Annals of Discrete Mathematics 21 (1984) 325–356

POLYNOMIAL ALGORITHMS FOR PERFECT GRAPHS

M. GRÖTSCHEL
Institut für Mathematik, Universität Augsburg, Augsburg, W. Germany

L. LOVÁSZ
Institute of Mathematics, Eötvös Loránd University, H-1088 Budapest, Hungary

A. SCHRIJVER
Department of Econometrics, Tilburg University, Tilburg, The Netherlands

We show that the weighted versions of the stable set problem, the clique problem, the coloring problem and the clique covering problem are solvable in polynomial time for perfect graphs. Our algorithms are based on the ellipsoid method and a polynomial time separation algorithm for a certain class of positive semidefinite matrices related to Lovász's bound $\vartheta(G)$ on the Shannon capacity of a graph. We show that $\vartheta(G)$ can be computed in polynomial time for all graphs G and also give a new characterization of perfect graphs in terms of this number $\vartheta(G)$. In addition we prove that the problem of verifying that a graph is imperfect is in *NP*. Moreover, we show that the computation of the stability number and the fractional stability number of a graph are unrelated with respect to hardness (if $P \neq NP$).

1. Introduction and notation

It is well known that the stable set problem, the clique problem, the chromatic number problem and the clique cover problem are *NP*-complete problems for general graphs, cf. [3]. The purpose of this paper is to show that these problems, and even their weighted versions, are solvable in polynomial time for perfect graphs. The algorithms presented here are based on the ellipsoid method (cf. [8], [2], [4]) and on a computationally tractable characterization of the number $\vartheta(G)$ introduced by Lovász [10] in connection with the Shannon capacity of a graph.

In the remaining part of this section we shall introduce our notation and state the problems we shall investigate. The second section gives a brief review of the ellipsoid method and some properties of this method which are important for our purposes. In Section 3 we show that the stable set problem is unrelated to the fractional stable set problem with respect to hardness for general graphs. The Shannon capacity and the numbers $\vartheta(G)$, $\vartheta_w(G)$, which are important for the design of our algorithms, are treated in Section 4, and a polynomial separation algorithm for a certain class of positive semidefinite matrices related to $\vartheta(G)$ is presented in Section 5. This algorithm is utilized together with the ellipsoid

method in Sections 6 and 7 to obtain polynomial time algorithms for the weighted versions of the stable set problem, clique problem, coloring problem and clique cover problem on perfect graphs.

All graphs in this paper are finite and undirected. Since loops and multiple edges do not play a role for the concepts we consider, we assume that all graphs are without such edges, i.e., are simple. A graph is denoted by $G = (V(G), E(G))$ where $V(G)$ (or just V) is the vertex set and $E(G)$ (or just E) is the edge set of G. An edge connecting two vertices i and j is denoted by ij, and we say that two vertices are *adjacent* if they are equal or connected by an edge. The *complementary graph* of a graph G is defined as the graph $\bar{G}$ with $V(\bar{G}) = V(G)$ and in which two different vertices are adjacent if and only if they are nonadjacent in G.

A *stable set* of a graph G is a set of vertices $W \subseteq V(G)$ such that any two vertices of W are nonadjacent in G, and a *clique* of G is a set of vertices $C \subseteq V(G)$ such that any two vertices of C are adjacent in G. The maximum cardinality of a stable set in G is called the *stability number* of G and is denoted by $\alpha(G)$. The maximum cardinality of a clique in G is called the *clique number* of G and is denoted by $\omega(G)$. Clearly, a stable set of G is a clique of $\bar{G}$, and vice versa, thus $\alpha(G) = \omega(\bar{G})$ and $\omega(G) = \alpha(\bar{G})$ hold.

A *k-coloration* of G is a partition of $V(G)$ into k stable sets of G, and the least integer k for which G admits a k-coloration is called the *chromatic number* of G, denoted by $\chi(G)$. A *k-clique cover* of G is a partition of $V(G)$ into k cliques of G, and the least k for which G admits a k-clique cover is called the *clique cover number* of G which is denoted by $\rho(G)$. By definition, every k-coloration of G is a k-clique cover of $\bar{G}$, and vice versa, which implies $\chi(G) = \rho(\bar{G})$ and $\rho(G) = \chi(\bar{G})$.

The problem of finding the stability number (clique number, chromatic number, clique cover number) of a graph is called the *stable set* (*clique*, *coloring*, *clique cover*) *problem*. These four problems have natural weighted versions.

Given a graph $G = (V, E)$ and a 'weight' $w_v \in \mathbb{Z}_+$ for all $v \in V$ ($\mathbb{Z}_+$ is the set of positive integers), then the *weighted stable set problem* (*weighted clique problem*) is to find a stable set W (a clique C) of G such that the sum of the weights of the vertices in W (in C) is as large as possible. The *weighted coloring problem* (*weighted clique problem*) is the following: find stable sets $W_1, W_2, \ldots, W_t$ (cliques $C_1, C_2, \ldots, C_t$) and positive integers $y_1, y_2, \ldots, y_t$ such that for all $v \in V$, $\sum_{W_i \ni v} y_i \geqslant w_v$ ($\sum_{C_i \ni v} y_i \geqslant w_v$) holds, and such that $\sum_{i=1}^{t} y_i$ is as small as possible.

The optimum values of these four problems are denoted by $\alpha_w(G)$, $\omega_w(G)$, $\chi_w(G)$, $\rho_w(G)$ and are called the *weighted stability*, *weighted clique*, *weighted chromatic*, *weighted clique cover number*.

It is obvious that for any graph G, $\alpha(G) \leqslant \rho(G)$ and $\omega(G) \leqslant \chi(G)$ hold. A graph G is called *perfect* if

$$\alpha(G[W]) = \rho(G[W]) \quad \text{for all } W \subseteq V(G)$$

where $G[W]$ denotes the subgraph of G induced by the vertex set $W \subseteq V(G)$. Lovász [9] has shown the so-called *perfect graph theorem*, namely, that a graph G is perfect if and only if its complement $\bar{G}$ is perfect. So the perfect graph theorem is equivalent to the following: a graph G is perfect if and only if

$$\omega(G[W]) = \chi(G[W]) \quad \text{for all } W \subseteq V(G).$$

Due to the perfect graph theorem it suffices to design polynomial time algorithms only for the weighted stable set and the weighted clique cover problem in order to obtain polynomial time algorithms for all the four problems described above on perfect graphs. Namely, suppose we have a polynomial time algorithm for the weighted stable set problem on perfect graphs and we want to find the maximum weighted clique in a perfect graph G. Then, obviously, the set of maximum weighted cliques of G equals the set of maximum weighted stable sets of $\bar{G}$. Since by the perfect graph theorem $\bar{G}$ is perfect, we can apply our polynomial time algorithm to calculate a maximum weighted stable set in $\bar{G}$ and thereby obtain a maximum weighted clique in G. Similarly, if we have a polynomial time algorithm for the weighted clique cover problem in perfect graphs we can obtain a minimum weighted coloring of a perfect graph G by applying our polynomial time algorithm to the (perfect) complementary graph $\bar{G}$.

Therefore, we shall concentrate in the sequel on designing polynomial time algorithms for the weighted stable set and clique cover problem on perfect graphs, keeping in mind that these also yield polynomial time algorithms for the weighted clique and coloring problem on perfect graphs.

There are various classes of graphs known for which the weighted versions of the stable set, the clique, the coloring or the clique cover problem can be solved in polynomial time. For a survey of such results see [3]. These classes of graphs include several classes of perfect graphs, e.g., bipartite, triangulated and comparability graphs as well as line graphs of bipartite graphs. Recently, Hsu [6] has shown that the coloring problem, and Hsu and Nemhauser [7] have shown that the clique and clique cover problem, are solvable in polynomial time for claw-free perfect graphs.

2. The ellipsoid method

Based on an algorithm due to Shor [13], Khachiyan [8] recently devised a method which solves linear programming problems in polynomial time; for proofs, see [2]. This so-called ellipsoid method can be used to derive the

polynomial solvability of a more general class of problems, in particular to obtain a powerful tool for solving combinatorial optimization problems as was described by Grötschel et al. [4]. In this section we give a brief survey of this method and state those theorems of Grötschel et al. [4] which are of interest for the design of polynomial time algorithms on perfect graphs.

A *convex body* is a closed, bounded, fully dimensional, and convex subset of $\mathbb{R}^n$, $n \geq 2$. More precisely, if we speak of a convex body K we always assume that the following information is known: the integer $n \geq 2$ with $K \subseteq \mathbb{R}^n$, two rational numbers $0 < r \leq R$, and a vector $a_0 \in K$ such that

$$S(a_0, r) \subseteq K \subseteq S(a_0, R),$$

where $S(a_0, s) = \{x \in \mathbb{R}^n \mid \|x - a_0\| \leq s\}$ ($\|\cdot\|$ is the euclidean norm), denotes the ball with center a_0 and radius s. Therefore, we also denote a convex body by the quintuple $(K; n, a_0, r, R)$ where we assume that $n \geq 2$, $a_0 \in \mathbb{Q}^n$, $0 < r \leq R$ are given explicitly.

The following two problems are of particular interest and — as we shall see later — polynomially related.

Assume that a convex body $(K; n, a_0, r, R)$ is given.

(2.1) Optimization Problem. Given a vector $c \in \mathbb{Q}^n$ and a rational number $\varepsilon > 0$, find a vector $y \in \mathbb{Q}^n$ such that $d(y, K) \leq \varepsilon$ and $c^T x \leq c^T y + \varepsilon$ for all $x \in K$ (i.e. y is almost in K and almost maximizes $c^T x$ on K).

(2.2) Separation Problem. Given a vector $y \in \mathbb{Q}^n$ and a rational number $\delta > 0$, conclude with one of the following:

(i) asserting that $d(y, K) \leq \delta$ (i.e., y is almost in K); or

(ii) finding a vector $c \in \mathbb{Q}^n$ such that $\|c\| \geq 1$ and for every $x \in K$, $c^T x \leq c^T y + \delta$ (i.e. finding an almost separating hyperplane).

(Here $d(.,.)$ denotes the distance function, i.e., $d(x, y) = \|x - y\|$ and $d(y, K) = \inf\{d(x, y) \mid x \in K\}$.)

The method of Yudin and Nemirovskii [14], section 4.5, would enable us to show that the following third problem is also polynomially related to problems (2.1) and (2.2) above:

(2.2′) Feasibility Problem. Given a vector $y \in \mathbb{Q}^n$ and a rational number $\delta > 0$. conclude with one of the following:

(i) asserting that $d(y, K) \leq \delta$, or

(ii) asserting that $d(y, \mathbb{R}^n \setminus K) \leq \delta$.

Clearly this problem is easier than the separation problem. However, in the

applications in this paper, we shall obtain almost separating hyperplanes automatically.

To speak of a polynomial time algorithm for a convex body K we have to specify how we measure the input length of K. Whenever something is encoded we assume that the (usual) binary encoding is used. A rational number is encoded by encoding the numerator and the denominator.

If $x \in \mathbb{Q}^n$ ($\mathbb{Q}$ is the set of rational numbers) then $\|x\|_\infty$ denotes the maximum of the absolute values of the integers appearing as numerator or denominator in the coefficients of x. In other words, to encode x at least $\log\|x\|_\infty + n$ places are necessary. (In this paper all logarithms have base two.)

For a convex body $(K; n, a_0, r, R)$ we assume that the parameters n, $a_0 \in \mathbb{Q}^n$, $r \in \mathbb{Q}$ and $R \in \mathbb{Q}$ are coded. If $\mathcal{K}$ is a class of convex bodies then the *input* of the optimization (or separation) problem for $\mathcal{K}$ is the code of some member $(K; n, a_0, r, R) \in \mathcal{K}$, of a vector $c \in \mathbb{Q}^n$ and of a rational number $\varepsilon > 0$ (of a vector $y \in \mathbf{q}^n$ and a rational number $\delta > 0$). The *length* or size of the input is the length of this (binary) encoding. Thus, the length of the input is at least

$$n + \log\|r\|_\infty + \log\|R\|_\infty + \log\|\gamma\|_\infty$$

where $\gamma = \varepsilon$ or $\gamma = \delta$. An algorithm to solve the optimization (separation) problem for the class $\mathcal{K}$ is called *polynomial* if its running time is bounded by some polynomial of the size of the input.

