Les Cahiers du GERAD ISSN: 0711–2440

Polynomially Solvable Cases of the Constant Rank Unconstrained Quadratic 0-1 Programming Problem

E. Çela, B. Klinz,
C. Meyer

G–2006–36

May 2006

Polynomially Solvable Cases of the Constant Rank Unconstrained Quadratic 0-1 Programming Problem

Eranda Çela, Bettina Klinz
Institut für Optimierung und Diskrete Mathematik (Mathematik B)
Technische Universität Graz
Steyrergasse 30
A-8010 Graz, Austria
{cela; klinz}@opt.math.tu-graz.ac.at

Christophe Meyer
GERAD and Département d'informatique et de recherche opérationnelle
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec) Canada H3C 3J7
christophe.meyer@gerad.ca

May 2006

Les Cahiers du GERAD
G–2006–36

Abstract

In this paper we consider the constant rank unconstrained quadratic 0-1 optimization problem, CR-QP01 for short. This problem consists in minimizing the quadratic function $\langle x, Ax\rangle + \langle c, x\rangle$ over the set $\{0,1\}^n$ where c is a vector in $\mathbb{R}^n$ and A is a symmetric real $n \times n$ matrix of constant rank r.

We first present a pseudo-polynomial algorithm for solving the problem CR-QP01, which is known to be NP-hard already for $r = 1$. We then derive two new classes of special cases of the CR-QP01 which can be solved in polynomial time. These classes result from further restrictions on the matrix A. Finally we compare our algorithm with the algorithm of Allemand et al. (2001) for the CR-QP01 with negative semidefinite A and extend the range of applicability of the latter algorithm. It turns out that neither of the two algorithms dominates the other with respect to the class of instances which can be solved in polynomial time.

Key Words: Quadratic 0-1 programming, Special case, Local minima, Constant rank matrix, Stable sets.

Résumé

Dans cet article nous considérons le problème de minimisation quadratique 0-1 non-contraint avec une matrice de rang constant, noté CR-QP01. Ce problème consiste à minimiser la fonction quadratique $\langle x, Ax\rangle + \langle c, x\rangle$ sur l'ensemble $\{0,1\}^n$ où c est un vecteur de $\mathbb{R}^n$ et A est une matrice symétrique réelle de dimension $n \times n$ et de rang constant r.

Nous présentons d'abord un algorithme pseudo-polynomial pour résoudre le problème CR-QP01, qui est connu pour être NP-difficile déjà pour $r = 1$. Nous dérivons ensuite deux nouvelles classes de cas spéciaux de CR-QP01 qui peuvent être résolues en temps polynomial. Ces classes s'obtiennent en ajoutant des restrictions supplémentaires sur la matrice A. Finalement nous comparons notre algorithme avec l'algorithme de Allemand et al. (2001) pour CR-QP01 lorsque A est une matrice semi-définie négative et nous étendons le domaine d'application de ce dernier algorithme. Nous montrons qu'aucun des deux algorithmes ne domine l'autre par rapport aux classes d'instances qui peuvent être résolues en temps polynomial.

Mots clés : programmation quadratique 0-1, cas spécial, complexité, minimum local, matrice de rang constant.

Acknowledgments: This research has been supported by the Spezialforschungsbereich F 003 "Optimierung und Kontrolle", Projektbereich Diskrete Optimierung. This work was mainly done during a postdoctoral stay of the third author at the TU Graz, which he gratefully thanks for its hospitality. A first version of this paper was published jointly as a technical report in Graz (SFB Report 235, TU Graz, April 2002) and in Montréal (Technical Report CRT-2002-11, CRT - University of Montréal, April 2002).

