lambda_{rc} = 1$.

Furthermore, the set of cells having the same entry from a latin square must be a *transversal*; i.e., they have distinct row indices and distinct column indices. Let s be a transversal, r a row, and c a column of S. Then $\lambda_{sr} = \lambda_{sc} = 1$. Thus we can add the $2(d+1)$ rows and columns of S to be columns of M and preserve the $\bar{\lambda} = 1$ property. The total number of columns in M is now

$$(d+1)d + 2(d+1) = (d+1)(d+2).$$

Note that the base set of the columns is the set $\{0, 1, \ldots, (d+1)^2 - 1\}$. By treating the base set as the set of tests, then we have constructed a $(d+1)^2 \times (d+1)(d+2)$ matrix with $\bar{\lambda} = 1$. □

3. A necessary and sufficient condition under $\bar{\lambda} = 1$. Constructing efficient d-disjunct matrices is a difficult task, with the simplest and most prevalent method being given by Corollary 2.3, i.e., constructing matrices with $\bar{\lambda} = 1$. In this section, we study the problem given in the title of this paper under the constraint $\bar{\lambda} = 1$.

Let $M_{t\times n}$ be a 0-1 matrix. Let *G(M)* denote the graph with the rows of M as vertices and an edge between two vertices if and only if the inner product of the two corresponding rows (viewed as subsets of $\{1, 2, \ldots, n\}$) is not zero. Let $d_G(v)$ denote the degree of v in G.

LEMMA 3.1. *Suppose M is weight-$(d+1)$ and $\bar{\lambda} = 1$. Then G(M) consists of n edge-disjoint K_{d+1} (complete graph of order $d+1$).*

Proof. Each column generates a K_{d+1}. The $\bar{\lambda} = 1$ property forces the K_{d+1}'s to be edge disjoint. □

Recently, W. T. Huang and Hwang observed (private communication) that a previous result of Weideman and Raghavarao [12] for the $\bar{d}$-separable matrix can be extended to the following lemma.

LEMMA 3.2. *Any d-disjunct matrix with $\bar{\lambda} = 1$ can be reduced to one where no column weight exceeds $d+1$ with the d-disjunct property preserved.*

Let $n(d, t)$ denote the largest n such that there exists a d-disjunct $M_{t\times n}$. A column in M is called *isolated* if there exists a row incident to this column only.

It is easily observed in the following lemma.

LEMMA 3.3. *Let $M_{t\times n}$ be a d-disjunct matrix containing an isolated column. Then*

$$n \le 1 + n(d, t-1).$$

Proof. Deleting the isolated column and its incident row does not affect the d-disjunct property. □

Note that to determine $N(d)$, we need to find an $M_{t\times n}$ satisfying $t < n$ and minimizing n. A matrix with no nonisolated column always satisfies $t \ge n$, and hence cannot be such a candidate, and consequently is of no interest. Therefore, we assume from now on that we consider only matrices with an isolated column.

Dyachkov and Rykov [2] proved the following lemma.

LEMMA 3.4. *Let M be a d-disjunct matrix. Then a column with weight at most d must be isolated.*

LEMMA 3.5. *Let M be a d-disjunct matrix with $\bar{\lambda} = 1$ and no column weight less than $d+1$. Then $n \le \lfloor \frac{t(t-1-r)}{(d+1)d} \rfloor$, where $t - 1 \equiv r \pmod{d}$.*

Proof. By Lemmas 3.2 and 3.4, we may obtain a constant weight-w matrix M^* with $w = d+1$ from M. Consider $G(M^*)$. Each vertex has a maximum possible

degree of $t-1$. By Lemma 3.1, $d \mid d_G(v)$. Hence, $d_G(v) \le (t-1)-r$. Thus the total number of edges in G satisfies

$$n \times \binom{d+1}{2} = \frac{\Sigma d_G(v)}{2} \le \frac{t[(t-1)-r]}{2},$$

$$\text{or } n \le \lfloor \frac{(t(t-1-r)}{(d+1)d} \rfloor. \qquad \square$$

We are now ready to prove the main result of this section.

THEOREM 3.6. *Under* $\bar{\lambda} = 1, N(d) = (d+1)^2$ *for* $d+1$, *a prime power.*

Proof. By Theorem 2.4, it suffices to prove $N(d) \ge (d+1)^2$, which is done by proving the nonexistence of a d-disjunct matrix $M_{[(d+1)^2-1]\times(d+1)^2}$ with $\bar{\lambda} = 1$.

Let $M_{t\times n}$ be a d-disjunct matrix with $\bar{\lambda} = 1$ and $t = (d+1)^2 - 1$. By Lemma 3.2, we may assume that every column of M has weight at most $d+1$.

Case (i). No column of M has weight less than $d+1$. Since $t-1 = (d+1)^2 - 2 = (d+1)d + d - 1$, $t-1-r = (d+1)d$. By Lemma 3.5,

$$n \le \frac{t(d+1)d}{d(d+1)} = t.$$

Case (ii). There exists a column of M with weight at most d. Let C be the set of columns with weight at most d. By Lemma 3.4, each column $c \in C$ is isolated; i.e., there exists a row $r(c)$ incident only to c. Let M' be obtained from M by deleting c and $\{r(c) : c \in C\}$. Then M' is a $(t- \mid C \mid) \times (n- \mid C \mid)$ weight-$(d+1)$ matrix with $\bar{\lambda} = 1$, since

$$t- \mid C \mid -1 = d(d+1) + d - 1 - \mid C \mid, \qquad t-1- \mid C \mid -r \le d(d+1).$$

By Lemma 3.5,

$$(n- \mid C \mid) \le \left\lfloor \frac{(t- \mid C \mid)(d+1)d}{d(d+1)} \right\rfloor,$$

which again leads to $n \le t$. $\square$

4. $N(d)$ for small d. Bassalygo (see [2]) proved the following lemma.

LEMMA 4.1. *Let M be a d-disjunct matrix and c a column of M with weight w. Then $t(d,n) \ge w + t(d-1, n-1)$.*

Spencer [11] proved the following lemma

LEMMA 4.2. $n(1,t) = \binom{t}{\lfloor \frac{t}{2} \rfloor}$.

THEOREM 4.3. $N(1) = 4$.

Proof. By Theorem 2.4, $N(1) \le 4$. Since $n(1,3) = 3$, $N(1) \ge 4$. Hence, $N(1) = 4$. $\square$

LEMMA 4.4. $n(2,7) = 7$.

Proof. Let $M_{7\times n}$ be a 2-disjunct matrix. If $\bar{\lambda} = 1$, then Lemma 4.4 follows from Theorem 3.6. Therefore, we may assume the existence of two columns, c and c', with $\lambda_{cc'} > 1$. Then c is either isolated or has weight at least 4.

(i) c is isolated. By Lemmas 1.1 and 3.3,

$$n \le 1 + n(2,6) = 1 + 6 = 7.$$

(ii) c has weight at least 4. Deleting c and its incident rows, the reduced matrix M' can have at most three rows and is 1-disjunct. By Lemma 4.2, M' can have at most three columns. Then $n \leq 4$. □

THEOREM 4.5. $N(2) = 9$.

Proof. Since $d + 1 = 3$ is a prime power, by Theorem 2.4, $N(2) \leq 9$. Therefore, it suffices to prove $N(2) \geq 9$, or $n(2, 8) = 8$.

Let $M_{8\times n}$ be 2-disjunct. By Theorem 3.6, we may assume the existence of two columns, c and c', with $\lambda_{cc'} > 1$.

(i) c is isolated. Then by Lemmas 3.3 and 4.4

$$n \leq 1 + n(2, 7) = 8.$$

(ii) c has weight at least 4. Then by Lemmas 4.1 and 4.2

$$t(2, 8) \geq 4 + t(1, 7), \qquad \text{or} \qquad n(2, 8) \leq 7. \qquad \square$$

COROLLARY 4.6. $N(3) \geq 13$.

Proof. By Lemmas 3.4 and 4.1, $t \geq (d + 1) + t(d - 1, n - 1)$ for a d-disjunct matrix $M_{t\times n}$ (with at least one nonisolated column), since $n(2, 8) = 8$, $t(2, 12) \geq 9$. Therefore, $t \geq 4 + t(2, n - 1) \geq 4 + 9 = 13$ for $n \geq 13$. It implies that $t(3, n) = n$ for $n \leq 13$, i.e., $N(3) \geq 13$. □

Let $I(c)$ be a collection of different b-subsets, $b \geq 2$, of $\{1, 2, 3, \ldots, t\}$ denoting the intersection property of column c with other columns. For example, $I(c) = \{(1, 2), (1, 3, 4)\}$ means that there exists at least one column c_1 intersecting c at rows 1 and 2, and there exists at least one column c_2 intersecting c at rows 1, 3, 4.

Let M_1 be a $t \times n_1$ weight-4 2-disjunct matrix with no isolated column. Then $\bar{\lambda} \leq 2$. We will do some deletions on M_1 to reduce weight 4 to weight 3 such that the 2-disjunct property is still preserved in $(M_1)^i, i \geq 0$, which is the reduced matrix after the ith deletion. Note that deleting a 1-entry of c will affect other $I(c')$ in general. Therefore, after each deletion, we need to reconsider the $I(c)$ of the reduced matrix, where c has weight 4.

Consider $I(c_j), j \leq n_1$, of $(M_1)^i, i \geq 0$, and $I(c_j) \neq \emptyset$, where c_j has weight 4. The deletion rule is as follows:

(1) If $I(c_j) \subseteq \{(x_j, y_j), (x_j, z_j), (x_j, v_j); x_j \neq y_j \neq z_j \neq v_j \in \{1, 2, 3, \ldots, t\}\}$ in $(M_1)^i$, then delete the 1-entry in row x_j of c_j; hence the reduced column has weight 3 and has an inner product at most one with any other column of $(M_1)^i$. $(M_1)^i$ remains 2-disjunct by Lemma 3.2.

(2) If $I(c_j) = \{(x_j, y_j), (x_j, z_j), (y_j, z_j); x_j \neq y_j \neq z_j \in \{1, 2, 3, \ldots, t\}\}$ in $(M_1)^i$, then do nothing at this moment until the last step.