(2.3) The ellipsoid method. Given a convex body $(K; n, a_0, r, R)$, a linear objective function $c^T x$ with $\|c\| \geqslant 1$ and a number $\varepsilon > 0$ (the required accuracy). We assume that there is a subroutine SEP(K, y, δ) which for the given convex body K, a vector $y \in \mathbb{Q}^n$ and a rational $\delta > 0$ either concludes that $y \in S(K, \delta) = \{x \in \mathbb{R}^n \mid d(x, K) \leqslant \delta\}$ or yields a vector $d \in \mathbb{Q}^n$ such that $d^T x \leqslant d^T y + \delta$ for all $x \in K$, i.e. SEP solves the separation problem for K. We first define the following numbers:

$$(2.3.1)\quad N := 4n^2 \left\lceil \log \frac{2R^2\|c\|_\infty}{r\varepsilon} \right\rceil,$$

$$(2.3.2)\quad \delta := \frac{R^2 4^{-N}}{300n}$$

$$(2.3.3)\quad p := 5N \left\lceil \log \frac{12\sqrt{n}}{R^2} \right\rceil,$$

and then proceed as follows:

(2.3.4) Set $x_0 = a_0$ (center of the first ellipsoid),
$A_0 := R^2 I_n$ (I_n is the (n, n)-identity matrix);

(2.3.5) **for** $k = 0$ **to** $N-1$ **do**;

(2.3.6) Run the subroutine SEP(K, x_k, δ).

(2.3.7) If SEP(K, x_k, δ) concludes that $x_k \in S(K, \delta)$, we say that k is a feasible index and set $a := c$.

(2.3.8) If SEP(K, x_k, δ) yields a vector $d \in \mathbb{R}^n$ such that $\|d\| \geq 1$ and $\sup\{d^T x \mid x \in K\} \leq d^T x_k + \delta$, we call k an infeasible index and set $a := -d$.

(2.3.9) $b_k := A_k a / \sqrt{a^T A_k a}$,

(2.3.10) $x_k^* := x_k + \frac{1}{n+1} b_k$,

(2.3.11) $A_k^* := \frac{2n^2+3}{2n^2}\left(A_k - \frac{2}{n+1} b_k b_k^T\right)$,

(2.3.12) $x_{k+1} :\approx x_k$, and $A_{k+1} :\approx A_k^*$.

end;

Above, the sign $\approx$ means that the left-hand side is obtained by rounding the binary expansion of the right-hand side after p places behind the point.

Since by construction $x_0 \in K$, the set of feasible indices is nonempty; moreover, we can show the following theorem, cf. [4].

(2.4) Theorem. *Let j be a feasible index for which*

$$c^T x_j = \max\{c^T x_k \mid 0 \leq k < N,\ k \text{ feasible}\}.$$

Then

$$c^T x_j \geq \sup\{c^T x \mid x \in K\} - \varepsilon. \quad \square$$

Clearly, the number N of iterations of the ellipsoid method is polynomial in the size of the input. One can also show that the entries of the intermediate vectors x_k and matrices A_k, $0 \leq k \leq N$, are polynomially bounded. Furthermore, the number δ used to run the separation subroutine is polynomial in the input length. Thus, the ellipsoid method is a polynomial algorithm for the optimization problem for K if and only if the subroutine SEP is a polynomial algorithm for the separation problem for K. This implies, in particular, that whenever there is a polynomial separation algorithm for a class of convex bodies $\mathcal{K}$ there is also a polynomial optimization algorithm for $\mathcal{K}$ (via the ellipsoid method). It is of particular importance that this implication also holds the other way round, namely:

(2.5) Theorem. *Let $\mathcal{K}$ be a class of convex bodies. There is a polynomial algorithm to solve the separation problem for the members of $\mathcal{K}$, if and only if there is a polynomial algorithm to solve the optimization problem for the members of $\mathcal{K}$.* □

Note that according to our definition neither the optimization nor the separation problem are solved exactly; in both cases we allow for a small error. This is necessary because the problem classes that are covered by Theorem (2.5) may also contain instances with a unique optimal solution which has irrational coefficients. But irrational numbers cannot be represented exactly.

In case our class of convex bodies $\mathcal{K}$ is a class of polytopes, then the optimization problem for $\mathcal{K}$ is nothing but a linear programming problem. If in addition all members of $\mathcal{K}$ have a rational defining inequality system, then both the separation and the optimization problem can be solved precisely, we shall say in the *strong sense*. Moreover, it is also possible to construct a dual optimal solution in polynomial time.

If $P \subseteq \mathbb{R}^n$ is a polytope with rational vertices, define $T(P)$ to be the maximum of the absolute values of numerators and denominators occurring in the entries of vertices of P. The pair (P, T) is called a *rational polytope* if $T(P) < T$. The input size of a rational polytope is at least $n + \lceil \log T \rceil$.

It is not difficult to prove that if (P, T) is a rational polytope, then $P \subseteq S(0, nT)$ and if P is fully dimensional then $S(a_0, (nT)^{-2n^3}) \subseteq P$ for some point a_0.

(2.6) Theorem. *Let $\mathcal{K}$ be a class of fully dimensional rational polytopes such that the optimization (or equivalently the separation) problem for $\mathcal{K}$ can be solved in polynomial time. Then the following holds:*

(a) *There is a polynomial optimization algorithm for $\mathcal{K}$ in the strong sense, i.e., which for every member $P \in \mathcal{K}$ and every rational vector c finds a vector $y \in P$ such that $c^T y = \max\{c^T x \mid x \in P\}$.*

(b) *There is a polynomial separation algorithm for $\mathcal{K}$ in the strong sense, i.e., which for every member $P \in \mathcal{K}$, $P \subseteq \mathbb{R}^n$, and every rational vector y either asserts that $y \in P$ or finds a rational vector c with $\|c\| \geq 1$ such that $c^T x < c^T y$ for all $x \in P$. In case $y \in P$ the algorithm also yields vertices $x_0, x_1, \ldots, x_n$ of P and rational numbers $\lambda_0, \lambda_1, \ldots, \lambda_n \geq 0$ such that $\sum_{i=0}^n \lambda_i = 1$ and $\sum_{i=0}^n \lambda_i x_i = y$.*

(c) *There exists a polynomial algorithm which for every $P \in \mathcal{K}$, $P \subseteq \mathbb{R}^n$ and $c \in \mathbb{Z}^n$ provides facets $a_i^T x \leq b_i$ $(i = 1, \ldots, n)$ of P and rational numbers $\lambda_i \geq 0$ $(i = 1, \ldots, n)$ such that $\sum_{i=1}^n \lambda_i a_i = c$ and $\sum_{i=1}^n \lambda_i b_i = \max\{c^T x \mid x \in P\}$.* □

Case (c) of Theorem 2.6 will play an important role in the sequel, since it will provide us with a method to construct a minimum weighted clique cover from a maximum weighted stable set.

A further class of convex bodies will be of interest for our purposes. Let $\mathbb{R}^n_+$ be

the nonnegative orthant and $K \subseteq \mathbb{R}^n$ be a convex body such that there are reals r and R, $0 < r \leq R$, with

$$\mathbb{R}^n_+ \cap S(0, r) \subseteq K \subseteq \mathbb{R}^n_+ \cap S(0, R), \tag{2.7}$$

$$0 \leq x \leq y \in K \Rightarrow x \in K. \tag{2.8}$$

The *anti-blocker* $A(K)$ of K is defined by

$$A(K) := \{y \in \mathbb{R}^n_+ \mid y^T x \leq 1 \text{ for every } x \in K\}. \tag{2.9}$$

It is easy to see that $A(A(K)) = K$ for a K satisfying (2.7) and (2.8), and that in case K is a polytope with vertices $x_1, x_2, \ldots, x_k$ then $A(K) = \{y \in \mathbb{R}^n_+ \mid y^T x_i \leq 1, i = 1, \ldots, k\}$. Moreover, if $\mathcal{K}$ is a class of convex bodies satisfying (2.7) and (2.8) we set $A(\mathcal{K}) = \{A(K) \mid K \in \mathcal{K}\}$.

(2.10) Theorem. *Let $\mathcal{K}$ be a class of convex bodies satisfying* (2.7) *and* (2.8). *Then the optimization problem for $\mathcal{K}$ can be solved in polynomial time if and only if the optimization problem for $A(\mathcal{K})$ can be solved in polynomial time.* □

3. The fractional stable set problem

To be able to utilize the ellipsoid method for combinatorial optimization problems one has to associate a class of convex bodies with the problem class under consideration. Natural candidates are usually the convex hulls of the incidence vectors of feasible solutions. In case of the stable set problem this is done as follows. Let $G = (V, E)$ be a graph with n vertices. For every $W \subseteq V(G)$ denote by x^W the (node-) incidence vector of W, i.e. $x^W_v = 1$ if $v \in W$ and $x^W_v = 0$ if $v \notin W$. Then

$$P(G) := \text{conv}\{x^W \in \mathbb{R}^n \mid W \subseteq V(G) \text{ is a stable set of } G\} \tag{3.1}$$

is called the *stable set polytope* of G. Clearly, every weighted stable set problem on G can be solved as a linear programming problem over $P(G)$. The polytope $P(G)$ is fully dimensional, has 0/1-vertices, and is contained in the unit hypercube, thus $P(G)$ is a rational polytope. If we were able to design a polynomial separation algorithm for $P(G)$, then by Theorem (2.5) the weighted stable set problem would be solvable in polynomial time. Since this problem is *NP*-complete, we cannot expect to find a polynomial separation algorithm for $P(G)$ in general.

A usual approach to solve difficult optimization problems is to consider tight relaxations of the problem in question which are polynomially solvable, and then proceed by branch-and-bound methods. A natural relaxation of the stable set

problem is the so-called *fractional stable set problem.* By definition, no two vertices of a stable set are adjacent. Thus, given any clique C of a graph G, at most one vertex of a stable set can belong to C. This implies that for every clique $C \subseteq V(G)$ and every incidence vector x^W of a stable set $W \subseteq V(G)$ the so-called *clique inequality*

$$\sum_{v \in C} x_v^W \leqslant 1$$

is satisfied. For any graph G with n vertices we call

$$P^*(G) := \left\{ x \in \mathbb{R}^n \mid x_v \geqslant 0 \text{ for all } v \in V(G) \text{ and } \sum_{v \in C} x_v \leqslant 1 \text{ for all cliques } C \subseteq V(G) \right\} \tag{3.2}$$

the *fractional stable set polytope* of G. $P^*(G)$ is clearly a rational polytope. Since obviouly $P(G) \subseteq P^*(G)$, the LP-solution over $P^*(G)$ provides an upper bound for the weight of the optimal stable set in G. For a given graph G and an objective function $w : V \to \mathbb{Z}_+$ let us define the following parameters:

$$\alpha_w(G) := \max\{w^T x \mid x \in P(G)\},$$
$$\alpha_w^*(G) := \max\{w^T x \mid x \in P^*(G)\},$$
$$\alpha^*(G) := \max\left\{ \sum_{v \in V} x_v \mid x \in P^*(G) \right\}.$$

The number $\alpha^*(G)$ is called the *fractional stability number* of G, $\alpha_w^*(G)$ is called the *fractional weighted stability number* of G, and as mentioned earlier $\alpha_w(G)$ is called the *weighted stability number* of G. By definition we have $\alpha(G) \leqslant \alpha^*(G)$ and $\alpha_w(G) \leqslant \alpha_w^*(G)$.