1 Introduction

Problem statement. In this paper we consider a special case of the unconstrained 0-1 quadratic programming problem, QP01 for short. The QP01 can be stated as follows:

$$\min_{x\in\{0,1\}^n} \langle x, Ax\rangle + \langle c, x\rangle \tag{1}$$

where c is a vector in $\mathbb{R}^n$, A is a symmetric real $n \times n$ matrix and $\langle\cdot,\cdot\rangle$ denotes the Euclidean inner product in $\mathbb{R}^n$. Note that since $x_i^2 = x_i$ for $x_i \in \{0, 1\}$, one could assume in problem (1) that there is no linear term, i.e., that $c = 0$. Applying this transformation, however, changes the diagonal elements of A. Since this paper is concerned with special cases of the QP01 which result from specially structured matrices, we prefer to work with the representation (1). Problem QP01 has been investigated in numerous papers and has many applications, see e.g. Boros and Hammer (1991) and the references cited therein.

It is well-known that QP01 is strongly NP-hard; for example, it is equivalent to the maximum cut problem (MC) which is well-known to belong to the class of strongly NP-hard problems (for the equivalence see Boros and Hammer (1991), for the complexity of the MC problem see Garey and Johnson (1979)).

The topic of this paper is the *constant rank unconstrained quadratic 0-1 programming problem*, CR-QP01 for short, which arises as special case of the QP01 by restricting the matrix A to the class of matrices with constant rank r. This restriction remains NP-hard even for the special case of rank 1 matrices (for details, see Section 2).

Related results. In the literature mainly two types of special cases of the QP01 have been investigated. The first type typically results from putting restrictions on the graph $G(A)$ obtained by introducing an edge $\{i, j\}$ for $a_{ij} \neq 0$. There is a close relationship between this class of special cases of the QP01 and special cases of the maximum cut problem for special graph classes. An example of this first type of special cases is the case which results from graphs $G(A)$ with bounded treewidth. This special case can be solved in polynomial time (see Crama, Hansen and Jaumard (1990) for a treatment in the more general setting of pseudo-Boolean programs), and subsumes the special cases where the graph $G(A)$ is series-parallel (Barahona, 1986) or where $G(A)$ is a binary tree (Pardalos and Jha, 1991). There exist quite a number of other polynomially solvable special cases of the QP01 which result from restrictions on the graph $G(A)$. As this paper deals with a different class of special cases, we refrain from giving further details.

The second class of special cases arises from putting restrictions on the matrix $A = (a_{ij})$. The best known example of this type is the case of nonpositive matrices A, i.e., more precisely, $a_{ij} \leq 0$ for $i, j = 1, \ldots, n$, $i \neq j$. This case can be solved by reduction to a maximum flow problem in a network with $O(n^2)$ nodes (see Picard and Ratcliff, 1975). The CR-QP01 belongs to this second class of special cases. The following special cases of the CR-QP01 have been treated in the literature.

- A is of rank $r = 1$ and there is no linear term, i.e., $c = 0$. This case can be solved in a straightforward way by inspection.
- A has at most one positive and at most one negative eigenvalue and the rank of the matrix (A, c) is equal to the number of nonzero eigenvalues of A. In this case, the objective function f in (1) can be written as product of two linear functions. This special case of the CR-QP01 is still NP-hard, see Hammer et al. (2002). In Hammer et al. (2002) an $O(n \log n)$ algorithm is proposed for solving the continuous relaxation, and then cases are characterized where the optimal solution of the relaxation is 0-1, i.e., constitutes an optimal solution of the CR-QP01.
- A is negative semidefinite and there is no linear term, i.e., $c = 0$. For this case, Allemand, Fukuda, Liebling and Steiner (2001) proposed an algorithm of complexity $O(n^{r-1})$ for $r \geq 3$ and $O(n^2)$ for $r = 2$. At the end of this paper we will show that their algorithm actually solves a broader class of problems, namely all quadratic 0-1 problems with a matrix of rank r that have the property that all optimal solutions of the continuous relaxation are integral.

Our results. Our main result is the identification of the following two new classes of polynomially solvable cases of the CR-QP01:

(C1) This class results from a hypergraph $H = (V(H), E(H))$ with bounded edge size. We require that $\sum_{i,j \in F} a_{ij} < 0$ holds for all edges $F \in E(H)$ and that the stable sets of H can be enumerated in polynomial time (for details see Section 4).