Finally, we will get a reduced matrix $(M_1)^f$ with no case (1) after f deletions. If case (2) does not occur, we are done. If case (2) occurs, then there exists a $t \times n_1'$ submatrix M_1' contained in $(M_1)^f$ which has the following four properties:

(1) M_1' has at least four columns and each column of M_1' has weight 4.

(2) $\forall c_j \in C(M_1'), I(c_j)$ has the form $\{(x_j, y_j), (x_j, z_j), (y_j, z_j); x_j \neq y_j \neq z_j \in \{1, 2, 3, \ldots, t\}\}$.

(3) $c_j \in M_1'$ and $|c_i \bigcap c_j| = 2 \Longrightarrow c_i \in M_1'$.

(4) M_1' doesn't contain a submatrix with fewer columns having properties (1), (2), and (3).

Let M_1'' be a $q \times n_1'$ submatrix of M_1', where the q rows of M_1'' are the collection of rows $\{x_i, y_i, z_i : c_i \in C(M'),\ I(c_i) = \{(x_i, y_i), (x_i, z_i), (y_i, z_i)\}\}$. Then we have the following lemma.

LEMMA 4.7. $n_1' \geq q$.

Proof. Suppose $c \in M_1'$ and $I(c) = \{(x,y),(x,z),(y,z)\}$. Then each row x, y, z must have at least three 1-entries in M_1''. For example, x appears once in c, once in a column intersecting c at (x,y), and once in a column intersecting c at (x,z). Suppose c has a fourth 1-entry v also in M_1''. Then $v \in \{x',y',z'\}$ for some column c' with $I(c') = \{(x',y'),(x',z'),(y',z')\}$. Therefore, row v has at least four 1-entries. Let k be the number of columns with four 1-entries in M''. Then counting the number of 1-entries by column and by row separately, we have $3n_1' + k \geq 3q + k$, or $n_1' \geq q$. ∎

LEMMA 4.8. *Suppose* $c = \{x,y,z,v\}$ *and* $I(c) = \{(x,y),(x,z),(y,z)\}$ $\forall c \in C(M_1')$. *Then* $c' \bigcap \{x,y,z\} \neq \emptyset$ *implies* $v \notin c'$ *in a 2-disjunct matrix.*

Proof. Without loss of generality, assume that c' contains x. Suppose to the contrary that c' also contains v. By the definition of $I(c)$, there exists a column c'' containing (y,z). Then $c' \bigcup c''$ contains c, contradicting the 2-disjunct. ∎

LEMMA 4.9. $n(2,9) = 12$.

Proof. The proof uses the Steiner triple system with $v = 9$ and $b = 12$ [9]. ∎

LEMMA 4.10. $n(2,10) \leq 13$.

Proof. Let $M_{10\times n}$ be a 2-disjunct matrix. If M has an isolated column, then by Lemmas 3.3 and 4.9, $n \leq 1 + n(2,9) = 1+12 = 13$. If $\bar{\lambda} = 1$ and there exists no isolated column in M, then $r \equiv 10 - 1 \equiv 1 \pmod 2$. By Lemma 3.5, $n \leq \lfloor \frac{10(10-1-1)}{3\cdot 2} \rfloor = 13$. Therefore, we may assume M has no isolated column and $\bar{\lambda} > 1$. Let column c have maximum weight of M. Then c has weight at least 4.

(i) c has weight greater than 4. Then by Lemmas 4.1 and 4.2,

$$t(2,14) \geq 5 + t(1,13) = 11, \qquad \text{or} \qquad n(2,10) \leq 13.$$

(ii) c has weight 4. Then M is 2-disjunct with $\bar{\lambda} = 2$, and each column has weight 3 or 4. Let $M = M_1 \bigcup M_2$, where M_1 is a $10 \times n_1$ weight-4 matrix, and M_2 is a $10 \times (n - n_1)$ weight-3 matrix.

We first make an observation. Because M is 2-disjunct with $\bar{\lambda} = 2$ and has maximum weight 4, it forces both M_1, M_2 to be 2-disjunct with M_1 having $\bar{\lambda} = 2, M_2$ having $\bar{\lambda} = 1$, and $\forall c_1 \in M_1, c_2 \in M_2, \lambda_{c_1 c_2} \leq 1$.

Next we want to claim that M_1 with $\bar{\lambda} = 2$ can be reduced to a matrix with $\bar{\lambda} = 1$ by deleting some 1-entries and remains 2-disjunct.

By the method we used previously, we get matrix $(M_1)^f, M_1', M_1''$ from $(M_1)^f$. If M_1', hence M_1'', does not exist, then we are done. Otherwise, by relabeling the rows, we may assume that $c_1 = \{1,2,3,4\}$. The completion of $I(c_1) = \{(1,2),(1,3),(2,3)\}$ implies the following submatrices must be in M_1' (see Figure 1):

Note that the completion of $I(c_2), I(c_3), I(c_4)$ requires at least one more row. In fact, if $q = 4$, then (b) shows the only way to complete $I(c_2), I(c_3)$, and $I(c_4)$. This case can be taken care of by deleting the circled 1's. It is easily verified that any two columns intersect at most once after the deletion. Hence, $\bar{\lambda} = 1$ in M_1'. In other words, M can be reduced to a matrix which remains 2-disjunct with $\bar{\lambda} = 1$ and has column weight 3 or 4. Hence, by Lemma 3.5, $t - 1 - r = 8$, $n \leq \lfloor \frac{10\times 8}{3\times 2} \rfloor = 13$.

If $q \geq 5$, then there are other ways to complete $I(c_2), I(c_3)$, and $I(c_4)$. By Lemma 4.7, $n_1' \geq q \geq 5$. Let c_5 be a new column in M_1''. Since M_1'' is minimal, $I(c_5)$ must intersect $\{(1,2),(1,3),(2,3)\}$. Without loss of generality, assume $(1,2) \in I(c_5)$.

Since c_5 intersects all c_1, c_2, c_3, and c_4, by Lemma 4.8, c_5 cannot intersect the fourth 1-entries of these columns. Therefore, the other two 1-entries of c_5 must take up new rows, say, rows 8 and 9 (see Figure 2(a)).

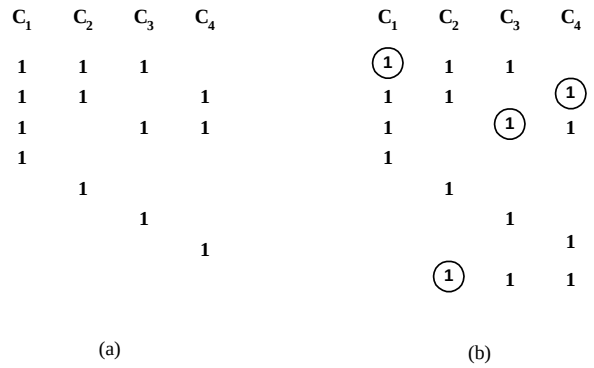


FIG. 1. *Forced submatrices in* M_1'.

c_1	c_2	c_3	c_4	c_5
1	1	1		1
1	1		1	1
1		1	1	
1				
	1			
		1		
			1	
				1
				1

(a)

c_1	c_2	c_3	c_4	c_5	c_6
1	1	1		1	1
1	1		1	1	
1		1	1		
1					
	1				
		1			
			1		1
				1	1
				1	

(b)

FIG. 2. *Larger forced submatrices.*

Inspecting the relation between c_5 and the other c_i, we notice its first two 1-entries are symmetric, and so are its last two. Therefore, we may assume $(1, 8) \in I(c_5)$. To complete $I(c_5)$, there must exist c_6 containing rows 1 and 8. Since c_6 intersects all other columns except c_4, among the existing rows, it can contain only row 7. Therefore, the fourth 1-entry of c_6 must be a new row, say, row 10. However, we still need at least one more row to complete $I(c_2), I(c_3)$, and $I(c_4)$. Hence, the total number of rows exceed 10. □

LEMMA 4.11. $n(3, 13) = 13$.

Proof. Let $M_{13\times n}$ be a 3-disjunct matrix. If $\bar{\lambda} = 1$, then Lemma 4.11 follows from Theorem 3.6. Therefore, we may assume the existence of two columns c and c' with $\lambda_{cc'} > 1$. Then c is either isolated or has weight at least 5.

(i) c is isolated. By Lemma 3.3 and Corollary 4.6,

$$n \le 1 + n(3, 12) = 1 + 12 = 13.$$

(ii) c has weight at least 5. Deleting c and its incident rows, the reduced matrix M' can have at most 8 rows and is 2-disjunct. By $n(2, 8) = 8$, then $n \le 9$. □

LEMMA 4.12. $n(3, 14) = 14$.

Proof. Let $M_{14\times n}$ be a 3-disjunct matrix. If $\bar{\lambda} = 1$, then Lemma 4.12 follows from Theorem 3.6. Therefore, we may assume the existence of two columns c and c' with $\lambda_{cc'} > 1$. Then c either is isolated or has weight at least 5.

(i) c is isolated. By Lemmas 3.3 and 4.11,

$$n \le 1 + n(3, 13) = 1 + 13 = 14.$$

(ii) c has weight at least 5. Deleting c and the incident rows, the reduced matrix M' can have at most 9 rows and is 2-disjunct. By Lemma 4.9,

$$n \le 1 + n(2, 9) = 13. \qquad \square$$

THEOREM 4.13. $N(3) = 16$.

Proof. Since $d + 1 = 4$ is a prime power, by Theorem 2.4, $N(3) \le 16$. Therefore, it suffices to prove $N(3) \ge 16$ or $n(3, 15) = 15$. Let $M_{15\times n}$ be a 3-disjunct matrix. By Theorem 3.6, we may assume the existence of two columns c and c' with $\lambda_{cc'} > 1$.

(i) c is isolated. Then by Lemmas 3.3 and 4.12

$$n \le 1 + n(3, 14) = 15.$$

(ii) c has weight at least 5. Then by Lemmas 4.1 and 4.10

$$n \le 1 + n(2, 10) \le 1 + 13 = 14. \qquad \square$$

With similar but slightly more complicated arguments, we can also prove $N(4) = 25$ [6].