At first sight the polytope $P^*(G)$ looks rather innocent. It is easy to see that its facets are the trivial inequalities $x_v \geqslant 0$ for all $v \in V(G)$ and the clique inequalities $\sum_{v \in C} x_v \leqslant 1$ for all maximal cliques $C \subseteq V(G)$ (maximal with respect to set inclusion). However, it is not known how to find all maximal cliques efficiently, even worse, there are classes of graphs (even perfect ones) such that the number of maximal cliques grows exponentially in $|V(G)|$. So there is no way to represent the constraint system of $P^*(G)$ efficiently. By Theorem (2.5) this is not necessarily crucial, since it is not the number of inequalities which matters; what matters is whether one can find a violated hyperplane in polynomial time. Since the constraint system of $P^*(G)$ looks quite simple one might hope to find a polynomial time separation algorithm for $P^*(G)$. But this is very unlikely as the complexity of the separation problem for $P^*(G)$ is closely related to the complexity of the weighted clique problem. More precisely:

(3.3) Proposition. *Let $\mathcal{G}$ be a class of graphs. Then there is a polynomial algorithm to solve the weighted fractional stable set problem for every member of $\mathcal{G}$ if and only if there is a polynomial algorithm to solve the weighted clique problem for every member of $\mathcal{G}$.*

Proof. For every member G of $\mathcal{G}$, the weighted clique problem can be solved in polynomial time if and only if the linear program $\max w^T x$, $x \in Q(G)$ can be solved in polynomial time, where $Q(G) := \text{conv}\{x^C \in \mathbb{R}^n \mid C \subseteq V(G)$ is a clique$\}$. By definition, the anti-blocker of $Q(G)$ is $A(Q(G)) = \{y \in \mathbb{R}^n_+ \mid y^T x^C \leqslant 1$ for all cliques $C \subseteq V(G)\}$, i.e. $A(Q(G))$ equals $P^*(G)$. Thus by Theorem (2.10) the linear program $\max w^T x$, $x \in Q(G)$ can be solved in polynomial time for every $G \in \mathcal{G}$ if and only if the linear program $\max w^T x$, $x \in P^*(G)$ can be polynomially solved for every $G \in \mathcal{G}$. □

It follows from the examples in [3] that there are various classes of graphs for which the weighted clique, and hence the fractional stable set problem, are solvable in polynomial time. However, since the weighted clique problem is *NP*-complete for the class of all graphs, Proposition (3.3) implies that the weighted fractional stable set problem is *NP*-equivalent. Proposition (3.3) therefore states that considering the fractional stable set problem instead of the stable set problem does not offer considerable advantages. Moreover, the problems of computing $\alpha_w(G)$ and $\alpha^*_w(G)$ seem to be unrelated with respect to difficulty. For planar graphs $\omega_w(G)$ (the weighted clique number) and hence $\alpha^*_w(G)$ can be computed easily in polynomial time, while the determination of $\alpha_w(G)$ for planar (even cubic planar) graphs is *NP*-complete; cf. [3]. So for the complementary graphs of planar graphs the determination of $\omega_w(G)$ and hence $\alpha^*_w(G)$ is *NP*-equivalent, while $\alpha_w(G)$ can be computed in polynomial time.

Although for general graphs the fractional stable set problem does not seem to be useful for computing $\alpha_w(G)$, the situation for perfect graphs is quite particular. Namely, Fulkerson has shown the following (see also [1]):

(3.4) Theorem. *Let G be a graph. Then $P(G) = P^*(G)$ holds if and only if G is perfect.* □

In other words, Theorem (3.4) implies that for every perfect graph G and every objective function w, $\alpha_w(G) = \alpha^*_w(G)$ holds. Therefore a computationally efficient procedure determining $\alpha^*_w(G)$ would yield the desired weighted stability number. As we shall see later $\alpha_w(G)$ and $\alpha^*_w(G)$ can be computed in polynomial time for perfect graphs, however, we do not make direct use of $P(G)$ resp. $P^*(G)$, but rather obtain this result via a detour which will be described in the next section.

4. The Shannon capacity, $\vartheta(G)$ and $\vartheta_w(G)$

The stable set problem has found some nontrivial applications in coding theory, in particular in finding the zero error capacity of a discrete memoryless channel; cf. [12]. Let us denote by $G \cdot H$ the *cartesian product* of the graphs G and H, i.e. $V(G \cdot H) = V(G) \times V(H)$ and two vertices (u, v), $(u', v') \in V(G \cdot H)$ are adjacent if and only if u is adjacent to u' in G and v is adjacent to v' in H. G^k denotes the cartesian product of k copies of G. As an interpretation, consider a graph G whose vertices are letters in an alphabet and in which two vertices are adjacent if and only if they are 'confoundable'. Then the maximum number of one-letter messages which can be sent without danger of confusion is clearly $\alpha(G)$, moreover, $\alpha(G^k)$ is the maximum number of k-letter messages such that any two of them are inconfoundable in at least one coordinate place. It is easy to see that there are at least $\alpha(G)^k$ inconfoundable k-letter words, but in general there may be many more such words. To measure the largest rate at which one can transmit information with an error probability exactly equal to zero, Shannon [12] introduced the following number:

$$\Theta(G) := \sup_k \sqrt[k]{\alpha(G^k)}, \tag{4.1}$$

which is now called the *Shannon capacity* of graph G.

From the fact that $\alpha(G^{k+e}) \geq \alpha(G^k)\alpha(G^e)$ it directly follows that

$$\Theta(G) = \lim_{k \to \infty} \sqrt[k]{\alpha(G^k)} \tag{4.2}$$

and, since $\alpha(G)^k \leq \alpha(G^k)$, that

$$\alpha(G) \leq \Theta(G). \tag{4.3}$$

Shannon [12] obtained an upper bound for $\Theta(G)$ by showing

$$\Theta(G) \leq \alpha^*(G). \tag{4.4}$$

However, both inequalities $\alpha(G) \leq \Theta(G) \leq \alpha^*(G)$ may be strict; e.g., for the pentagon C_5 we have $\alpha(C_5) = 2$ and $\alpha^*(C_5) = 5/2$, while $\Theta(C_5) = \sqrt{5}$; cf. Lovász [10]. For a perfect graph G, Theorem (3.4) implies that $\alpha(G) = \Theta(G) = \alpha^*(G)$ holds. Thus for calculating the stability number of a perfect graph, it would suffice to compute its Shannon capacity. Unfortunately, also the determination of $\Theta(G)$ seems to be a difficult problem, and its value is unknown for large classes of rather simple graphs. Since $\alpha(G)$ and $\alpha^*(G)$ are not very tight bounds for $\Theta(G)$ in general, moreover, they are difficult to compute, as we have seen in the previous section, several authors have introduced parameters which give

better bounds for the Shannon capacity. One such parameter, called $\vartheta(G)$, introduced by Lovász [10], will play a key role in our further development.

Let G be a graph and assume that its vertices are labeled $1, 2, \ldots, n$. We say that a system $(u_1, u_2, \ldots, u_n)$ of vectors in an Euclidean vector space is an *orthonormal representation of* G if each vector u_i has length one and if, for every pair i, j of nonadjacent vertices of G, the vectors u_i and u_j are orthogonal. It is obvious that every graph has an orthonormal representation, e.g., take a set of n orthonormal vectors. Let $\mathcal{U}(G)$ be the set of all orthonormal representations of G, and $\boldsymbol{U}$ be the set of vectors of unit length, then set

$$\vartheta(G) := \min_{(u_1,\ldots,u_n)\in\mathcal{U}(G)} \min_{c\in U} \max_{1\leq i\leq n} \frac{1}{(c^{\mathrm{T}}u_i)^2}. \tag{4.5}$$

Lovász [10] has given various characterizations of this number which we shall list in the sequel. Using the complementary graph $\bar{G}$, $\vartheta(G)$ can be defined alternatively as follows:

$$\vartheta(G) = \max_{(v_1,\ldots,v_n)\in\mathcal{U}(\bar{G})} \max_{d\in U} \sum_{i=1}^{n} (d^{\mathrm{T}}v_i)^2. \tag{4.6}$$

This formula can be used to show that $\vartheta(G)$ is not greater than the fractional stability number. Let $(v_1, \ldots, v_n)$ be an orthonormal representation of $\bar{G}$ and d be a vector of unit length such that these vectors maximize (4.6), i.e., $\vartheta(G) = \sum_{i=1}^{n} (d^{\mathrm{T}}v_i)^2$. Let C be any clique of G. Then, by definition, the vectors v_i, $i \in C$, are pairwise orthogonal, and so $\sum_{i\in C} (d^{\mathrm{T}}v_i)^2 \leq d^{\mathrm{T}}d = 1$. Defining the vector $x = (x_1, \ldots, x_n)^{\mathrm{T}}$ by $x_i := (d^{\mathrm{T}}v_i)^2 \geq 0$ we obtain $\sum_{i\in C} x_i \leq 1$ for all cliques $C \subseteq V(G)$, and thus $x \in P^*(G)$. This implies

$$\vartheta(G) = \sum_{i=1}^{n} (d^{\mathrm{T}}v_i)^2 = \sum_{i=1}^{n} x_i \leq \alpha^*(G). \tag{4.7}$$

The formulas (4.5) and (4.6) do not seem to be very handy computationally, but there are other characterizations of $\vartheta(G)$ which use representations of G by means of symmetric matrices. For any graph G with n vertices we set

$$\begin{aligned}\mathcal{A}(G) := \{A = (a_{ij}) \mid\ & A \text{ is a symmetric } (n,n)\text{-matrix such that } a_{ij} = 1 \\ & \text{if } i = j \text{ or if } i \text{ and } j \text{ are nonadjacent}\};\end{aligned} \tag{4.8}$$

then $\vartheta(G)$ can be described as the following minimum:

$$\vartheta(G) = \min\{\Lambda(A) \mid A \in \mathcal{A}(G)\} \tag{4.9}$$

where $\Lambda(A)$ denotes the largest eigenvalue of A. Equation (4.9) implies that $\vartheta(G)$ is an upper bound on the stability number and on the Shannon capacity of G. Namely, suppose $\alpha(G) = k$, then by definition (4.8) every matrix $A \in \mathcal{A}(G)$ has a principal (k, k)-submatrix, say A_k, all of whose entries are one. Since

$\Lambda(A) \geqslant \Lambda(A_k)$ and k is an eigenvalue of A_k, we have $\Lambda(A) \geqslant k$ for all $A \in \mathcal{A}(G)$, i.e., $\vartheta(G) \geqslant \alpha(G)$. Now Lovász proved

$$\vartheta(G \cdot H) = \vartheta(G)\vartheta(H) \tag{4.10}$$

for all graphs G and H, which implies that

$$\alpha(G^k) \leqslant \vartheta(G^k) = \vartheta(G)^k$$

and hence

$$\Theta(G) \leqslant \vartheta(G). \tag{4.11}$$

The number $\vartheta(G)$ can also be characterized as a maximum of the sum of the entries of certain matrices representing G. Denoting the trace $\sum_{i=1}^{n} b_{ii}$ of a matrix B by $\mathrm{tr}(B)$, we define

$$\begin{aligned}\mathcal{B}(G) := \{B = (b_{ij}) \mid\ & B \text{ is a symmetric positive semidefinite} \\ & (n,n)\text{-matrix with } \mathrm{tr}(B) = 1 \text{ such that} \\ & b_{ij} = 0 \text{ whenever } i,j \in E(G)\};\end{aligned} \tag{4.12}$$

then Lovász [10] showed that

$$\vartheta(G) = \max\left\{ \sum_{i,j=1}^{n} b_{ij} \mid B \in \mathcal{B}(G) \right\}. \tag{4.13}$$

Thus, $\vartheta(G)$ can be considered as a maximum (cf. (4.6) and (4.13)), and as a minimum (cf. (4.5) and (4.9)). Among these characterizations of $\vartheta(G)$, (4.13) will be the most important one in our subsequent investigations.

As a side remark we want to mention that a complementary slackness relation links the two characterizations (4.9) and (4.13) of $\vartheta(G)$. Namely, suppose $B \in \mathcal{B}(G)$, $A \in \mathcal{A}(G)$ and $\Lambda(A)$ is the largest eigenvalue of A, then

$$\begin{aligned}B(\Lambda(A)I_n - A) = 0 \Leftrightarrow\ & B \text{ is optimal for (4.13)} \\ & \text{and } A \text{ is optimal for (4.9).}\end{aligned} \tag{4.14}$$

The inequalities (4.3), (4.7) and (4.11) imply that

$$\alpha(G) \leqslant \Theta(G) \leqslant \vartheta(G) \leqslant \alpha^*(G) \tag{4.15}$$

holds for all graphs G. We remarked earlier that for the pentagon C_5, $\alpha(C_5) = 2 < \Theta(C_5) = \sqrt{5} < \alpha^*(C_5) = \frac{5}{2}$. Since for the pentagon $\Theta(C_5)$ equals $\vartheta(C_5)$, the last inequality in (4.15) may also be strict. Haemers [5] showed the existence of graphs G with $\Theta(G) < \vartheta(G)$. Therefore all these four numbers, $\alpha(G)$, $\vartheta(G)$, $\Theta(G)$ and $\alpha^*(G)$, are different in general. However, for a perfect graph G, Theorem (3.4) implies that equality holds in all inequalities (4.15).

A graph $G = (V, E)$ is called *critically imperfect* if G is not perfect but if the vertex deleted subgraph $G - v$ is perfect for all $v \in V$. There are only two

classes of critically imperfect graphs known, namely, the cycles of odd length and their complementary graphs. We shall now prove that for critically imperfect graphs, $\alpha(G) < \vartheta(G) < \alpha^*(G)$.

Padberg [11] showed that if G is a critically imperfect graph with n vertices then $n = \alpha(G)\omega(G)+1$, and that every critically imperfect graph has exactly n stable sets of cardinality $\alpha(G)$ and n cliques of cardinality $\omega(G)$. He also proved that every vertex of G is contained in exactly $\alpha(G)$ maximum stable sets and in exactly $\omega(G)$ maximum cliques.

Moreover, Padberg [11] showed that the so-called stable set-vertex incidence matrix of a critically imperfect graph is nonsingular, i.e., $S = (s_{ij})$ is an (n, n)-matrix whose rows correspond to the n maximum stable sets of size $\alpha(G)$, whose columns correspond to the n vertices of G, and where $s_{ij} = 1$ ($s_{ij} = 0$) if vertex j belongs (does not belong) to the maximum stable set i. By the properties of critically imperfect graphs mentioned above, S is an (n, n)-matrix such that every row and every column contains exactly $\alpha(G)$ ones.

Now consider for a critically imperfect graph the matrix $B' := S^{\mathrm{T}}S$. The properties of S imply that B' is positive define. An entry b'_{ij} of B' counts the number of maximum stable sets to which both i and j belong. Hence $b'_{ii} = \alpha(G)$, and $b'_{ij} = 0$ if $ij \in E$. Moreover, the sum of the entries of each row (or column) of B' equals $\alpha(G)^2$. Let $\lambda'_1 \geqslant \lambda'_2 \geqslant \cdots \geqslant \lambda'_n$ be the eigenvalues of B'. Since B' is positive definite we have $\lambda'_n > 0$; moreover, since B' is integral, $\det(B') \geqslant 1$. It follows that

$$\sum_{i=1}^{n} \lambda'_i = \operatorname{tr}(B') = n\alpha(G) \quad \text{and} \quad \prod_{i=1}^{n} \lambda'_i = \det(B') \geqslant 1.$$

Using a rough estimate we obtain $\lambda'_n \geqslant (\prod_{i=1}^{n-1} \lambda'_i)^{-1} \geqslant (n\alpha(G))^{-n+1}$. These observations imply that the matrix

$$B'' := \frac{1}{n\alpha(G)} B'$$

is a positive definite matrix contained in $\mathcal{B}(G)$ whose smallest eigenvalue λ''_n is not smaller than $(n\alpha(G))^{-n}$ and for which $\sum_{i,j=1}^{n} b''_{ij} = \alpha(G)$. Hence, if we subtract $(n\alpha(G))^{-n}$ from every element of the main diagonal of B'' and then multiply the resulting matrix by $(n\alpha(G))^n/((n\alpha(G))^n - n)$ we obtain a matrix B which has trace one, is positive semidefinite, and satisfies $b_{ij} = 0$ if $i, j \in E(G)$. Thus $B \in \mathcal{B}(G)$, and since $\alpha(G) \geqslant 2$

$$\sum_{i,j=1}^{n} b_{ij} = \frac{(n\alpha(G))^n - n/\alpha(G)}{(n\alpha(G))^n - n}\, \alpha(G) > \alpha(G),$$

and it follows from (4.13) that for a critically imperfect graph G

$$\vartheta(G) \geqslant \frac{(n\alpha(G))^n - \frac{1}{2}n}{(n\alpha(G))^n - n}\, \alpha(G). \tag{4.16}$$

We now prove that $\vartheta(G) < \alpha^*(G)$. Padberg [11] has shown that for a critically imperfect graph G, $\alpha^*(G) = \alpha(G) + 1/\omega(G)$, and that there is a unique point $y \in P^*(G)$, namely,

$$y = \frac{1}{\omega(G)}(1, \ldots, 1)^T,$$

which satisfies $\sum_{i=1}^n y_i = \alpha^*(G)$. By (4.6) there exists an orthonormal representation $v_1, \ldots, v_n$ of $\bar{G}$ and a vector d of unit length such that $\vartheta(G) = \sum_{i=1}^n (d^T v_i)^2$. As shown in the section following formula (4.6), the vector $x \in \mathbb{R}^n$ with $x_i = (d^T v_i)^2$, $i = 1, \ldots, n$ is contained in the fractional stable set polyhedron $P^*(G)$. Now suppose that $\vartheta(G) = \alpha^*(G)$, then

$$\sum_{i=1}^n x_i = \vartheta(G) = \alpha^*(G) = \sum_{i=1}^n y_i,$$

hence by the uniqueness of y we have

$$\frac{1}{\omega(G)} = y_i = x_i = (d^T v_i)^2, \quad i = 1, \ldots, n.$$

Thus for the orthogonal representation $v_1, \ldots, v_n$ of $\bar{G}$ and the vector $d \in U$ we have

$$\max_{1 \leq i \leq n} \frac{1}{(d^T v_i)^2} = \omega(G)$$

and therefore formula (4.5) implies that $\vartheta(\bar{G}) \leq \omega(G) = \alpha(\bar{G})$. However, the complementary graph of a critically imperfect graph is critically imperfect too, so $\vartheta(\bar{G}) \leq \alpha(\bar{G})$ contradicts (4.16). This implies that $\vartheta(G)$ cannot equal $\alpha^*(G)$. Summing up we have shown that for any critically imperfect graph G

$$\alpha(G) < \vartheta(G) < \alpha^*(G) = \alpha(G) + \frac{1}{\omega(G)}. \tag{4.17}$$

Every imperfect graph contains an induced subgraph which is critically imperfect, i.e., a subgraph for which (4.17) holds, while for every induced subgraph G' of a perfect graph G, $\alpha(G') = \vartheta(G') = \alpha^*(G')$ is satisfied. This implies the following characterization of perfect graphs.

(4.18) Theorem. *A graph G is perfect if and only if the following holds:*

$$\alpha(G[W]) = \vartheta(G[W]) \quad \textit{for all } W \subseteq V(G). \quad \square$$

Our discussion above yields a further characterization, namely, a graph G is perfect if and only if $\alpha^*(G[W]) = \vartheta(G[W])$ for all $W \subseteq V(G)$.

The number $\vartheta(G)$ is not necessarily equal to $\alpha(G)$, even worse, Konjagin (unpublished) has constructed a sequence of graphs G_n with n vertices such that

$\alpha(G_n) = 2$ and $\vartheta(G_n) \to \infty$. This implies that there is no function f at all such that $\vartheta(G) \leq f(\alpha(G))$ holds for all graphs G. It seems to be an interesting problem if there exists a polynomially computable function $\varphi(G)$ and a function f such that $\alpha(G) \leq \varphi(G) \leq f(\alpha(G))$.

An efficient way to calculate $\vartheta(G)$ provides us only with a good algorithm for the unweighted stable set problem in perfect graphs. In order to cover the weighted case too we now generalize $\vartheta(G)$ to a weighted version $\vartheta_w(G)$.

Assume that a graph $G = (V, E)$ and a weight function $w : V \to \mathbb{Z}_+$ are given. Define the graph G_w to be the graph arising from G by replacing each vertex v of G by w_v pairwise nonadjacent new vertices and where two vertices of G_w are adjacent if and only if their originals in G are different adjacent vertices. This construction implies that $\alpha_w(G) = \alpha(G_w)$ holds. Moreover, Lovász [9] has shown that if G is perfect, then G_w is also perfect (in fact, this is the key lemma for the perfect graph theorem). Hence for perfect graphs we have $\alpha_w(G) = \vartheta(G_w) = \alpha_w^*(G)$. Therefore we define

$$\vartheta_w(G) := \vartheta(G_w). \tag{4.19}$$

Note, however, that the existence of a polynomial algorithm for calculating $\vartheta(G)$ does not give a polynomial algorithm for $\vartheta_w(G)$ by applying this algorithm to G_w, since no algorithm making up G_w from G and w is polynomial in the input length $O(|E| + \log \|w\|_\infty)$. A characterization of $\vartheta_w(G)$ using the set $\mathcal{B}(G)$ defined in (4.12) and avoiding this construction is given in the following theorem.

(4.20) Theorem. *Let $G = (V, E)$ be a graph and $w : V \to \mathbb{Z}_+$ a weight function, then*

$$\vartheta_w(G) = \max \left\{ \sum_{i,j=1}^{n} \sqrt{w_i w_j}\, b_{ij} \;\middle|\; B = (b_{ij}) \in \mathcal{B}(G) \right\}. \tag{4.21}$$

Proof. Let M be the maximum in (4.21). We first prove $M \leq \vartheta_w(G)$ using formula (4.13) for $\vartheta(G)$. Suppose $B = (b_{ij}) \in \mathcal{B}(G)$. Replace every entry b_{ij} by a (w_i, w_j)-matrix all of whose entries are $(w_i w_j)^{-1/2} b_{ij}$ to obtain an (r, r)-matrix B', where $r = \sum_{i=1}^{n} w_i$. B' is clearly symmetric and satisfies

$$\operatorname{tr}(B') = \sum_{i=1}^{r} b'_{ii} = \sum_{i=1}^{n} b_{ii} = 1;$$

moreover, the definition implies that $b'_{ij} = 0$ if $i, j \notin E(G)$. It is also easy to see that B' is positive semidefinite. Furthermore, simple calculation shows

$$\sum_{i,j=1}^{r} b'_{ij} = \sum_{i,j=1}^{n} \sqrt{w_i w_j}\, b_{ij}$$

which implies $M \leqslant \vartheta_w(G)$.

Conversely, with each matrix $B' \in \mathscr{B}(G_w)$ we can associate a matrix $\bar{B} \in \mathscr{B}(G)$ such that $\sum'_{i,j=1} b'_{ij} \leqslant \sum_{i,j=1}^{n} \sqrt{w_i w_j}\, \bar{b}_{ij}$ holds, which proves $\vartheta_w(G) \leqslant M$. Namely, take $B' \in \mathscr{B}(G_w)$. Replace the (w_i, w_j)-submatrix induced by the copies of i and j, by the sum of its entries divided by $\sqrt{w_i w_j}$, and eventually add a nonnegative number to any diagonal entry to make the trace equal to one. □

5. A separation algorithm for a class of positive semidefinite matrices with trace one

In this section we shall describe a polynomial time separation algorithm for the class of positive semidefinite matrices $\mathscr{B}(G)$ defined in (4.12). Every set $\mathscr{B}(G)$ is clearly convex and bounded, but not fully dimensional. Since the ellipsoid method, as described in Section 2, can only be applied to convex bodies, we have to replace the sets $\mathscr{B}(G)$ for technical reasons by fully dimensional ones.