5. Conclusion. In this paper, we studied the problem of when individual testing is optimal for nonadaptive group testing. We showed that $N(d) \le (d + 1)^2$ is a necessary condition by constructing the d-disjunct matrix $M_{[(d+1)^2-1]\times(d+1)^2}$, where $d + 1$ is a prime power. Besides, we showed that $N(d) \ge (d + 1)^2$ is a sufficient condition under $\bar{\lambda} = 1$. Hence, under $\bar{\lambda} = 1$ and $d+1$ a prime power, $N(d) = (d+1)^2$ is a necessary and sufficient condition of the problem we studied. We also prove that $N(d) = (d+1)^2$ for $d = 1, 2, 3, 4$ without any constraint, giving further support to the conjecture $N(d) = (d+1)^2$ for $d+1$, a prime power. However, for $d+1$, not a prime power, we still know little about $N(d)$.

REFERENCES

[1] D. Z. DU AND F. K. HWANG, *Minimizing a combinatorial function*, SIAM J. Algebraic Discrete Methods, 3 (1982), pp. 523–528.

[2] A. G. DYACHKOV AND V. V. RYKOV, *A survey of superimposed code theory*, Problems Control Inform. Theory, 12 (1983), pp. 229–242 (in Russian).

[3] P. ERDÖS, P. FRANKL, AND D. FÜREDI, *Families of finite sets in which no set is covered by the union of r others*, Israel J. Math., 51 (1985), pp. 79–89.

[4] P. FRANKL AND D. FÜREDI, *Union-free hypergraphs and probability theory*, European J. Combin., 5 (1984), pp. 127–131.

[5] M. C. HU, F. K. HWANG, AND J. K. WANG, *A boundary problem for group testing*, SIAM J. Algebraic Discrete Methods, 2 (1981), pp. 81–87.

[6] S. H. HUANG, *When is Individual Testing Optimal for Nonadaptive Group Testing?* Master thesis, Department of Applied Mathematics, National Chiao-Tung University, Hsinchu, Taiwan, 1999.

[7] F. K. HWANG, *A minimax procedure on group testing problems*, Tamkang J. Math., 2 (1971), pp. 39–44.

[8] W. H. KAUTZ AND R. R. SINGLETON, *Nonrandom binary superimposed codes*, IEEE Trans. Inform. Theory, 10 (1964), pp. 363–377.

[9] C. C. LINDNER AND C. A. RODGER, *Design Theory*, CRC Press, New York, 1997.

[10] Q. A. Nguyen and T. Zeisel, *Bounds on constant weight binary superimposed codes*, Problems Control Inform. Theory, 17 (1988), pp. 223–230.
[11] J. Spencer, *Minimal completely separating systems*, J. Combin. Theory, 8 (1970), pp. 446–447.
[12] C. A. Weideman and D. Raghavarao, *Nonadaptive hypergeometric group testing designs for identifying at most two defectives*, Commun. Statist. Theory Methods, 16 (1987), pp. 2997–3006.

SIAM J. DISCRETE MATH.
Vol. 14, No. 4, pp. 549–564

ASYMPTOTIC MINIMUM COVERING RADIUS OF BLOCK CODES*

PO-NING CHEN† AND YUNGHSIANG S. HAN‡

Abstract. In this paper, we restudy the covering radius of block codes from an information theoretic point of view by ignoring the combinatorial formulation of the problem. In the new setting, the formula of the statistically defined *minimum covering radius*, for which the probability mass of uncovered space by M spheres can be made arbitrarily small, is reduced to a minimization of a statistically defined spectrum formula among codeword-selecting distributions. The advantage of the new view is that no assumptions need to be made on the code alphabet (such as finite, countable, etc.) and the distance measure (such as additive, symmetric, bounded, etc.) in the problem transformation, and hence the spectrum formula can be applied in most general situations. We next address a sufficient condition under which uniform codeword-selecting distribution minimizes the spectrum formula. With the condition, the asymptotic minimum covering radius for *block codes under J-ary quantized channels* and *constant weight codes under Hamming distance measure* are determined to display the usage of the spectrum formula.

Key words. covering radius, block codes, information spectrum

AMS subject classifications. 94B65, 94A24

PII. S0895480100379993

1. Introduction.

We first introduce the notations used in this paper. We denote the n-tuple alphabet by $\mathcal{X}^n = \mathcal{X} \times \mathcal{X} \times \cdots \times \mathcal{X}$. For any two elements $x^n = (x_0, x_1, \ldots, x_{n-1})$ and $y^n = (y_0, y_1, \ldots, y_{n-1})$ in $\mathcal{X}^n$, we use $\mu_n(x^n, y^n)$ to denote the n-fold measure[1] on the "distance" between them. In our study, the codewords are drawn from a pregiven codeword set $\mathcal{S}_n$, which can be either the entire space $\mathcal{X}^n$ or its proper subset. Such a generalization will be useful for some specific applications, such as constant weight codes, where the codewords are drawn from a subset (of $\mathcal{X}^n$) containing only words of fixed weight.

Based on the above notations, the problem on the minimum covering radius becomes the following: for given M and $\mathcal{S}_n$, determine the minimum radius $\rho(M, \mathcal{S}_n)$ for which M spheres that center at M selected elements from $\mathcal{S}_n$ jointly cover the entire space $\mathcal{X}^n$. Specifically,

$$\rho(M, \mathcal{S}_n) \stackrel{\triangle}{=} \min_{\substack{\mathcal{C} \subset \mathcal{S}_n \\ |\mathcal{C}| = M}} \max_{x^n \in \mathcal{X}^n} \min_{y^n \in \mathcal{C}} \mu_n(x^n, y^n), \tag{1}$$

*Received by the editors October 23, 2000; accepted for publication (in revised form) August 16, 2001; published electronically October 23, 2001.

http://www.siam.org/journals/sidma/14-4/37999.html

†Department of Communications Engineering, National Chiao Tung University, Hsin Chu, Taiwan 30050, ROC (poning@cc.nctu.edu.tw). The work of this author was supported by the National Science Council of Taiwan under project code NSC 89-2213-E-009-106.

‡Department of Computer Science and Information Engineering, National Chi Nan University, Nan Tou, Taiwan 545, ROC (yshan@csie.ncnu.edu.tw). The work of this author was supported by the National Science Council of Taiwan under project code NSC 89-2213-E-260-002.

[1]Conventionally, a *distance* [16, p. 139] should satisfy the properties of (i) nonnegativity, (ii) being zero iff two points coincide, (iii) symmetry, and (iv) triangle inequality. The spectrum formula derived in this paper, however, is applicable to any measurable function defined over the alphabets. Since none of the above four properties are assumed, the measurable function on the "distance" between two code letters is therefore termed *generalized distance* function. For simplicity, we will abbreviate the *generalized distance* function simply as the *distance* function in the remaining part of the paper.

where $|\mathcal{C}|$ denotes the size of the set $\mathcal{C}$. Here we do not assume that the M elements drawn from $\mathcal{S}_n$ must be distinct. In other words, one can choose M identical elements from $\mathcal{S}_n$ as long as the resultant codebook gives the minimum covering radius. In addition, we implicitly assume that $|\mathcal{S}_n| > 0$.

The problem of determining the covering radius has been studied by many researchers [2], [5], [6], [7], [8], [9], [10], [11], [12], [14], [15], [17], [18], [19], [20], [21] among which [2], [5], [6], [11], [17], and [19] focused on its asymptotic behavior with respect to block length n under an exponentially increasing size $M = e^{nR}$ and a fixed rate R. Specifically, [5], [17], and [19] investigate this problem based on combinatorial techniques, while the studies in [2] and [11] introduce probabilistic approaches. All the mentioned works concentrated on codes transmitted over binary symmetric channel, where Hamming distance is the only distance measure.

In this paper, we employ a new notion from information-spectrum methodologies [3], [13] to determine the asymptotic minimum covering radius among (a prespecified class of) block codes. As a result, the asymptotic minimum covering radius formula can be established under any alphabet $\mathcal{X}^n$ and any measure on the "distance" between elements in $\mathcal{X}^n$. With the new expression, we can now, for example, investigate the asymptotic minimum covering radius not only for Hamming distance but also for the "quantized distance measure" defined for codes transmitted over quantized channels.

The rest of the paper is organized as follows. In section 2, we transform the problem of determining the *asymptotic minimum covering radius* among block codes into one that minimizes a spectrum function $\Omega_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$ among all codeword-selecting distributions $\boldsymbol{Y}$. A sufficient condition under which uniform $\boldsymbol{Y}$ minimizes the spectrum function is next addressed in section 3. Based on the sufficient condition, the asymptotic minimum covering radius for arbitrary block codes under J-ary quantized channels and constant weight codes under Hamming distance are established in section 4 to display the usage of the spectrum formula.

Throughout the paper, the natural logarithm is employed unless otherwise stated.

2. Asymptotic minimum covering radius for block codes. Define a sphere centered at y^n with radius r as

$$\mathcal{B}_r(y^n) \stackrel{\triangle}{=} \{x^n \in \mathcal{X}^n : \mu_n(x^n, y^n) \le r\}.$$

DEFINITION 2.1 (minimum α-covering radius under codeword set $\mathcal{S}_n$ and covering distribution P_{X^n}). *Fix $\alpha \in [0,1]$. The minimum α-covering radius under codeword set $\mathcal{S}_n$ and distribution P_{X^n} is given by*

$$\rho_\alpha\left(M, \mathcal{S}_n \| X^n\right) \stackrel{\triangle}{=} \inf_{\substack{\mathcal{C}\subset\mathcal{S}_n \\ |\mathcal{C}|=M}} \inf\left\{r \in \Re : P_{X^n}\left(\bigcup_{y^n\in\mathcal{C}} \mathcal{B}_r(y^n)\right) \ge \alpha\right\}.$$

The *α-covering radius* for one specific block code is the smallest sphere radius for which the probability mass of all words, covered by M spheres, is no smaller than α (cf. Figure. 2.1). The probability mass placed on each element x^n in $\mathcal{X}^n$ is assumed to be defined through the covering distribution P_{X^n}. Taking the minimum one among all α-covering radii yields the *minimum α-covering radius*.

It can be verified that the conventional definition of the minimum covering radius $\rho(M, \mathcal{S}_n)$ (cf. (1)) is exactly the 1-*covering radius* under a full-support[2] covering

[2]The support of a distribution is the smallest set with probability mass being equal to 1. Here, "full-support" means the support of P_{X^n} is $\mathcal{X}^n$.