For every (n, n)-matrix $B = (b_{ij})$, i.e., $B \in \mathbb{R}^{n \times n}$, and every graph G we define the following *projection operation* $\mathbb{R}^{n \times n} \to \mathbb{R}^{\bar{n}}$ where $\bar{n} := n + \binom{n}{2} - |E(G)| - 1$. Discard from B all elements below the main diagonal; all $|E(G)|$ elements b_{ij}, $i < j$, corresponding to different adjacent vertices $i, j \in V(G)$; and the element b_{nn}. Denote the vector of $\mathbb{R}^{\bar{n}}$ obtained from B in this way by $\bar{B}$, i.e., $\bar{B}$ is an $\bar{n}$-vector whose components are indexed by $F = \{ii \mid i = 1, \ldots, n-1\} \cup \{ij \mid i < j$ and $ij \notin E(G)\}$.

Conversely, for every vector $\bar{B} \in \mathbb{R}^{\bar{n}}$ we define an *extension operation* $\mathbb{R}^{\bar{n}} \to \mathbb{R}^{n \times n}$ by setting $b_{ii} := \bar{b}_{ii}$, $i = 1, \ldots, n-1$; $b_{nn} := 1 - \sum_{i=1}^{n-1} \bar{b}_{ii}$; $b_{ij} = b_{ji} = 0$ for $i, j \in E(G)$; $b_{ij} = b_{ji} = \bar{b}_{ij}$ for $i, j \notin E(G)$, $i < j$. By definition the (n, n)-matrix B obtained by extending $\bar{B} \in \mathbb{R}^{\bar{n}}$ is a symmetric matrix with trace one. Now set

$$\bar{\mathscr{B}}(G) := \{B \in \mathbb{R}^{\bar{n}} \mid B \in \mathscr{B}(G)\}; \tag{5.1}$$

$\bar{\mathscr{B}}(G)$ is convex, since it is the projection of a convex set. We now prove that $\bar{\mathscr{B}}(G)$ is fully dimensional and bounded, i.e. that $\bar{\mathscr{B}}(G)$ is a convex body. Denote by $\bar{B}_n$ the projection of the matrix

$$B_n := \frac{1}{n} I_n \in \mathscr{B}(G)$$

(I_n is the (n, n)-identity matrix); then the following holds:

$$S\left(\bar{B}_n, \frac{1}{n^2\sqrt{n}}\right) \subseteq \bar{\mathscr{B}}(G) \subseteq S(\bar{B}_n, 1). \tag{5.2}$$

Set $r = (n^2\sqrt{n})^{-1}$. To prove that $S(\bar{B}_n, r) \in \bar{\mathscr{B}}(G)$ we show that for any $\bar{B} \in S(\bar{B}_n, r)$ the extension $B \in \mathbb{R}^{n \times n}$ is in $\mathscr{B}(G)$. First observe the following. If

$\bar{B} \in \mathbb{R}^{\bar{n}}$ and $B \in \mathbb{R}^{n \times n}$ is the extension of $\bar{B}$ then $\|\bar{B} - \bar{B}_n\| \leq r$ implies $\|B - B_n\| \leq \sqrt{n}r$. Now take any $\bar{B} \in S(\bar{B}_n, r)$ and let B be its extension. If b_{ij}, $i \neq j$, is any nonzero entry of B, then because of symmetry $b_{ij} = b_{ji} \neq 0$. Now $\|B - B_n\| \leq \sqrt{n}r$ implies $\sqrt{2}\, b_{ij} \leq \sqrt{n}r$, i.e., $|b_{ij}| \leq (n^2\sqrt{2})^{-1}$. If b_{ii} is a diagonal element of B, then $\|B - B_n\| \leq \sqrt{n}r$ implies

$$\left| \frac{1}{n} - b_{ii} \right| \leq n^{-2} \quad \text{or} \quad b_{ii} \geq \frac{n-1}{n^2}.$$

It follows that

$$\sum_{i \neq j} |b_{ij}| \leq (n-1)(n^2\sqrt{2})^{-1} < \frac{n-1}{n^2} \leq b_{ii}.$$

Now by Gershgorin's theorem, all eigenvalues of the extension B of $\bar{B}$ are positive, hence $B \in \mathcal{B}(G)$, which proves that $\bar{B} \in \bar{\mathcal{B}}(G)$. Using the fact that for positive semidefinite matrices $b_{ii}b_{jj} \geq b_{ij}^2$ holds and that the trace of $B \in \mathcal{B}(G)$ is one, it easily follows that $\bar{\mathcal{B}}(G) \subseteq S(\bar{B}_n, 1)$.

Expression (5.2) shows that the logarithms of the radii $r = 1/n^2\sqrt{n}$ and $R = 1$ as well as the numerators and denominators of the interior point $\bar{B}_n$, are bounded in absolute value by a polynomial in n which is fixed over all graphs G, i.e., given a graph G with n vertices, these numbers can be computed in polynomial time. So we may apply Theorem (2.5) to compute $\vartheta_w(G)$ via Theorem (4.20) by using the projection $\bar{\mathcal{B}}(G)$ of $\mathcal{B}(G)$. More precisely, define the following class of convex bodies:

$$\bar{\mathcal{B}} := \{\bar{\mathcal{B}}(G) \mid G \text{ is a graph with } |V(G)| \geq 2\}. \tag{5.3}$$

In order to solve the optimization problem for $\bar{\mathcal{B}}$ it is sufficient to find a polynomial separation algorithm for $\bar{\mathcal{B}}$. We shall now show that the separation problem for $\bar{\mathcal{B}}$ is solvable in polynomial time, even in the strong sense. Given a graph G, then this problem is the following:

(5.4) Problem. Given a vector $\bar{B} \in \mathbb{R}^{\bar{n}}$, conclude with one of the following:

(i) asserting that $\bar{B} \in \bar{\mathcal{B}}(G)$, or

(ii) finding a vector $\bar{D} \in \mathbb{R}^{\bar{n}}$ such that $\|\bar{D}\| \geq 1$ and for every $\bar{X} \in \bar{\mathcal{B}}(G)$, $\bar{D}^{\mathrm{T}}\bar{X} \leq \bar{D}^{\mathrm{T}}\bar{B}$.

In principle, this separation problem reduces to checking the positive semidefiniteness of a symmetric (n, n)-matrix. Thus, given $\bar{B} \in \mathbb{R}^{\bar{n}}$ we extend $\bar{B}$ to a symmetric (n, n)-matrix $B = (b_{ij})$ with trace one and $b_{ij} = b_{ji} = 0$ if $i, j \in E(G)$. To assert that $\bar{B} \in \bar{\mathcal{B}}(G)$ we have to prove that B is positive semidefinite. There are various characterizations of positive semidefiniteness which can be used for the design of an efficient proof of this property. For

instance, polynomial time algorithms can be obtained from Gaussian elimination and Cholesky decomposition. In Gaussian elimination we allow pivots on the main diagonal only; if the rank of the matrix is found and only positive pivots have been carried out, then the matrix is positive semidefinite. Cholesky decomposition can be used in a similar way. A further method is to compute the smallest eigenvalue λ_n; if λ_n is nonnegative then the matrix is positive semidefinite. This algorithm may be fast in practice but is not necessarily polynomially bounded. The method we shall describe now is based on Gaussian elimination and certain determinant calculations. In the following we assume that a graph G with n vertices is given. $\mathcal{B}(G)$ and $\bar{\mathcal{B}}(G)$ are the sets defined in (4.12) resp. (5.1)

(5.5) Separation algorithm for $\bar{\mathcal{B}}(G)$, SEP$(G, \bar{B})$. Given a vector $\bar{B} \in \mathbb{Q}^{\bar{n}}$, where $\bar{n} = n + \binom{n}{2} - |E(G)| - 1$.

(5.5.1) Extend $\bar{B}$ to a symmetric (n, n)-matrix B with trace 1.

(5.5.2) Use Gaussian elimination to compute the rank, say k, of B and a principal (k, k)-submatrix of B having full rank.

(Note that Gaussian elimination automatically gives a nonsingular (k, k)-submatrix, say B_{IJ} (where I is a row- and J a column-index set with $|I| = |J| = k$), of B. Since $\mathrm{tr}(B) = 1$, k is nonzero. It is well-known that if B_{IJ} is nonsingular and has the same rank as B, then both principal (k, k)-submatrices B_{II} and B_{JJ} are nonsingular, in fact

$$\det(B_{II})\det(B_{JJ}) = \det(B_{IJ})^2.$$

Let us denote the principal (i, i)-submatrix of B consisting of the first i rows and columns of B by B_i. For ease of exposition we assume that the principal (k, k)-submatrix B_k of B has rank k and is the one obtained in step (5.5.2). One can easily show that B is positive semidefinite if and only if B_k is positive definite. Moreover, positive definiteness is easy to check, namely, B_k is positive definite if and only if $\det(B_i) > 0$ for $i = 1, \ldots, k$. Therefore our algorithm continues as follows.)

(5.5.3) Compute $\det(B_i)$ for $i = 1, \ldots, k$.

(5.5.4) If $\det(B_i) > 0$ for $i = 1, \ldots, k$, then B is positive semidefinite and hence $\bar{B} \in \bar{\mathcal{B}}(G)$ is proved. **stop!**

(If the test in (5.5.4) is failed, then B is not positive semidefinite and we have to calculate a separating hyperplane.)

(5.5.5) Let t be the smallest index such that $\det(B_t) \leqslant 0$ and define a vector $d = (d_1, \ldots, d_n)^T$ as follows:

$$d_i := 0 \quad \text{for all } i > t,$$

$$d_1 := -1 \quad \text{if } t = 1,$$

$$d_i := (-1)^i \det(B_{it}), \quad i = 1, \ldots, t, \quad \text{if } t > 1,$$

where B_{it} denotes the $(t-1, t-1)$-submatrix of B_t obtained by removing the i-th row and t-th column from B_t.

(5.5.6) Define the following vector $\tilde{D} \in \mathbb{Q}^{\bar{n}}$:

$$\tilde{d}_{ii} := d_n^2 - d_i^2, \quad i = 1, \ldots, n-1,$$

$$\tilde{d}_{ij} := -2d_i d_j, \quad \text{for } i, j \notin E(G) \text{ and } i < j,$$

and return $\tilde{D}$ (if $\|\tilde{D}\| < 1$ we have to scale such that $\|\tilde{D}\| \geqslant 1$). □

The vector $\tilde{D} \in \mathbb{R}^{\bar{n}}$ gives the desired separating hyperplane in case $\bar{B}$ does not belong to $\bar{\mathcal{B}}(G)$. More exactly

$$\tilde{D}\bar{X} \leqslant d_n^2 \leqslant \tilde{D}\bar{B} \quad \text{for all } \bar{X} \in \bar{\mathcal{B}}(G). \tag{5.6}$$

To prove (5.6) define the (n, n)-matrix $D = (d_{ij})$ by setting $d_{ij} = d_i d_j$, i.e., $D = dd^T$. It is obvious from the definitions of D and $\tilde{D}$ that for every vector $\bar{X} \in \mathbb{R}^{\bar{n}}$ and its extension $X \in \mathbb{R}^{n \times n}$ we have

$$\tilde{D}^T \bar{X} = d_{nn} - \sum_{i,j=1}^{n} d_i x_{ij} d_j = d_n^2 - d^T X d.$$

Now if $\bar{X} \in \bar{\mathcal{B}}(G)$, then the extension X is positive semidefinite, i.e., $d^T X d \geqslant 0$, which implies $D^T \bar{X} \leqslant d_n^2$. If $\bar{X} = \bar{B}$ and B is the extension of $\bar{B}$ then the following holds:

$$d^T B d = \sum_{i,j=1}^{n} d_i d_j b_{ij} = \det(B_t)\det(B_{t-1}) \leqslant 0, \tag{5.7}$$

where in case $t = 1$, $\det(B_0)$ is assumed to be one. (5.7) can be obtained by exploiting the definition of d, cf. (5.5.5), and using determinant expansions. Thus, (5.7) shows that $\tilde{D}\bar{B} \geqslant d_n^2$ and (5.6) is proved.

Altogether we have used Gaussian elimination once in (5.5.2) and we have performed at most $2n$ determinant calculations in (5.5.3) and (5.5.5). Since Gaussian elimination and determinant calculation can be done in $O(n^3)$ time the overall running time of our separation algorithm is at most $O(n^4)$ (not considering the length of numbers). Summarizing the discussion above we get the following theorem.