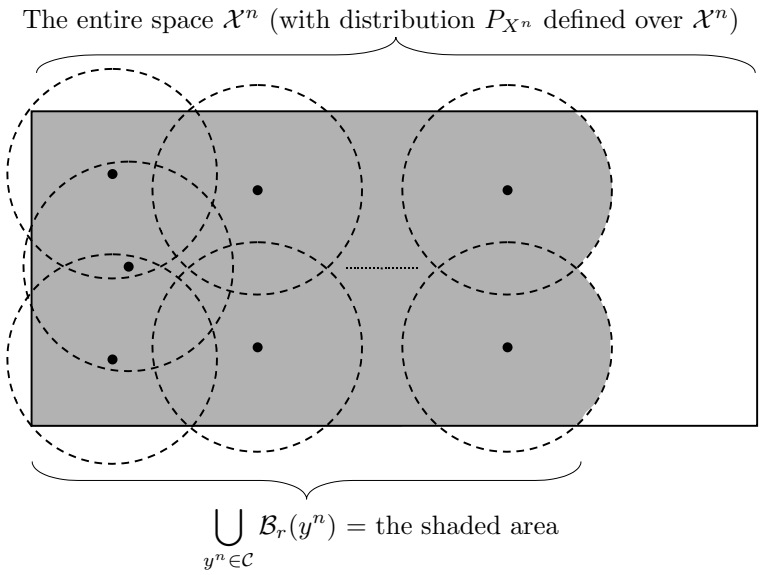


FIG. 2.1. *The α-covering radius for a block code is the smallest radius r such that $P_{X^n}(\cup_{y^n \in \mathcal{C}} \mathcal{B}_r(y^n)) \geq \alpha$.*

distribution P_{X^n}. Specifically, given that the support of P_{X^n} is $\mathcal{X}^n$,

$$\begin{aligned}\rho(M, \mathcal{S}_n) &\stackrel{\triangle}{=} \min_{\substack{\mathcal{C} \subset \mathcal{S}_n \\ |\mathcal{C}|=M}} \max_{x^n \in \mathcal{X}^n} \min_{y^n \in \mathcal{C}} \mu_n(x^n, y^n) \\ &= \inf_{\substack{\mathcal{C} \subset \mathcal{S}_n \\ |\mathcal{C}|=M}} \inf \left\{ r \in \Re : P_{X^n}\left(\bigcup_{y^n \in \mathcal{C}} \mathcal{B}_r(y^n) \right) = 1 \right\} \\ &= \rho_1(M, \mathcal{S}_n \,\|X^n)\,.\end{aligned}$$

Accordingly, the conventional asymptotic covering radius problem is to find the limit, as $n \to \infty$, of the quantity

$$\frac{1}{n}\rho(M, \mathcal{S}_n) = \frac{1}{n}\rho_1(M, \mathcal{S}_n \| X^n)$$

under a full-support distribution P_{X^n} and a fixed rate $R = \log(M)/n$. Since the quantity is investigated as n goes to infinity, it is justified to take $M = e^{nR}$ as integers. Now if the targeted quantity becomes $(1/n)\rho_\alpha(M, \mathcal{S}_n \| X^n)$ instead of $(1/n)\rho(M, \mathcal{S}_n)$, then the full-support assumption on covering distribution P_{X^n} can be relaxed. Equipped with the new setting, one can place larger probability mass on those elements that are considered more essential (to cover) than other elements.

The concept of our method is similar to that of the random coding technique employed in the channel reliability function [1]. Each codeword is assumed to be selected independently of all others from $\mathcal{S}_n$ through a generic distribution P_{Y^n} with $P_{Y^n}(\mathcal{S}_n) = 1$. Then the sphere centered at each random codeword with radius r becomes a *random* variable and so does the resultant codebook. For convenience, we use $\mathcal{B}_r(Y^n)$ and $\boldsymbol{\mathcal{C}}$ to denote the *random* sphere and the *random* codebook, respectively.

LEMMA 2.2. *Fix a sequence of codeword sets*

$$\boldsymbol{\mathcal{S}} = \{\mathcal{S}_n\}_{n \geq 1}, \text{ where } \mathcal{S}_n \subset \mathcal{X}^n \text{ and } |\mathcal{S}_n| > 0,$$

and a triangular array of covering distributions $P_{\boldsymbol{X}} = \{P_{X^n}\}_{n=1}^{\infty}$. *For any triangular-array codeword-selecting process*

$$\boldsymbol{Y} = \boldsymbol{Y}(\boldsymbol{S}) = \left\{Y^n = \left(Y_1^{(n)}, Y_2^{(n)}, \ldots, Y_n^{(n)}\right)\right\}_{n=1}^{\infty}$$

satisfying $P_{Y^n}(\mathcal{S}_n) = 1$ *for each* n, *and any* $\alpha \in [0,1)$,

$$\limsup_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n \| X^n) \le \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) \tag{2}$$

and

$$\liminf_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n \| X^n) \le \underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R), \tag{3}$$

where[3]

$$\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) \stackrel{\triangle}{=} \inf\left\{a \in \Re : \limsup_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > a \middle| X^n\right)^M\right] = 0\right\}$$

and

$$\underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) \stackrel{\triangle}{=} \inf\left\{a \in \Re : \liminf_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > a \middle| X^n\right)^M\right] = 0\right\}.$$

Proof. We will prove only (2). Inequality (3) can be proved by simply following the same procedure.

Let

$$\lambda \stackrel{\triangle}{=} \inf\left\{a \in \Re \ : \ \limsup_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > a \middle| X^n\right)^M\right] = 0\right\}.$$

By definition,

$$\limsup_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > \lambda + \varepsilon \middle| X^n\right)^M\right] = 0 \tag{4}$$

for any $\varepsilon > 0$. Equation (4) then implies that for sufficiently large n,

$$E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > \lambda + \varepsilon \middle| X^n\right)^M\right] < \frac{1-\alpha}{2}. \tag{5}$$

Now for a given codebook $\mathcal{C}$, define

$$\phi_r(x^n | \ \mathcal{C}) \stackrel{\triangle}{=} \begin{cases} 1 & \text{if } x^n \in \bigcup\limits_{y^n \in \mathcal{C}} \mathcal{B}_r(y^n), \\ 0 & \text{otherwise.} \end{cases}$$

[3]Here, we adopt the notation that

$$E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > a \middle| X^n\right)^M\right] = \int_{\mathcal{X}^n} \left(P_{Y^n|X^n}\{y^n \in \mathcal{S}_n : \mu_n(x^n, y^n) > na\}\right)^M dP_{X^n}(x^n)$$
$$= \int_{\mathcal{X}^n} \left(P_{Y^n}\{y^n \in \mathcal{S}_n : \mu_n(x^n, y^n) > na\}\right)^M dP_{X^n}(x^n),$$

where the last step follows since X^n and Y^n are implicitly assumed to be independent.

Then

$$E\left[P_{X^n}\left\{\bigcup_{y^n\in\boldsymbol{\mathcal{C}}}\mathcal{B}_r(y^n)\right\}\right] = E\left[\int_{\mathcal{X}^n}\phi_r(x^n|\boldsymbol{\mathcal{C}})dP_{X^n}(x^n)\right]$$
$$= \int_{\mathcal{X}^n} E\left[\phi_r(x^n|\boldsymbol{\mathcal{C}})\right] dP_{X^n}(x^n),$$

where the expectation is taken with respect to the random codebook $\boldsymbol{\mathcal{C}}$ drawn independently from $\mathcal{S}_n$ according to codeword-selecting process Y^n. By definition,

$$\begin{aligned} E\left[\phi_r(x^n|\boldsymbol{\mathcal{C}})\right] &= 1-\Pr\left\{x^n\notin\mathcal{B}_r(Y_1^n) \text{ and } x^n\notin\mathcal{B}_r(Y_2^n) \text{ and } \cdots \text{ and } x^n\notin\mathcal{B}_r(Y_M^n)\right\}\\ &= 1-\left(\Pr\left\{x^n\notin\mathcal{B}_r(Y^n)\right\}\right)^M\\ &= 1-\left(P_{Y^n}\left\{y^n\in\mathcal{S}_n : \mu_n(x^n,y^n)>r\right\}\right)^M. \end{aligned}$$

Therefore, by taking $r=n(\lambda+\varepsilon)$ and for those n satisfying (5), we obtain

$$\begin{aligned} &E\left[P_{X^n}\left\{\bigcup_{y^n\in\boldsymbol{\mathcal{C}}}\mathcal{B}_{n(\lambda+\varepsilon)}(y^n)\right\}\right]\\ &= \int_{\mathcal{X}^n}\left[1-\left(P_{Y^n}\left\{y^n\in\mathcal{S}_n : \mu_n(x^n,y^n)>n(\lambda+\varepsilon)\right\}\right)^M\right]dP_{X^n}(x^n)\\ &= 1-E_{X^n}\left[\left(\Pr\left\{\frac{1}{n}\mu_n(X^n,Y^n)>\lambda+\varepsilon\,\middle|\, X^n\right\}\right)^M\right]\\ &> 1-\frac{1-\alpha}{2}>\alpha, \end{aligned}$$

which implies that among all possible selections, there exists one codebook $\mathcal{C}\subset\mathcal{S}_n$ satisfying

$$P_{X^n}\left\{\bigcup_{y^n\in\mathcal{C}}\mathcal{B}_{n(\lambda+\varepsilon)}(y^n)\right\}>\alpha.$$

Consequently, $n(\lambda+\varepsilon)\ge\rho_\alpha(M,\mathcal{S}_n\|X^n)$ or, equivalently,

$$\lambda+\varepsilon\ge\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n) \tag{6}$$

for all sufficiently large n. By taking the limsup with respect to n on (6), the proof is completed since ϵ is arbitrary. □

We are now ready to prove the main theorems of the paper.

THEOREM 2.3 (upper bound). *Fix a sequence of codeword set $\boldsymbol{S}=\{\mathcal{S}_n\}_{n\ge1}$. For any covering process $\boldsymbol{X}=\{X^n\}_{n=1}^\infty$,*

$$\sup_{0\le\alpha<1}\limsup_{n\to\infty}\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n)\le\inf_{\boldsymbol{Y}(\boldsymbol{S})}\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$$

and

$$\sup_{0\le\alpha<1}\liminf_{n\to\infty}\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n)\le\inf_{\boldsymbol{Y}(\boldsymbol{S})}\underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R),$$

where the infimum is taken over all processes $\boldsymbol{Y}$ with $P_{Y^n}(\mathcal{S}_n)=1$ (which for convenience is denoted by $\boldsymbol{Y}(\boldsymbol{S})$ in what follows).

Proof. The theorem follows immediately from Lemma 2.2. □

THEOREM 2.4 (lower bound). *Fix a sequence of codeword set* $\boldsymbol{S}=\{\mathcal{S}_n\}_{n\geq 1}$. *For any covering process* $\boldsymbol{X}$,

$$\sup_{0\leq\alpha<1}\limsup_{n\to\infty}\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n)\geq\sup_{\gamma>0}\inf_{\boldsymbol{Y}(\boldsymbol{S})}\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma) \tag{7}$$

and

$$\sup_{0\leq\alpha<1}\liminf_{n\to\infty}\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n)\geq\sup_{\gamma>0}\inf_{\boldsymbol{Y}(\boldsymbol{S})}\underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma). \tag{8}$$

Proof. Again, we will prove only (7), since (8) can be proved in a similar fashion. To prove the inequality in (7), it suffices to prove the existence of $\boldsymbol{Y}(\boldsymbol{S})$ such that

$$\sup_{0\leq\alpha<1}\limsup_{n\to\infty}\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n)+\varepsilon\geq\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma)$$

for any $\varepsilon>0$. This can be justified as follows.

Fix $\varepsilon>0$ and define

$$\lambda\triangleq\sup_{0\leq\alpha<1}\limsup_{n\to\infty}\frac{1}{n}\rho_\alpha(M,\mathcal{S}_n\|X^n).$$

By definition of infimum (cf. Definition 2.1), for any integer $m>1$, there exists a code $\mathcal{C}_n(m)\subset\mathcal{S}_n$ of size M (for each n) such that

$$\rho_{(m-1)/m}(M,\mathcal{S}_n\|X^n)\geq\inf\left\{r\in\Re:P_{X^n}\left[\bigcup_{y^n\in\mathcal{C}_n(m)}\mathcal{B}_r(y^n)\right]\geq\frac{m-1}{m}\right\}-\varepsilon.$$

Therefore,

$$\begin{aligned}\lambda&\geq\limsup_{n\to\infty}\frac{1}{n}\rho_{(m-1)/m}(M,\mathcal{S}_n\|X^n)\\&\geq\limsup_{n\to\infty}\frac{1}{n}\inf\left\{r\in\Re:P_{X^n}\left[\bigcup_{y^n\in\mathcal{C}_n(m)}\mathcal{B}_r(y^n)\right]\geq\frac{m-1}{m}\right\},\end{aligned}$$

which indicates the existence of N_m such that for all $n\geq N_m$,

$$\frac{1}{n}\inf\left\{r\in\Re:P_{X^n}\left[\bigcup_{y^n\in\mathcal{C}_n(m)}\mathcal{B}_r(y^n)\right]\geq\frac{m-1}{m}\right\}<\lambda+\varepsilon.$$

Hence, for $n\geq N_m$,

$$P_{X^n}\left[\bigcup_{y^n\in\mathcal{C}_n(m)}\mathcal{B}_{n(\lambda+\varepsilon)}(y^n)\right]\geq\frac{m-1}{m}.$$

Now, for $\max_{1<i\leq m}N_m\leq n<\max_{1<i\leq(m+1)}N_m$, choose Y^n to be a uniform distribution over $\mathcal{C}_n(m)$ and let

$$\mathcal{V}_n\triangleq\bigcup_{y^n\in\mathcal{C}_n(m)}\mathcal{B}_{n(\lambda+\varepsilon)}(y^n).$$

By noting that for any $x^n \in \mathcal{V}_n$ there exists $y^n \in \mathcal{C}_n(m)$ satisfying

$$\frac{1}{n}\mu_n(x^n, y^n) \le \lambda + \varepsilon,$$

we obtain for all $n \ge \max_{1<i\le m} N_m$,

$$\begin{aligned}
&E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > \lambda + \varepsilon \,\middle|\, X^n\right)^{e^{n(R+\gamma)}}\right] \\
&= \int_{\mathcal{V}_n} \Pr\left(\frac{1}{n}\mu_n(x^n, Y^n) > \lambda + \varepsilon\right)^{Me^{n\gamma}} dP_{X^n}(x^n) \\
&\quad + \int_{\mathcal{V}_n^c} \Pr\left(\frac{1}{n}\mu_n(x^n, Y^n) > \lambda + \varepsilon\right)^{Me^{n\gamma}} dP_{X^n}(x^n) \\
&\le \int_{\mathcal{V}_n} \left(1 - \frac{1}{M}\right)^{Me^{n\gamma}} dP_{X^n}(x^n) + \int_{\mathcal{V}_n^c} 1 \; dP_{X^n}(x^n) \\
&\le \left(1 - \frac{1}{M}\right)^{Me^{n\gamma}} + \left(1 - \frac{m-1}{m}\right) \\
&= \left(1 - \frac{1}{M}\right)^{Me^{n\gamma}} + \frac{1}{m},
\end{aligned}$$

where the superscript "c" applied on $\mathcal{V}_n$ represents the set complementary operation. This result immediately gives

$$\limsup_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > \lambda + \varepsilon \,\middle|\, X^n\right)^{Me^{n\gamma}}\right] \le \frac{1}{m}.$$

Since we can take arbitrarily large m,

$$\limsup_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n, Y^n) > \lambda + \varepsilon \,\middle|\, X^n\right)^{Me^{n\gamma}}\right] = 0.$$

Consequently, $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma) \le \lambda + \varepsilon$. □

Theorems 2.3 and 2.4 together conclude to

$$\inf_{\boldsymbol{Y}(\boldsymbol{S})} \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma) \le \sup_{0\le\alpha<1} \limsup_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n\|X^n) \le \inf_{\boldsymbol{Y}(\boldsymbol{S})} \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$$

and

$$\inf_{\boldsymbol{Y}(\boldsymbol{S})} \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma) \le \sup_{0\le\alpha<1} \liminf_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n\|X^n) \le \inf_{\boldsymbol{Y}(\boldsymbol{S})} \underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$$

for every $\gamma > 0$. A direct interpretation on the quantity of

$$\sup_{0\le\alpha<1} \limsup_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n\|X^n) \quad \left(\text{resp., } \sup_{0\le\alpha<1} \liminf_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n\|X^n)\right)$$

is as follows. It represents, in asymptotics, the minimum radius with which M spheres centered at some node in $\mathcal{S}_n$ can cover *almost* all the words in the support of P_{X^n}.

In other words, the overall probability mass of those words that are not covered can be made arbitrarily small. This requirement is a little weaker if compared to the conventional definition of covering radius, which dictates (as interpreted probabilistically under full-support covering distribution) the probability of all uncovered words being *zero*.

3. A sufficient condition for the minimization of $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$ and $\underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$. The previous section shows that the asymptotic minimum covering radius can be determined by finding

$$\inf_{\boldsymbol{Y}(\boldsymbol{S})} \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) \quad \text{and} \quad \inf_{\boldsymbol{Y}(\boldsymbol{S})} \underline{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R). \tag{9}$$

A natural query following this result is "What is the minimizer for (9)?" In our view, there may not exist a universal solution for this query (since in the spectrum formula, there is no restriction on the distance measure and code alphabet, as well as codeword-selection set and covering distribution.) However, when the distance measure $\mu_n(\cdot,\cdot)$ is symmetric, and a full-support uniform covering distribution under finite alphabet is taken, a sufficient condition under which the uniform $\boldsymbol{Y}$ (over the codeword set) is indeed the desired minimizer can be established. We justify this finding as follows.

By rewriting the spectrum formula as

$$\begin{aligned}\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) &\stackrel{\triangle}{=} \inf\left\{a \in \Re : \limsup_{n\to\infty} E_{X^n}\left[\Pr\left(\frac{1}{n}\mu_n(X^n,Y^n) > a \middle| X^n\right)^M\right] = 0\right\} \\ &= \inf\left\{a \in \Re : \limsup_{n\to\infty} \sum_{x^n\in\mathcal{X}^n} \frac{1}{q^n} P_{Y^n}[y^n \in \mathcal{S}_n : x^n \notin \mathcal{B}_{na}(y^n)]^M = 0\right\},\end{aligned}$$

where $q \stackrel{\triangle}{=} |\mathcal{X}|$, we note that if for any a,

$$\sum_{x^n\in\mathcal{X}^n} P_{Y^n}\left[y^n \in \mathcal{S}_n : x^n \notin \mathcal{B}_{na}(y^n)\right]^M \tag{10}$$

is minimized by uniform Y^n over $\mathcal{S}_n$, so is $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$. The next lemma then gives the basis for the validity of (10) being minimized by the Y^n that is uniformly distributed over $\mathcal{S}_n$.

LEMMA 3.1. *The function $(x_1 + x_2 + \cdots + x_k)^M$ is convex*[4] *in $(x_1, x_2, \ldots, x_k)$ over any pregiven convex set for any positive integer M.*

Proof. We prove this lemma by induction.

1. $M = 1$. The function $(x_1+x_2+\cdots+x_k)$ is apparently convex in $(x_1,x_2,\ldots,x_k)$ over the desired convex set.