(5.8) Theorem. *There exists an algorithm* SEP(.,.) *such that for any graph* G *with* n *vertices and any vector* $\bar{B} \in Q^{\bar{n}}$, $\bar{n} = n + \binom{n}{2} - |E(G)| - 1$, SEP$(G, \bar{B})$ *asserts whether* $\bar{B} \in \bar{\mathscr{B}}(G)$ *or produces a vector* $\tilde{D} \in Q^{\bar{n}}$ *such that* $\tilde{D}\bar{X} \leqslant \tilde{D}\bar{B}$ *for all* $\bar{X} \in \bar{\mathscr{B}}(G)$.

The running time of SEP$(G, \bar{B})$ *is bounded by a polynomial in* n *and in* $\lceil \log \| \bar{B} \|_\infty \rceil$.

To give an example for the sets $\mathscr{B}(G)$, $\bar{\mathscr{B}}(G)$ and the separation algorithm for $\bar{\mathscr{B}}(G)$ we consider the graph $\bar{K}_2$ which has two vertices and no edges. Then

$$\mathscr{B}(\bar{K}_2) = \left\{ \begin{pmatrix} a & b \\ b & c \end{pmatrix} \,\middle|\, a \geqslant 0,\ c \geqslant 0,\ a + c = 1,\ \det \begin{pmatrix} a & b \\ b & c \end{pmatrix} \geqslant 0 \right\},$$

$$\bar{\mathscr{B}}(\bar{K}_2) = \{(a, b)^{\mathrm{T}} \mid 0 \leqslant a \leqslant 1,\ (a - \tfrac{1}{2})^2 + b^2 \leqslant \tfrac{1}{4}\},$$

i.e., $\bar{\mathscr{B}}(\bar{K}_2)$ is the ball in $\mathbb{R}^2$ around the point $(\frac{1}{2}, 0)^{\mathrm{T}}$ with radius $\frac{1}{2}$.

Consider the point $\bar{B} = (\frac{1}{2}, 1)^{\mathrm{T}} \in \mathbb{R}^2$. The extension of $\bar{B}$ is the matrix

$$B = \begin{pmatrix} \frac{1}{2} & 1 \\ 1 & \frac{1}{2} \end{pmatrix}.$$

B has rank 2 and $\det(B_1) = \frac{1}{2}$, $\det(B_2) = \det(B) = -\frac{3}{4}$. So B is not positive semidefinite and the smallest index t with $\det(B_t) < 0$ is $t = 2$. Using (5.5.5) and (5.5.6) we obtain $d_1 = -1$, $\det(B_{12}) = -1$, $d_2 = \det(B_{22}) = \frac{1}{2}$, and hence $\tilde{d}_{11} = d_2^2 - d_1^2 = -\frac{3}{4}$, $\tilde{d}_{12} = 1$, i.e., $\tilde{D} = (-\frac{3}{4}, 1)^{\mathrm{T}}$. Thus, by (5.6) we have

$$\tilde{D}\bar{X} = -\tfrac{3}{4}a + b \leqslant d_2^2 = \tfrac{1}{4} \leqslant \tilde{D}\bar{B} = (-\tfrac{3}{4}, 1)\begin{pmatrix} \frac{1}{2} \\ 1 \end{pmatrix} = \tfrac{5}{8}$$

for all $\bar{X} \in \bar{\mathscr{B}}(\bar{K}_2)$. The set $\bar{\mathscr{B}}(\bar{K}_2)$, the hyperplane $\tilde{D}\bar{X} = \frac{1}{4}$ and the point $\bar{B}$ are shown in Fig. 5.1.

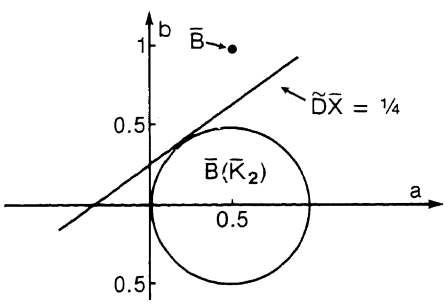

Fig. 5.1.

6. Polynomial algorithms to compute $\vartheta_w(G)$ for all graphs and to solve the weighted stable set and clique problem in perfect graphs

We shall now use the separation algorithm SEP(.,.) for $\bar{\mathcal{B}}(G)$ described in the foregoing section as a subroutine of the ellipsoid method to compute $\vartheta_w(G)$ for every graph G and to find a maximum weighted stable set (or clique) of a perfect graph. Recall that for any graph G with a weight function $w: V \to \mathbb{Z}_+$ Theorem (4.20) implies

$$\vartheta_w(G) = \max\left\{ \sum_{i,j=1}^{n} \sqrt{w_i w_j} b_{ij} \mid B = (b_{ij}) \in \mathcal{B}(G) \right\},$$

i.e. $\vartheta_w(G)$ is the maximum value of a programming problem with a convex feasible region and with a linear objective function (having a particular form). We saw that it is necessary to replace $\mathcal{B}(G)$ by the convex body $\bar{\mathcal{B}}(G)$ (cf. (4.12) and (5.1)) for technical reasons. We therefore have to replace the optimization problem over $\mathcal{B}(G)$ by a corresponding optimization problem over $\bar{\mathcal{B}}(G)$.

So, given an objective function $\sum_{i,j} c_{ij} b_{ij}$ such that $c_{ij} = c_{ji}$ for $\mathcal{B}(G)$ as above, we then define the following objective function $\hat{c}$ for $\bar{\mathcal{B}}(G)$ by setting

$$\hat{c}_{ii} := c_{ii} - c_{nn}, \quad i = 1, \ldots, n-1,$$

$$\hat{c}_{ij} := 2c_{ij} \quad \text{if } i, j \notin E(G),\ i < j.$$

Defining

$$\hat{\vartheta}_c(G) := \max\left\{ \sum_{i,j \in F} \hat{c}_{ij} \bar{b}_{ij} \mid \bar{B} \in \bar{\mathcal{B}}(G) \right\}$$

it is then easy to see that

$$\hat{\vartheta}_c(G) = \max\left\{ \sum_{i,j=1}^{n} c_{ij} b_{ij} \mid B \in \mathcal{B}(G) \right\} - c_{nn}. \tag{6.1}$$

Therefore, in order to approximate $\vartheta_w(G)$ it is sufficient to approximate $\hat{\vartheta}_c(G)$ for $c_{ij} = \sqrt{w_i w_j}$.

Since the numbers $\sqrt{w_i w_j}$ occurring in the objective function are not necessarily rational, we have to approximate these numbers in such a way that the optimum of the problem with the perturbed objective function does not differ too much from the true optimum.

Suppose we want to calculate $\vartheta_w(G)$ up to an error $\varepsilon > 0$, then we claim that the following approximation of the $\sqrt{w_i w_j}$ is sufficient. Using, e.g., the method of continued fractions, determine rational numbers u_{ij} with

$$|u_{ij} - \sqrt{w_i w_j}| < \frac{\varepsilon}{2n(n+1)} \tag{6.2}$$

where, in addition, the denominators of the u_{ij} are at most $2n(n+1)/\varepsilon$. Then we solve the program

$$\hat{\vartheta}_u(G) = \max\left\{ \sum_{i,j\in F} \hat{u}_{ij}\bar{b}_{ij} \mid \bar{B} \in \bar{\mathcal{B}}(G)\right\}.$$

Assume that $\bar{B} \in \bar{\mathcal{B}}(G)$ is the true optimum solution of this program, then since $\|\bar{B}\| \leq 1$, no component $\bar{B}_{ij}$ of $\bar{B}$ is larger than one in absolute value. This implies that the error we make with respect to the original objective function is componentwise

$$|(\hat{u}_{ij} - 2\sqrt{w_i w_j})\bar{B}_{ij}| < \frac{\varepsilon}{n(n+1)}.$$

As the number of components $\bar{n}$ of $\bar{B}$ is at most $n(n+1)/2-1$ and the error in u_{nn} is also at most $\varepsilon/2n(n+1)$ we have from (6.1) that

$$|\hat{\vartheta}_u(G) + u_{nn} - \vartheta_w(G)| < \varepsilon/2.$$

In other words, if the u_{ij} are chosen according to (6.2) then the desired number $\vartheta_w(G)$ is contained in the interval $(\hat{\vartheta}_u(G) + u_{nn} - \varepsilon/2,\ \hat{\vartheta}_u(G) + u_{nn} + \varepsilon/2)$. This implies that if we compute $\hat{\vartheta}_u(G) + u_{nn}$ up to an error $\varepsilon/2$ we obtain $\vartheta_w(G)$ within an accuracy of ε. Such an approximation can be achieved with the ellipsoid method.

So suppose a graph G, $|V(G)| = n$, with weight function $w : V(G) \to \mathbb{Z}_+$ and a required accuracy $\varepsilon > 0$ for $\vartheta_w(G)$ are given, then the following algorithm THETA $(G, w, \varepsilon, \tau)$ finds a number τ with $|\tau - \vartheta_w(G)| < \varepsilon$.

(6.3) Algorithm. THETA$(G, w, \varepsilon, \tau)$. The graph $G = (V(G), E(G))$, $|V(G)| = n \geq 2$, the natural numbers w_i, $i \in V(G)$, and the rational number $\varepsilon > 0$ are the input of the algorithm, while the number τ is the output of the algorithm.

(6.3.1) Approximate the numbers $\sqrt{w_i w_j}$, $1 \leq i \leq j \leq n$, by rationals u_{ij} satisfying (6.2) and whose denominators are at most $2n(n+1)/\varepsilon$. Set

$$\hat{u}_{ii} := u_{ii} - u_{nn},\ i = 1, \ldots, n-1 \text{ and } \hat{u}_{ij} := 2u_{ij} \text{ for } i < j,\ i, j \notin E(G).$$

(We now approximate the optimum value of the program $\max\{\hat{u}^{\mathrm{T}}\bar{B} \mid \bar{B} \in \bar{\mathcal{B}}(G)\} + u_{nn}$ up to an error $\varepsilon/2$ using the ellipsoid method.)

(6.3.2) Set $r = 1/n^2\sqrt{n}$, $R = 1$, $\varepsilon := \varepsilon/2 - \varepsilon/2n(n+1)$ and define the paramters N, δ, p as in the ellipsoid method (2.3). (For the choice of the radii, cf. (5.2), the accuracy ε is chosen according to the previous discussion.)

(6.3.3) Set $A_0 := R^2 I_{\bar{n}}$ and choose as center x_0 of the first ellipsoid the projection $\bar{B}_n \in \bar{\mathcal{B}}(G)$ of $(1/n)I_n$.

(Recall that $\bar{n} = n + \binom{n}{2} - |E(G)| - 1$, and that by (5.2) $\bar{B}_n$ is an interior point of $\bar{\mathcal{B}}(G)$.)

(6.3.4) **for** $k = 0$ **to** $N-1$ **do**;
1. Run the separation algorithm SEP(G, x_k) defined in (5.5).
2. If $x_k \in \bar{\mathcal{B}}(G)$ then set $a := \hat{u}$.
3. If $x_k \notin \bar{\mathcal{B}}(G)$ and if $\tilde{D}$ is the vector returned by SEP(G, x_k), cf. (5.5.6), then set $a := -\tilde{D}$.
4. Make the updates of the ellipsoid method as described in (2.3.9)–(2.3.11).

End;

(6.3.5) Let σ be the value of the best feasible solution of $\max\{\hat{u}^T \bar{B} \mid \bar{B} \in \bar{\mathcal{B}}(G)\}$ found in (6.3.4).
Set $\tau := \sigma + u_{nn}$ and return τ. □

Algorithm (6.3) describes how we can approximte $\vartheta_w(G)$ in polynomial time. Letting w be the vector all of whose components are one, we can use algorithm THETA to compute $\vartheta(G)$ up to any given accuracy in polynomial time for every graph G. So $\vartheta(G)$ is not only well-characterized by the formulas (4.5), (4.6), (4.9) and (4.13), it is also well-behaved computationally.