2. Assume that the above claim holds for $(M-1)$. Then

$$\begin{aligned}&\lambda(x_1+\cdots+x_k)^M + (1-\lambda)(y_1+\cdots+y_k)^M \\ &\quad -[\lambda(x_1+\cdots+x_k) + (1-\lambda)(y_1+\cdots+y_k)]^M \\ &\geq \lambda(x_1+\cdots+x_k)^M + (1-\lambda)(y_1+\cdots+y_k)^M \\ &\quad - \left[\lambda(x_1+\cdots+x_k)^{M-1} + (1-\lambda)(y_1+\cdots+y_k)^{M-1}\right]\end{aligned}$$

[4] A subset of real vector space is said to be *convex* if $\boldsymbol{x} \in \mathcal{A}$ and $\boldsymbol{y} \in \mathcal{A}$ imply that $\lambda\boldsymbol{x}+(1-\lambda)\boldsymbol{y} \in \mathcal{A}$ for all $\lambda \in [0,1]$. A real-valued function $f(\boldsymbol{x})$ that is defined over a convex set $\mathcal{A}$ is called a convex function if for all $\lambda \in [0,1]$, and for all $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{A}$, $f(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) \leq \lambda f(\boldsymbol{x}) + (1-\lambda)f(\boldsymbol{y})$.

$$
\begin{aligned}
&\times[\lambda(x_1+\cdots+x_k)+(1-\lambda)(y_1+\cdots+y_k)]\\
&=\lambda(1-\lambda)(x_1+\cdots+x_k)^M+\lambda(1-\lambda)(y_1+\cdots+y_k)^M\\
&\quad-\lambda(1-\lambda)(x_1+\cdots+x_k)^{M-1}(y_1+\cdots+y_k)\\
&\quad-\lambda(1-\lambda)(x_1+\cdots+x_k)(y_1+\cdots+y_k)^{M-1}\\
&=\lambda(1-\lambda)[(x_1+\cdots+x_k)-(y_1+\cdots+y_k)]\\
&\quad\times\left[(x_1+\cdots+x_k)^{M-1}-(y_1+\cdots+y_k)^{M-1}\right]\geq 0. \qquad \square
\end{aligned}
$$

LEMMA 3.2. *Under the assumption that* $|\mathcal{X}|<\infty$,

$$\sum_{x^n\in\mathcal{X}^n} P_{Y^n}\left[y^n\in\mathcal{S}_n : x^n\notin\mathcal{B}_{na}(y^n)\right]^M$$

is a convex function in P_{Y^n} *over the convex set*

$$\left\{(P_{Y^n}(y_1^n),\ldots,P_{Y^n}(y_N^n))\in[0,1]^N \;:\; \sum_{i=1}^N P_{Y^n}(y_i^n)=1\right\},$$

where $N=|\mathcal{S}_n|$.

Proof. This can be proved by Lemma 3.1 and the observation that a finite sum of convex functions is convex. $\square$

When the distance measure is symmetric, the quantity (10) can be reformulated as

$$\sum_{x^n\in\mathcal{X}^n} P_{Y^n}\left[\mathcal{S}_n/\mathcal{B}_{na}(x^n)\right]^M, \tag{11}$$

where "/" represents the set subtraction operation. Since it is a convex function defined over a convex set, we can use the Lagrange multiplier technique and the Kuhn–Tucker theorem [1, Thm. A.6] to obtain its global minimizer. To be specific, let

$$f\left(P_{Y^n(\mathcal{S}_n)}\right)\stackrel{\triangle}{=}\sum_{x^n\in\mathcal{X}^n} P_{Y^n}\left[\mathcal{S}_n/\mathcal{B}_{na}(x^n)\right]^M+\lambda\left(\sum_{y^n\in\mathcal{S}_n} P_{Y^n}(y^n)-1\right).$$

Then

$$
\begin{aligned}
\frac{\partial f\left(P_{Y^n(\mathcal{S}_n)}\right)}{\partial P_{Y^n}(y^n)}&=\sum_{x^n\in\mathcal{X}^n} M\cdot P_{Y^n}\left[\mathcal{S}_n/\mathcal{B}_{na}(x^n)\right]^{M-1}\cdot\mathbf{1}\left\{y^n\in\mathcal{S}_n/\mathcal{B}_{na}(x^n)\right\}+\lambda\\
&=M\cdot\sum_{x^n\in\mathcal{X}^n} P_{Y^n}\left[\mathcal{S}_n/\mathcal{B}_{na}(x^n)\right]^{M-1}\cdot\mathbf{1}\left\{x^n\notin\mathcal{B}_{na}(y^n)\right\}+\lambda \;=\; 0, \qquad (12)
\end{aligned}
$$

where (12) follows from the symmetry of the distance measure, and $\mathbf{1}(\cdot)$ is the set indicator function. Taking uniform P_{Y^n} on $\mathcal{S}_n$ into the above equation, we obtain

$$\sum_{x^n\in\mathcal{X}^n}\left(1-\frac{|\mathcal{S}_n\cap\mathcal{B}_{na}(x^n)|}{|\mathcal{S}_n|}\right)^{M-1}\cdot\mathbf{1}\left\{x^n\notin\mathcal{B}_{na}(y^n)\right\}=-\frac{\lambda}{M}. \tag{13}$$

Therefore, if the left-hand side of (13) is independent of $y^n\in\mathcal{S}_n$, then uniform Y^n over $\mathcal{S}_n$ is indeed a solution of (12) (and minimizes $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$).

We conclude the above discussions in the next corollary.

COROLLARY 3.3. *Assume that* $|\mathcal{X}| < \infty$ *and the (generalized) distance measure* $\mu(\cdot,\cdot)$ *is symmetric. If, for every* n,

$$b_r(y^n) = b_r(z^n) \tag{14}$$

holds for every r *and every* y^n, z^n *in* $\mathcal{S}_n$, *then uniform* $\boldsymbol{Y}$ *over* $\boldsymbol{S}$ *minimizes* $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$, *where*

$$b_r(y^n) \stackrel{\triangle}{=} \sum_{x^n \in \mathcal{X}^n} \left(1 - \frac{|\mathcal{S}_n \cap \mathcal{B}_r(x^n)|}{|\mathcal{S}_n|}\right)^{M-1} \cdot \mathbf{1}\left\{x^n \notin \mathcal{B}_r(y^n)\right\}.$$

The condition in (14) may not hold in general. A quick example is to take the codeword set $\mathcal{S}_3 = \{000, 001, 011, 111\}$ under the symmetric Hamming distance metric and binary code alphabet. In such a case, $b_1(000) = b_1(111) = 2 \neq b_1(001) = b_1(011) = 2.5$ for $M = 2$. As a result, the best codeword-selecting distribution that minimizes (11) is $P_{Y^3}(000) = P_{Y^3}(111) = 1/2$ and $P_{Y^3}(001) = P_{Y^3}(011) = 0$, which is uniformly distributed only over a proper subset of $\mathcal{S}_3$.

Two queries can be further studied: whether it suffices to always take $\boldsymbol{Y}$ to be a uniform distribution over a subset of $\boldsymbol{S}$ as hinted by the previous example and whether sufficient condition (14) is also *necessary*. The proof of Theorem 2.4 indicates that the answer to the first query is affirmative; so to speak, taking $\boldsymbol{Y}$ to be the one chosen in the proof of Theorem 2.4, which is uniformly distributed over $\mathcal{C}_n^{(m)}$ for $\max_{1<i\leq m} N_m \leq n < \max_{1<i\leq(m+1)} N_m$, and following the proof of Lemma 2.2, we obtain

$$\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma) - \varepsilon \leq \sup_{0\leq\alpha<1} \limsup_{n\to\infty} \frac{1}{n}\rho_\alpha(M, \mathcal{S}_n \| X^n) \leq \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) + \varepsilon$$

for every $\gamma > 0$ and arbitrary $\varepsilon > 0$. By noting that $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) = \lim_{\gamma\downarrow 0} \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R+\gamma)$, except for countably many points in R, the first query is answered. We, however, have no answer to the second query. From several example trials, it seems affirmative as well. Nevertheless, equipped with the corollary, we can determine the asymptotic minimum covering radius for the examples in the next section.

4. Asymptotic minimum covering radius for specific block coding schemes. In this section, we demonstrate the usage of the new formula to investigate the asymptotic minimum covering radius in terms of two examples: *arbitrary block codes under* J*-ary quantized channels* and *constant weight codes under Hamming distance*.

4.1. Arbitrary block codes under J-ary quantized channels. We consider a model that is frequently used in practical channels (especially when the soft-decision decoding scheme is performed [4]).

Assume that a binary block code is transmitted over a memoryless channel whose output takes values from $\mathcal{N}_J \stackrel{\triangle}{=} \{0, 1, \ldots, J-1\}$; i.e., the output of the channel is quantized to J levels. The distance measure for quantized channels is defined as

$$\mu_n(x^n, y^n) = \sum_{i=0}^{n-1} |x_i - y_i|,$$

where x^n and y^n are in $\mathcal{N}_J^n$. The codeword set and the entire space are, respectively, $\mathcal{S}_n = \{0, (J-1)\}^n$ and $\mathcal{X}^n = \mathcal{N}_J^n$.

To derive the asymptotic minimum covering radius for this channel, we need to first show that $b_r(y^n)$ is independent of $y^n \in \mathcal{S}_n$ (and, therefore, uniform Y^n over $\mathcal{S}_n$ for each n minimizes $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$). We justify this claim as follows.

Observe that for any $y^n \in \mathcal{S}_n$,

(i) $x^n \in \mathcal{B}_r(\mathbf{0})$ iff $z^n \in \mathcal{B}_r(y^n)$, where $\mathbf{0}$ represents the all-zero element, and $z_i = |y_i - x_i|$ for $0 \le i \le n-1$;

(ii) furthermore, for any element $x^n \in \mathcal{X}^n$, $u^n \in \mathcal{S}_n \cap \mathcal{B}_r(x^n)$ iff $v^n \in \mathcal{S}_n \cap \mathcal{B}_r(z^n)$, where z^n is defined the same as above, and

$$v_i \stackrel{\triangle}{=} \begin{cases} (J-1) - u_i & \text{if } x_i \neq z_i, \\ u_i & \text{otherwise} \end{cases}$$

for $0 \le i \le n-1$.

Thus,

$$\begin{aligned} b_r(\mathbf{0}) &= \sum_{x^n \in \mathcal{X}^n} \left(1 - \frac{|\mathcal{S}_n \cap \mathcal{B}_r(x^n)|}{|\mathcal{S}_n|}\right)^{M-1} \cdot \mathbf{1}\left\{x^n \notin \mathcal{B}_r(\mathbf{0})\right\} \\ &= \sum_{z^n \in \mathcal{X}^n} \left(1 - \frac{|\mathcal{S}_n \cap \mathcal{B}_r(z^n)|}{|\mathcal{S}_n|}\right)^{M-1} \cdot \mathbf{1}\left\{z^n \notin \mathcal{B}_r(y^n)\right\} = b_r(y^n) \end{aligned}$$

for all $y^n \in \mathcal{S}_n$.

Now, for uniform Y^n over $\mathcal{S}_n = \{0, J-1\}^n$ (and also uniform X^n over $\mathcal{X}^n = \mathcal{N}_J^n$),

$$\begin{aligned} &\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) \\ &= \inf\left\{a \in \Re : \limsup_{n\to\infty} \frac{1}{J^n} \sum_{x^n \in \mathcal{X}^n} P_{Y^n}\left[y^n \in \mathcal{S}_n \ : \ y^n \notin \mathcal{B}_{na}(x^n)\right]^M = 0\right\} \\ &= \inf\left\{a \in \Re : \limsup_{n\to\infty} \sum_{x^n \in \mathcal{X}^n} \frac{1}{J^n}\left[1 - \frac{|\mathcal{S}_n \cap \mathcal{B}_{na}(x^n)|}{2^n}\right]^M = 0\right\} \\ &= \inf\left\{a \in \Re : \limsup_{n\to\infty} \sum_{\substack{n_0+n_1+\cdots \\ +n_{J-1}=n}} \frac{\frac{n!}{n_0!n_1!\cdots n_{J-1}!}}{J^n}\left[1 - \frac{f_{na}(n_0, n_1, \ldots, n_{J-1})}{2^n}\right]^M = 0\right\}, \end{aligned}$$

where n_i is the number of occurrence i's in x^n, and $f_{na}(n_0, n_1, \ldots, n_{J-1})$ is the summation of all $\binom{n_0}{i_0}\binom{n_1}{i_1}\cdots\binom{n_{J-1}}{i_{J-1}}$ satisfying $0 \le i_j \le n_j$ for $0 \le j \le J-1$ and $\sum_{j=0}^{J-1}[i_j j + (n_j - i_j)(J-j-1)] \le na$. By using typical asymptotic approximation for binomial coefficients, we obtain that for $v_0 + v_1 + \cdots + v_{J-1} = 1$,

$$\begin{aligned} &g(v_0, v_1, \ldots, v_{J-1}, a) \\ &\stackrel{\triangle}{=} \lim_{n\to\infty} \frac{1}{n} \log_2 \frac{2^n}{f_{na}(nv_0, nv_1, \ldots, nv_{J-1})} \end{aligned}$$

$$= \begin{cases} \infty & \text{if } a < \sum_{j=0}^{J-1} v_j \cdot \min\{j, (J-1)-j\}, \\ 1 - \max\limits_{(\delta_0,\ldots,\delta_{J-1})\in\mathcal{D}_J} \left[v_0 H(\delta_0) + \cdots + v_{J-1} H(\delta_{J-1})\right] & \\ \qquad \text{if } \sum_{j=0}^{J-1} v_j \cdot \min\{j, (J-1)-j\} \le a \le \dfrac{J-1}{2}, & \\ 0 & \text{if } a > \dfrac{J-1}{2}, \end{cases}$$

where $H(x) \stackrel{\triangle}{=} -x\log_2(x) - (1-x)\log_2(1-x)$ is the binary entropy function, $\mathcal{D}_J$ consists of all $(\delta_0, \ldots, \delta_{J-1})$ satisfying $0 \le \delta_j \le 1$ for $0 \le j \le J-1$ and $\sum_{j=0}^{J-1}[v_j\delta_j j + v_j(1-\delta_j)(J-j-1)] \le a$, and the result for $a > (J-1)/2$ follows by taking $\delta_1 = \delta_2 = \cdots = \delta_{J-1} = 1/2$. Thus,

$$\sup_{0\le\alpha<1} \lim_{n\to\infty} \frac{1}{n}\rho_\alpha\left(M = e^{nR}, \{0, J-1\}^n\right) = \inf\left\{a \in \Re : \frac{R}{\log(2)} > \max_{v_0+\cdots+v_{J-1}=1} g(v_0, \ldots, v_{J-1}, a)\right\}. \tag{15}$$

We can then derive the asymptotic minimum covering radius for different J values based on (15).

Case A. For J odd,

$$\max_{v_0+\cdots+v_{J-1}=1} g(v_0, \ldots, v_{J-1}, a) = \begin{cases} \infty & \text{if } a < \frac{J-1}{2}, \\ 0 & \text{if } a \ge \frac{J-1}{2}, \end{cases}$$

and, therefore,

$$\sup_{0\le\alpha<1} \lim_{n\to\infty} \frac{1}{n}\rho_\alpha\left(M = e^{nR}, \{0, J-1\}^n\right) = \frac{J-1}{2} \quad \text{for } 0 < R \le \log(2).$$

It can easily be seen that $(J-1)/2$ is the trivial lower bound for the asymptotic minimum covering radius at J odd, since any codeword in $\{0, J-1\}^n$ require radius $(J-1)n/2$ to cover the all-$[(J-1)/2]$ element. Here, in lieu of the new formula, we show that $(J-1)/2$ is actually the asymptotic minimum covering radius at J odd.

Case B. J is even.

Since the case of $J = 2$ reduces to a simple *binary block code* under *Hamming distance*, for which the derivation of its asymptotic minimum covering radius can be easily computed through combinatorial approaches, we will therefore focus on the case of $J \ge 4$.

To derive the asymptotic minimum covering radius, we first establish a general lower bound by using the middle point $x^n_{\text{mid}} \stackrel{\triangle}{=} (J/2-1, J/2-1, \ldots, J/2-1)$ as follows:

$$\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) = \inf\left\{a \in \Re : \limsup_{n\to\infty} \frac{1}{J^n} \sum_{\substack{n_0+n_1+\cdots \\ +n_{J-1}=n}} \frac{n!}{n_0! n_1! \cdots n_{J-1}!} \left[1 - \frac{|\mathcal{S}_n \cap \mathcal{B}_{na}(x^n)|}{2^n}\right]^M = 0\right\}$$

$$\geq \inf\left\{a \in \Re : \limsup_{n\to\infty} \frac{1}{J^n}\left[1 - \frac{|\mathcal{S}_n \cap \mathcal{B}_{na}(x^n_{\text{mid}})|}{2^n}\right]^M = 0\right\}$$
$$= \inf\left\{a \in \Re \ : \ \frac{R}{\log(2)} > 1 - H\left(a - \left(\frac{J}{2} - 1\right)\right)\right\} \text{ for } 0 < R \leq \log(2).$$

We then observe that for $v_0 + v_1 + v_2 + v_3 = 1$ and $1 \leq a \leq 3/2$,

$$g(v_0, v_1, v_2, v_3, a) = 1 - \max_{(\delta_0,\ldots,\delta_3)\in\mathcal{D}_4} [v_0 H(\delta_0) + \cdots + v_3 H(\delta_3)] \leq 1 - H(a-1),$$

where the last step follows by taking $1 - \delta_0 = 1 - \delta_1 = \delta_2 = \delta_3 = a - 1$, which is in the range of the maximization operation. Thus, for $0 < R \leq \log(2)$,

$$\begin{aligned}
&\sup_{0\leq\alpha<1} \lim_{n\to\infty} \frac{1}{n}\rho_\alpha\left(M = e^{nR}, \{0,3\}^n\right) \\
&= \inf\left\{a \in \Re : \frac{R}{\log(2)} > \max_{v_0+v_1+v_2+v_3=1} g(v_0, v_1, v_2, v_3, a)\right\} \\
&\leq \inf\left\{a \in \left[1, \frac{3}{2}\right] : \frac{R}{\log(2)} > \max_{v_0+v_1+v_2+v_3=1} g(v_0, v_1, v_2, v_3, a)\right\} \\
&\leq \inf\left\{a \in \left[1, \frac{3}{2}\right] : \frac{R}{\log(2)} > 1 - H(a-1)\right\}.
\end{aligned}$$

As a result, the general lower bound is tight at $J = 4$.

For $J \geq 6$ even, there seems no simple expression for the asymptotic minimum covering radius. However, one can still obtain a numerically plotted curve for the asymptotic minimum covering radius at $J \geq 6$ even, whenever the algorithmic complexity of the optimization operation for (15) is feasible.

4.2. Binary constant weight codes under Hamming distance. Define the codeword set as

$$\mathcal{S}_n(w) \ = \ \{y^n \in \{0,1\}^n \ : \ W(y^n) = w\},$$

where $W(y^n)$ is the number of 1's in y^n. The covering space is assumed to be the entire space $\mathcal{X}^n = \{0,1\}^n$. Let the distance measure $\mu_n(\cdot,\cdot)$ be the n-fold Hamming distance.

In this case, the asymptotic minimum covering radius for codeword set $\mathcal{S}_n(nv)$ is apparently lower bounded by $\max\{v, 1-v\}$, since the code must cover both the all-zero element and the all-one element. Now, in lieu of the new formula, we can show that $\max\{v, 1-v\}$ is indeed the exact asymptotic minimum covering radius for constant weight codes.

Define $f_r(w,\eta) \stackrel{\triangle}{=} |\mathcal{S}_n(w) \cap \mathcal{B}_r(x^n)|$, where $\eta = W(x^n)$. Then, for $y^n \in \mathcal{S}_n(w)$,

$$\begin{aligned}
b_r(y^n) &= \sum_{\eta=0}^{n} \left(1 - \frac{f_r(w,\eta)}{\binom{n}{w}}\right)^{M-1} \left[\sum_{\{x^n \ : \ W(x^n)=\eta\}} \mathbf{1}\left\{x^n \notin \mathcal{B}_r(y^n)\right\}\right] \\
&= \sum_{\eta=0}^{n} \left(1 - \frac{f_r(w,\eta)}{\binom{n}{w}}\right)^{M-1} \left(|\mathcal{S}_n(\eta)| - |\mathcal{S}_n(\eta) \cap \mathcal{B}_r(y^n)|\right) \\
&= \sum_{\eta=0}^{n} \left(1 - \frac{f_r(w,\eta)}{\binom{n}{w}}\right)^{M-1} \left[\binom{n}{\eta} - f_r(\eta, w)\right],
\end{aligned}$$

which is apparently independent of $y^n \in \mathcal{S}_n(w)$ for every r. Hence, uniform Y^n over $\mathcal{S}_n(w)$ for each n minimizes $\bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R)$.