Theorem (3.2) and Theorem (4.20) imply that for perfect graphs the numbers $\alpha_w(G)$ and $\vartheta_w(G)$ coincide. Moreover, since our weight function w is integer valued, we know that the value $\alpha_w(G)$ of the optimum weighted stable set is an integer. Therefore, in order to find the optimum value of a weighted stable set problem on a perfect graph we only need to approximate $\vartheta_w(G)$ up to an error $\varepsilon \leqslant \frac{1}{2}$ with the algorithm THETA$(G, w, \varepsilon, \tau)$ and round τ to the next integer to obtain $\alpha_w(G)$.

We can also use the algorithm THETA to find a maximum weighted stable set explicitly. This goes as follows.

Let a graph $G = (V, E)$ with $n \geqslant 2$ vertices, weights $w_i \in \mathbb{Z}_+$ for all $i \in V$ and an accuracy $0 < \varepsilon \leqslant \frac{1}{2}$ be given.

(6.4) Algorithm. STABLESET(G, w, ε).

(6.4.1) *Initialization and first guess for* $\alpha(G)$:
1. Run THETA$(G, w, \varepsilon, \tau)$ and round τ to the next integer, say t. (Clearly, $t \geqslant \alpha(G)$ and if G is perfect then $t = \alpha(G)$.)
2. If $|t - \tau| \geqslant \varepsilon$, then **stop** and conclude that G is not perfect.
3. Call all vertices of G unlabeled.

(6.4.2) *Termination check*
If all vertices of the present graph G are labeled, then do:

1. If $V(G)$ is not stable then **stop** and conclude that G is not perfect.
2. If $V(G)$ is stable then $V(G)$ constitutes a maximum weighted stable set of the original graph. **stop!**

(6.4.3) Choose an unlabeled vertex $v \in V$ and tentatively remove v from G, i.e., set $G' = G - v$ and set $w'_u = w_u$ for all $u \in V \setminus \{v\}$.

(6.4.4) Run THETA$(G', w', \varepsilon, \tau)$ and round τ to the next integer, say s. If $|s - \tau| \geqslant \varepsilon$, then **stop** and conclude that G is not perfect.

(6.4.5) If $s = t$, then remove v definitely, i.e., set $G = G'$ and $w = w'$.

(6.4.6) If $|V(G)| = 1$, then label the remaining node.

(6.4.7) If $s < t$, then label v.

(6.4.8) Go to (6.4.2). □

To prove the correctness of the algorithm suppose first that G is a perfect graph. Then for every induced subgraph G' of G, $\vartheta_{w'}(G') = \alpha_{w'}(G')$. In step (6.4.1) we approximate $\vartheta_w(G)$ by $\varepsilon \leqslant \frac{1}{2}$, and, therefore, rounding τ to the next integer gives the true value t for $\alpha(G)$. Now we remove a vertex v from G. If the number s calculated in (6.4.4) satisfies $s = t$ then we know that $G - v$ contains a stable set which is a maximum weighted stable set of G, so we can remove v without distroying all optimum solutions of the stable set problem for G, and we can continue with this procedure. If however $s \neq t$, then all optimum stable sets of G necessarily contain vertex v. Thus we label v, keep v in our vertex set and continue. This way we will finally end up with a graph G' whose set of vertices is labeled, i.e., none of the vertices can be removed without reducing $s = \alpha_{w'}(G') = \alpha_w(G) = t$. This means that every vertex of G' is contained in all optimum stable sets of G'. In other words, the vertex set of G' is itself a stable set, and since $\alpha_{w'}(G') = \alpha_w(G)$, this vertex set is a maximum weighted stable set of G. Thus if G is perfect, then STABLESET will produce an optimum weighted stable set of G.

If however G is not perfect, then STABLESET may detect the imperfectness of G but may also deliver a maximum weighted stable set (without recognizing the imperfectness of G). If in step (6.4.1) or (6.4.4) we find that $|t - \tau| \geqslant \varepsilon$ resp. $|s - \tau| \geqslant \varepsilon$ then the interval $(\tau - \varepsilon, \tau + \varepsilon)$ contains no integer. Since $\alpha_{w'}(G')$ is an integer for all induced subgraphs G' of G and THETA guarantees $\vartheta_{w'}(G') \in (\tau - \varepsilon, \tau + \varepsilon)$ this implies $\alpha_{w'}(G') \neq \vartheta_{w'}(G')$, i.e., by Theorem (4.18) we can conclude that G is not perfect. It may however happen that in every step (6.4.1) and (6.4.4) the approximation τ of $\vartheta_{w'}(G')$ is in the ε-neighborhood of an integer and we will end up in step (6.4.2) with an induced subgraph G' of G whose vertex set V' is labeled. If V' is not a stable set, then V' is not a solution of our

stable set problem. Since the algorithm works for perfect graphs, we can conclude that G is not perfect. If V' is a stable set we have to show that V' is in fact a maximum weighted stable set of G. This can be seen as follows. Suppose G'' is the last subgraph of G created during the algorithm such that a vertex, say v, was definitely removed from G''. Then we know that all other vertices of G'' will finally be labeled, so $V(G'') = V' \cup \{v\}$. Moreover, v was removed because the number s obtained in (6.4.4) by running THETA$(G'' - v, w', \varepsilon, \tau)$ satisfies $s = t$. Since V' is stable, $G' = G'' - v$ is a perfect graph, so we have $s = \alpha_{w'}(G')$, and since t is an upper bound for $\alpha_w(G)$, s is the weighted stability number of G. This proves our claim.

The following (imprecise) argument shows that in case of imperfect graphs all outcomes described above are possible. Consider the pentagon C_5, then $\vartheta(C_5) = \sqrt{5} = 2.236\ldots$. Suppose we run STABLESET with $\varepsilon = \frac{1}{2}$, then the τ we get in step (6.4.1) may equal 2.1 or 2.6. If $\tau = 2.1$ then $t = 2$ and the algorithm will continue finding a maximum stable set of C_5. If $\tau = 2.6$ then $t = 3$, and since the removal of every vertex from C_5 results in a perfect graph we shall get $s = 2$ every time we execute step (6.4.4). This means that finally all vertices of C_5 will be labeled, but these of course do not solve our problem. If however we had chosen $\varepsilon = 0.1$, then $\tau \in (\sqrt{5} - 0.1, \sqrt{5} + 0.1)$ and we obtain $t = 2$ and $|t - \varepsilon| \geqslant 0.13 > \varepsilon$. This implies that the algorithm would stop in step (6.4.1) concluding that C_5 is not perfect.

Thus algorithm STABLESET has two possible outcomes. Either a maximum weighted stable set of G is found or imperfectness of G is proved. Note that in the former case it does not prove perfectness.

STABLESET can also be used to find a maximum weighted clique of a perfect graph G. We simply run STABLESET on the complementary graph $\bar{G}$ which by the perfect graph theorem is perfect again.

Summarizing the observations of this section we obtain the following theorem.

(6.5) Theorem. (a) *There exists an algorithm which for any graph $G = (V, E)$, $|V| \geqslant 2$, any weight function $w : V \to \mathbb{Z}_+$ and any rational $\varepsilon > 0$ finds a number τ such that $|\tau - \vartheta_w(G)| < \varepsilon$ holds.*

The running time of this algorithm is polynomial in $|V|$, $\lceil \log \| w \|_\infty \rceil$ and $\lceil \log \| \varepsilon \|_\infty \rceil$.

(b) *There exists an algorithm which for any perfect graph $G = (V, E)$, $|V| \geqslant 2$ and any weight function $w : V \to \mathbb{Z}_+$ finds a maximum stable set (resp. maximum weighted clique) and the running time of which is polynomial in $|V|$ and $\lceil \log \| w \|_\infty \rceil$.* □

Algorithm THETA and inequality (4.16) can be combined to design a polynomial time nondeterministic algorithm which checks the imperfectness of a

given graph. Namely, suppose G' is a critically imperfect graph with n vertices, then choosing a suitable ε, e.g., $\varepsilon = \frac{1}{2}n^{-2n}$, we run THETA$(G', e, \varepsilon, \tau)$ where e is the vector all of whose components are one. By Theorem (2.4), the choice of ε, and inequality (4.16) the number τ we obtain satisfies

$$\tau \in (\vartheta(G') - \varepsilon, \vartheta(G') + \varepsilon), \quad \tau < \alpha(G) + \frac{1}{\omega(G)} + \varepsilon$$

and therefore $\tau - \varepsilon > \alpha(G')$, $\tau + \varepsilon < \alpha(G) + 1$. Since $\log \| \varepsilon \|_\infty$ is polynomial in n, THETA runs in time polynomial in n. In other words, given a critically imperfect graph, we can verify its imperfectness in polynomial time. As every imperfect graph contains a critically imperfect graph, say G', we can guess this graph G' and then apply the algorithm described above. This shows that verification of imperfectness is an *NP*-problem, hence verification of perfectness is a co-*NP* problem. Note that if the strong perfect graph conjecture is true, then this fact is trivial, since checking imperfectness would then be possible by guessing an odd hole or antihole.

7. A polynomial algorithm for the weighted clique cover and coloring problem for perfect graphs

The separation algorithm for $\bar{\mathcal{B}}(G)$ presented in Section 5 provides us — as shown in Section 6 — via the ellipsoid method with a polynomial time algorithm for solving the weighted stable set problem for perfect graphs. Seen from a different point of view this means that the class of linear programming problems $\max c^T x$, $x \in P(G) = \text{conv}\{x^W \mid W \text{ stable set in } G\}$, G a perfect graph, is solvable in polynomial time. Since $P(G)$ is a fully dimensional rational polytope, Theorem (2.6) implies that the optimization problem as well as the separation problem for $P(G)$, G perfect, are solvable in polynomial time, even in the strong sense. By Theorem (3.2), for a perfect graph G the stable set polytope $P(G)$ equals the fractional stable set polytope $P^*(G)$, thus for this class of graphs we can decide in polynomial time whether a given vector y belongs to

$$P(G) = P^*(G) = \Big\{ x \mid x_v \geqq 0 \text{ for all } v \in V, \sum_{v \in C} x_v \leqq 1 \text{ for all cliques } C \subseteq V(G) \Big\}.$$

A different approach to solving the separation problem for $P(G)$ not using Theorems (2.5) and (2.6) is of course to apply the algorithm which finds a maximum weighted clique, where the given vector $y \geqq 0$ is used as the vector defining the objective function. If the maximum clique, say C, satisfies $y^T x^C =$

$\sum_{v \in C} y_v \leqslant 1$ then $y \in P(G)$, otherwise this clique inequality provides a separating hyperplane. Since the optimum clique algorithm is nothing but the optimum stable set algorithm applied to the complementary graph $\bar{G}$ (which is also perfect), we can use the algorithm STABLESET directly to solve the separation problem for $P(G)$, G perfect.