Now, from the observations that for fixed x^n with $W(x^n)=\eta$ the total number of y^n in $\mathcal{S}_n(w)$ satisfying that the weights (1's) of x^n and y^n coincide with each other in exactly d positions is equal to $\binom{\eta}{d}\binom{n-\eta}{w-d}$, and that $W(x^n)+W(y^n)-2d$ is the Hamming distance between x^n and y^n with d coincidences in their weights, we get[5]

$$\begin{aligned} f_r(w,\eta) &= \sum_{\left\{d\,:\ \substack{0\le d\le \min\{w,\eta\}\\ 0\le w-d\le n-\eta,\ 0\le w+\eta-2d\le r}\right\}} \binom{\eta}{d}\binom{n-\eta}{w-d} \\ &= \sum_{\left\{i\,:\ \substack{0\le (w+\eta-i)/2\le \min\{w,\eta\}\\ 0\le (w-\eta+i)/2\le n-\eta,\ 0\le i\le r}\right\}} \binom{\eta}{\frac{w+\eta-i}{2}}\binom{n-\eta}{\frac{w-\eta+i}{2}} \times \mathbf{1}\left\{(w+\eta-i)\text{ even}\right\} \\ &= \sum_{i=|w-\eta|}^{\min\{r,\ w+\eta,\ 2n-w-\eta\}} \binom{\eta}{\frac{\eta-w+i}{2}}\binom{n-\eta}{\frac{w-\eta+i}{2}} \times \mathbf{1}\left\{(w+\eta-i)\text{ even}\right\}. \end{aligned} \tag{16}$$

Accordingly, for uniform $\boldsymbol{Y}$ over $\boldsymbol{S}(w)$ (and also uniform $\boldsymbol{X}$ over the entire space),

$$\begin{aligned} \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) &= \inf\left\{a\in\Re : \limsup_{n\to\infty}\frac{1}{2^n}\sum_{x^n\in\mathcal{X}^n} P_{Y^n}\left[y^n\in\mathcal{S}_n : y^n\notin\mathcal{B}_{na}(x^n)\right]^M = 0\right\} \\ &= \inf\left\{a\in\Re : \limsup_{n\to\infty}\frac{1}{2^n}\sum_{\eta=0}^{n}\binom{n}{\eta}\left[1-\frac{f_{na}(w,\eta)}{\binom{n}{w}}\right]^M = 0\right\}. \end{aligned}$$

By using typical asymptotic approximation for binomial coefficients, we obtain[6]

$$\begin{aligned} \bar{g}(v,\hat{v},a) &\stackrel{\triangle}{=} \lim_{n\to\infty}\frac{1}{n}\log_2\frac{\binom{n}{nv}}{f_{na}(nv,n\hat{v})} \\ &= \begin{cases} H(v) - \max\limits_{|v-\hat{v}|\le j\le \min\{a,v+\hat{v},2-(v+\hat{v})\}}\left[\hat{v}H\left(\frac{\hat{v}-v+j}{2\hat{v}}\right)+(1-\hat{v})H\left(\frac{v-\hat{v}+j}{2(1-\hat{v})}\right)\right] & \\ \qquad\qquad \text{if } |v-\hat{v}|\le a\le 1, \\ \infty \qquad \text{if } 0\le a<|v-\hat{v}| \end{cases} \\ &= \begin{cases} H(v)-[\hat{v}\cdot H(1-v)+(1-\hat{v})\cdot H(v)] & \text{if } v+\hat{v}-2v\hat{v}\le a\le 1, \\ H(v)-\left[\hat{v}\cdot H\left(\frac{\hat{v}-v+a}{2\hat{v}}\right)+(1-\hat{v})\cdot H\left(\frac{v-\hat{v}+a}{2(1-\hat{v})}\right)\right] & \\ \qquad\qquad \text{if } |v-\hat{v}|\le a<v+\hat{v}-2v\hat{v}, \\ \infty \qquad \text{if } 0\le a<|v-\hat{v}| \end{cases} \\ &= \begin{cases} 0 & \text{if } v+\hat{v}-2v\hat{v}\le a\le 1, \\ H(v)-\left[\hat{v}\cdot H\left(\frac{\hat{v}-v+a}{2\hat{v}}\right)+(1-\hat{v})\cdot H\left(\frac{v-\hat{v}+a}{2(1-\hat{v})}\right)\right] & \\ \qquad\qquad \text{if } |v-\hat{v}|\le a<v+\hat{v}-2v\hat{v}, \\ \infty \qquad \text{if } 0\le a<|v-\hat{v}|. \end{cases} \end{aligned}$$

[5] By definition, d is the number of coincidences in the weights of x^n and y^n, and hence $0\le d\le \min\{w,\eta\}$.

[6] The function $\hat{v}\cdot H(\frac{\hat{v}-v+j}{2\hat{v}})+(1-\hat{v})H(\frac{v-\hat{v}+j}{2(1-\hat{v})})$ is a concave truncated function of j, and is maximized at $j=v+\hat{v}-2v\hat{v}$, if $|v-\hat{v}|\le v+\hat{v}-2v\hat{v}\le\min\{a,\,v+\hat{v},\,2-(v+\hat{v})\}$. Note that $|v-\hat{v}|\le v+\hat{v}-2v\hat{v}\le\min\{v+\hat{v},\,2-(v+\hat{v})\}$ is valid for every $0\le v,\hat{v}\le 1$.

Thus, for $0 \le v \le 1/2$ and $0 < R \le \log(2)$,

$$\begin{aligned}\sup_{0\le\alpha<1} \lim_{n\to\infty} \frac{1}{n}\rho_\alpha\left(M = e^{nR}, \mathcal{S}_n(nv)\right) &= \bar{\Omega}_{\boldsymbol{Y}\|\boldsymbol{X}}(R) \\ &= \inf\left\{a \in \Re \ : \ \frac{R}{\log(2)} > \max_{0\le\hat{v}\le 1} g(v,\hat{v},a)\right\} \\ &= 1 - v,\end{aligned}$$

where the last step follows from[7]

$$\max_{0\le\hat{v}\le 1} g(v,\hat{v},a) = \begin{cases} \infty & \text{if } 0 \le a < \max\{v, 1-v\} = 1-v, \\ 0 & \text{if } \max\{v,1-v\} \le a \le 1. \end{cases}$$

Similarly, for $1/2 < v \le 1$,

$$\sup_{0\le\alpha<1} \lim_{n\to\infty} \frac{1}{n}\rho_\alpha\left(M = e^{nR}, \mathcal{S}_n(nv)\right) = v.$$

Acknowledgment. The authors wish to thank the anonymous reviewers for their valuable suggestions and comments that greatly helped to improve the paper.

REFERENCES

[1] R. E. Blahut, *Principles and Practice of Information Theory*, Addison-Wesley, Reading, MA, 1987.
[2] V. M. Blinovskii, *Lower asymptotic bound on the number of linear code words in a sphere of given radius in f_q^n*, Problemy Peredachi Informatsii, 23 (1987), pp. 50–53.
[3] P.-N. Chen, T.-Y. Lee, and Y. S. Han, *Distance-spectrum formulas on the largest minimum distance of block codes*, IEEE Trans. Inform. Theory, 46 (2000), pp. 869–885.
[4] G. C. Clark, Jr. and J. B. Cain, *Error-Correction Coding for Digital Communications*, Plenum Press, New York, 1981.
[5] G. D. Cohen, *A nonconstructive upper bound on covering radius*, IEEE Trans. Inform. Theory, 29 (1983), pp. 352–353.
[6] G. D. Cohen and P. Frankl, *Good coverings of Hamming spaces with spheres*, Discrete Math., 56 (1985), pp. 125–131.
[7] G. D. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*, North-Holland, Amsterdam, 1997.
[8] G. D. Cohen, M. G. Karpovsky, H. F. Mattson, Jr., and J. R. Schatz, *Covering radius–survey and recent results*, IEEE Trans. Inform. Theory, 31 (1985), pp. 328–343.
[9] G. D. Cohen, S. N. Litsyn, and G. Zémor, *On greedy algorithms in coding theory*, IEEE Trans. Inform. Theory, 42 (1996), pp. 2053–2057.
[10] G. D. Cohen, S. N. Litsyn, A. C. Lobstein, and H. F. Mattson, Jr., *Covering radius 1985–1994*, Appl. Engrg. Comm. Comput., 8 (1997), pp. 173–239.
[11] P. Delsarte and P. Piret, *Do most binary linear codes achieve the Goblick bound on the covering radius?*, IEEE Trans. Inform. Theory, 32 (1986), pp. 826–828.
[12] R. L. Graham and N. J. A. Sloane, *On the covering radius of codes*, IEEE Trans. Inform. Theory, 31 (1985), pp. 385–401.
[13] T. S. Han, *Information-Spectrum Methods in Information Theory*, Baifukan Press, Tokyo, 1998 (in Japanese).
[14] T. Helleseth, T. Kløve, and J. Mykkeltveit, *On the covering radius of binary codes*, IEEE Trans. Inform. Theory, 24 (1978), pp. 627–628.

[7] There exists no $\hat{v} \in [0,1]$ satisfying $|v - \hat{v}| \le a < v + \hat{v} - 2v\hat{v}$, $1 - v \le a \le 1$, and $0 \le v \le 1/2$. This observation can easily be justified by

$$(1-2v) \ge (1-2v)\hat{v} > a - v \ge (1-v) - v = 1 - 2v \quad \text{for } 0 \le \hat{v} \le 1.$$

[15] H. Janwa, *Some new upper bounds on the covering radius of binary linear codes*, IEEE Trans. Inform. Theory, 35 (1989), pp. 110–122.
[16] H. L. Royden, *Real Analysis*, 3rd ed., Macmillan, New York, 1988.
[17] P. Solé, *Asymptotic bounds on the covering radius of binary codes*, IEEE Trans. Inform. Theory, 36 (1990), pp. 1470–1472.
[18] P. Solé, *Packing radius, covering radius, and dual distance*, IEEE Trans. Inform. Theory, 41 (1995), pp. 268–272.
[19] A. A. Tietäväinen, *An asymptotic bound on the covering radii of binary BCH codes*, IEEE Trans. Inform. Theory, 36 (1990), pp. 211–213.
[20] A. A. Tietäväinen, *An upper bound on the covering radius as a function of the dual distance*, IEEE Trans. Inform. Theory, 36 (1990), pp. 1472–1474.
[21] F. Levy-dit-Vehel and S. Litsyn, *More on the covering radius of BCH codes*, IEEE Trans. Inform. Theory, 42 (1996), pp. 1023–1028.