Theorem (2.6) has a further important consequence. Since the optimization problem for the class of rational polytopes $P(G)$, G perfect, is solvable in polynomial time we can find facets of $P(G)$ and rationals $\lambda_i \geqslant 0$ satisfying the conditions of statement (c) of (2.6). Since the facets of $P(G)$ are of the form $-x_v \leqslant 0$, $v \in V(G)$, and $\sum_{v \in C} x_v \leqslant 1$, $C \subseteq V(G)$ a maximal clique, Theorem (2.6) (c) implies that for any objective function $w : V(G) \to \mathbb{Z}_+$ we can find in polynomial time (maximal) cliques $C_i \subseteq V(G)$ and positive rational numbers λ_i, $i = 1, \ldots, r \leqslant |V(G)|$ such that

$$\sum_{i=1}^{r} \lambda_i = \alpha_w(G) \quad \text{and} \quad \sum_{i=1,\, v \in C_i}^{r} \lambda_i \geqslant w_v \quad \text{for all } v \in V(G).$$

Suppose for a node $v \in V(G)$ strict inequality holds in the above inequality, say $u_v := \sum'_{i=1, v \in C_i} \lambda_i - w_v > 0$, then pick a clique, say C_j, which contains v. If $\lambda_j \leqslant u_v$ then replace C_j by the clique $C_j \setminus \{v\}$, otherwise add the new clique $C_j \setminus \{v\}$ as clique C_{r+1} to our list of cliques and define new parameters as follows: $\lambda_j := \lambda_j - u_v$, $\lambda_{r+1} := u_v$. Then the sum of the λ_i's still equals $\alpha_w(G)$ and the gap u_v of the inequality corresponding to v is either zero or is strictly reduced. By continuing this process we end up with a list of cliques $C_1, \ldots, C_t$ and positive rationals $\lambda_1, \ldots, \lambda_t$ such that

$$\begin{aligned} &\sum_{i=1}^{t} \lambda_i = \alpha_w(G), \\ &\sum_{i=1,\, v \in C_i}^{t} \lambda_i = w_v \quad \text{for all } v \in V(G). \end{aligned} \tag{7.1}$$

Note that in the algorithm described above only those vertices $v \in V(G)$ were considered for which the inequality $-x_v \leqslant 0$ had a positive multiplier λ_v. Since for every such vertex at most one additional clique was added, we still have $t \leqslant |V(G)|$.

By definition, for a perfect graph G the stability number $\alpha(G)$ equals the clique cover number $\rho(G)$. Moreover, since for a perfect graph G the graph G_w (cf. Section 4) is also perfect and as $\alpha_w(G) = \alpha(G_w)$, $\rho_w(G) = \rho(G_w)$, the weighted clique cover number $\rho_w(G)$ equals the weighted stability number $\alpha_w(G)$. This implies that for a perfect graph G, algorithm THETA (or STABLESET) also calculates $\rho_w(G)$, thus, by definition, there exist integers $\lambda_i > 0$ which satisfy (7.1). Note that the numbers constructed by the algorithm of Theorem (2.6) (c) (plus scaling afterwards) need not be integral in general.

However, we can find such integers for perfect graphs. We first show how this can be done in the cardinality case, i.e., $w = (1, \ldots, 1)^T$.

Assume that $G = (V, E)$, $|V| \geq 2$, is a perfect graph and we want to find a clique cover of G.

(7.2) Algorithm. Cardinality clique cover.

(7.2.1) Apply the algorithm of Theorem (2.6) (c) to find cliques $C_i \subseteq V$ and (possibly nonintegral) rationals $\lambda_i > 0$, $i = 1, \ldots, t$ satisfying (7.1).
(We claim that every clique C_i with $\lambda_i > 0$ intersects every stable set of G of cardinality $\alpha(G)$. Suppose C_j is such a clique and $W \subseteq V$ is a maximum stable set with $W \cap C_j = \emptyset$. Then (7.1) implies

$$\alpha(G) = |W| = \sum_{v \in W} w_v = \sum_{v \in W} \sum_{i=1, v \in C_i}^{t} \lambda_i \leq \sum_{\substack{i=1 \\ i \neq j}}^{t} \lambda_i < \sum_{i=1}^{t} \lambda_i = \alpha(G),$$

a contradiction. Therefore we continue as follows.)

(7.2.2) Remove clique C_1 from G, i.e., set $G := G - C_1$. If $V(G) = \emptyset$. **stop.** Otherwise go to (7.2.1).

(Note that the graph G' obtained from G by removing clique C_1 satisfies $\alpha(G') = \alpha(G) - 1$ since every maximum stable set of G will lose one vertex. So after exactly $\alpha(G)$ executions of (7.2.1) and (7.2.2) we have found $\alpha(G)$ cliques which cover G. These cliques are those which have been removed in (7.2.2). Note also that every vertex of V is contained in exactly one such clique and that these cliques are not necessarily maximal cliques of G.) □

Since the algorithm of Theorem (2.6) (c) can be shown to be polynomial in $|V|$ the overall running time of algorithm (7.2) is also polynomial in $|V|$. We shall now extend this algorithm to the weighted case.

Assume that a perfect graph $G = (V, E)$ and a weight function $w : V \to \mathbb{Z}_+$ are given.

(7.3) Algorithm. Weighted clique cover.

(7.3.1) Apply the algorithm of Theorem (2.6) (c) to find cliques $C_i \subseteq V$ and (possibly nonintegral) rationals $\lambda_i > 0$, $i = 1, \ldots, t$, satisfying (7.1).

(7.3.2) Set $\lambda_i' := \lfloor \lambda_i \rfloor$, $i = 1, \ldots, t$,

$$w_v' := w_v - \sum_{i=1, v \in C_i}^{t} \lambda_i' \quad \text{for all } v \in V,$$

and construct the graph $G_{w'}$.
(Since $\lambda_i - \lambda_i' < 1$ we obtain from (7.1)

$$w'_v = \sum_{i=1,\, v\in C_i}^{t} (\lambda_i - \lambda'_i) < t \leqslant |V| \tag{7.4}$$

so the graph $G_{w'}$ obtained by replacing every node of v by w'_v nonadjacent copies and linking two nodes in $G_{w'}$ by an edge if their originals in G are adjacent has less than $|V|^2$ vertices. This implies that $G_{w'}$ can be constructed from G in time polynomial in $|V|$ and $\lceil \log \| w \|_\infty \rceil$.)

(7.3.3) Apply algorithm (7.2) to $G_{w'}$ to obtain cliques D'_i, $i = 1, \ldots, \alpha(G_{w'})$ covering each vertex of $G_{w'}$ exactly once.
(Since every clique D'_i of $G_{w'}$ contains at most one copy of every vertex $v \in V(G)$, every D'_i corresponds to a clique, say D_i, of G. Note that for $D'_i \neq D'_j$ the corresponding cliques D_i, D_j of G may be identical.)

(7.3.4) Construct the cliques $D_1, \ldots, D_{\alpha(G_{w'})}$ of G corresponding to the cliques $D'_1, \ldots, D'_{\alpha(G_{w'})}$ of $G_{w'}$. Let $B_1, \ldots, B_r$ be the different cliques occurring in the sequence D_i, $i = 1, \ldots, \alpha(G_{w'})$ and let μ_j, $j = 1, \ldots, r$, be the number of times clique B_j occurs in the sequence D_i, $i = 1, \ldots, \alpha(G_{w'})$. Then proceed as follows: Set $k = t$ and for $j = 1$ to r do; If B_j is equal to one of the cliques C_i, $i \in \{1, \ldots, t\}$, then set $\lambda'_i := \lambda'_i + \mu_j$. Otherwise set $k := k + 1$, $\lambda'_k := \mu_j$ and $C_k := B_j$. □

We claim that the cliques C_i and integers λ'_i, $i = 1, \ldots, s$, defined in step (7.3.4) solve the clique cover problem considered. Obviously, in the above algorithm every vertex $v \in V$ is covered $w_v - w'_v$ times after the execution of step (7.3.2). By applying algorithm (7.2) to the graph $G_{w'}$ and making the construction described in (7.3.4) every vertex will be covered a further w'_v times. So the cliques $C_1, \ldots, C_s$ and integers $\lambda'_1, \ldots, \lambda'_s$ satisfy

$$\sum_{i=1,\, v\in C_i}^{s} \lambda'_i = w_v \quad \text{for all } v \in V.$$

Similarly, note that $G_{w'}$ is designed in such a way that $\alpha(G_{w'}) = \alpha_w(G) - \sum_{i=1}^{t} (\lambda_i - \lfloor \lambda_i \rfloor)$ holds. Since G is perfect $G_{w'}$ is also perfect, so $\rho(G_{w'}) = \sum_{j=1}^{r} \mu_j = \alpha(G_{w'})$ which implies that the λ'_i defined in (7.3.4) satisfy

$$\sum_{i=1}^{s} \lambda'_i = \alpha_w(G) = \rho_w(G).$$

Thus algorithm (7.3) produces the desired solution of the clique cover problem for a perfect graph.

Since the algorithm STABLESET, the algorithm of Theorem (2.6) (c) and the algorithm (7.2) run in time polynomial in $|V(G)|$ and $\lceil \log \| w \|_\infty \rceil$ the overall

running time of algorithm (7.3) is polynomial in $|V(G)|$ and $\lceil \log \|w\|_\infty \rceil$ for every perfect graph G and every objective function $w : V(G) \to \mathbb{Z}_+$.

As before it is now easy to obtain a polynomial time algorithm for the weighted coloring problem for perfect graphs. Since the weighted chromatic number $\chi_w(G)$ equals the weighted clique cover number $\rho_w(\bar{G})$ of the complementary graph $\bar{G}$ we simply apply algorithm (7.3) to the perfect graph $\bar{G}$ which will yield the desired optimum weighted coloring of G.

8. Conclusions

In the previous section we have described polynomial time algorithms for various linear programming problems on perfect graphs. All these algorithms are based on the ellipsoid method and use a polynomial time separation algorithm for a convex, nonpolyhedral set. Although these algorithms are polynomial (and thus are theoretically good) we do not recommend them for practical use.

Just for curiosity we have done some computational experiments with the separation algorithm for $\bar{\mathcal{B}}(G)$ described in Section 5. As expected, the numerical problems were such that even for small problem sizes, say $|V(G)|$ equal to 10 or 20, it was almost impossible to obtain a correct answer. An alternative approach is to use (4.9) for the design of a polynomial algorithm to compute $\vartheta(G)$. This amounts to minimizing a convex function on an affine space. In principle, this can be done by the ellipsoid method in polynomial time. In practice, it is probably better to use some simpler descent method. The first experiments with this dual approach seem to be more promising.

Our analysis of these problems should be viewed as a theoretical contribution showing that certain programming problems for perfect graphs are indeed polynomially solvable. Future research should be directed toward finding practically good algorithms for these problems. These algorithms should have a more combinatorial nature and should not suffer from the numerical instability (due to our present-day computer technology of fixed precision arithmetic) of the ellipsoid method and the separation problem for $\bar{\mathcal{B}}(G)$.

References

[1] V. Chvátal, On certain polytopes associated with graphs, J. Comb. Theory, Ser. B 18 (1975) 138–154.

[2] P. Gács and L. Lovász, Khachian's algorithm for linear programming, Math. Program. Studies 14 (1981) 61–68.

[3] M.R. Garey and D.S. Johnson, Computers and Intractability, A Guide to the NP-Completeness (Freeman, San Francisco, 1979).
[4] M. Grötschel, L. Lovász and A. Schrijver, The ellipsoid method and its cons combinatorial optimization, Combinatorica 1 (1981) 169–197.
[5] W. Haemers, On some problems of Lovász concerning the Shannon capacity of a g Trans Inform. Theory IT-25 (1979) 231–232.
[6] W.-L. Hsu, How to color claw-free perfect graphs, Ann. Discrete Math. 11 (198
[7] W.-L. Hsu and G.L. Nemhauser, Algorithms for minimum coverings by cliques an cliques in claw-free perfect graphs, Discrete Math. 37 (1981) 181–191 (this volume, p
[8] L.G. Khachiyan, A polynomial algorithm in linear programming, Dokl. Akad. Nau (1979) 1093–1096 (English transl. Soviet Math. Dokl. 20 (1979) 191–194).
[9] L. Lovász, Normal hypergraphs and the perfect graph conjecture, Discrete Ma 253–267 (this volume, pp. 29–42).
[10] L. Lovász, On the Shannon capacity of a graph, IEEE Trans. Inform. Theory IT-25
[11] M.W. Padberg, Almost integral polyhedra related to certain combinatorial problems, Linear Algebra & Appl. 15 (1976) 69–88.
[12] C. Shannon, The zero error capacity of a noisy channel, IRE Trans. Inform. Theory 8–19.
[13] N.Z. Shor, Convergence rate of the gradient descent method with dilatation o Kibernetika 2 (1970) 80–85 (English transl. Cybernetics 6 (1970) 102–108).
[14] D.B. Yudin and A. S. Nemirovskii, Informational complexity and effective method for convex extremal problems, Ekonomika i Mat. Metody 12 (1976) 357–369 (En Matekon: Transl. of Russian and East European Math. Economics 13 (1976) 24