(C2) This class results from an undirected loop-free graph $G = (V(G), E(G))$. We require that $a_{ii} + a_{jj} - 2|a_{ij}| < 0$ holds for all edges $\{i, j\} \in E(G)$ and that the stable sets of G can be enumerated in polynomial time (for details see Section 5).

Note that when H is a loop-free graph, the class C2 subsumes the class C1.

Note also that the class (C2) can also be defined in terms of the complement $G' = \overline{G}$ of the graph G:

(C2') This class results from an undirected loop-free graph $G' = (V(G'), E(G'))$. We require that

$$a_{ii} + a_{jj} - 2|a_{ij}| \geq 0 \quad \Rightarrow \quad \{i, j\} \in E(G') \qquad \text{for all } \{i, j\} \in V(G') \times V(G'), i \neq j$$

and that the cliques of G' can be enumerated in polynomial time.

A class of graphs that has recently attracted a lot of attention and for which the cliques can be enumerated in polynomial time is the class of bounded treewidth graphs. Indeed a characterization of graphs with treewidth at most k is the following: a graph G has treewidth at most k if and only if G is a subgraph of a chordal graph that has maximum clique size at most k, see Bodlaender (1998). Obviously the size of a maximum clique in G is also bounded by k, hence the number of cliques in G is $O\left(n^k\right)$. Since there exist algorithms that enumerate the (maximal) cliques in time polynomial in the total size of the cliques,

see, e.g., Tsukiyama, Ide, Aviyoshi and Shirakawa (1977), the cliques can be enumerated in polynomial time in graphs with bounded treewidth (although this is probably not the most efficient way to do: exploiting the tree decomposition of those graphs is likely to yield more efficient algorithms). A well-known subclass of the class of bounded treewidth graph is the class of series parallel graphs, which itself contains the trees. Another class of graphs having a polynomial number of cliques is the class of planar graphs, and more generally the class of graphs with bounded thickness. The thickness of a graph G is defined as the minimum number of planar graphs whose union gives G. The size of a maximum clique in a graph with thickness t is $\leq 6t-2$, see Beineke (1997).
By considering the complementary classes of these classes, we get classes of graphs for which the stable sets can be enumerated in polynomial time. Unfortunately, we are not aware of any non-trivial classes of hypergraphs for which all stable sets can be enumerated in polynomial time. Trivial classes of hypergraphs with this property include the subclasses of graphs we mentioned above (any graph is a special hypergraph) and complete p-hypergraphs (i.e., the hypergraph with all possible edges of size p) for a fixed number p since this class has $O(n^p)$ stable sets (namely all possible subsets of the vertex set with size $\leq p-1$).

Organization of the paper. The paper is organized as follows. In Section 2, we discuss the complexity of problem CR-QP01 and present a pseudopolynomial algorithm for its solution. In Section 3, we present the general framework of our approach. Section 4 deals with the special case C1 and Section 5 deals with the special case C2. In Section 6, we compare our approach with the approach of Allemand, Fukuda, Liebling and Steiner (2001). In particular, we show that the range of applicability of the approach of (Allemand, Fukuda, Liebling and Steiner, 2001) can be extended. We furthermore provide examples which show that neither of the two approaches dominates the other in terms of the classes of instances of the CR-QP01 that can be solved in polynomial time. The paper is closed with a short conclusion in Section 7.

2 Complexity aspects of the CR-QP01

In this section, we are going to investigate the complexity of the CR-QP01 in some more detail. In particular, we will present a pseudopolynomial time algorithm for CR-QP01. This shows that, in contrast to the general QP01, the special case CR-QP01 with a matrix A of constant rank is not NP-hard in the strong sense.

For the rest of the paper we will make use of the following alternative representation of problem CR-QP01:

$$\min_{x\in\{0,1\}^n} f(x) = \langle \widetilde{c}, x\rangle + \sum_{\ell=1}^{d} \lambda_\ell \left(\beta_\ell + \langle u^\ell, x\rangle\right)^2 \tag{2}$$

where d is a constant, $\widetilde{c}$ and $u^1,\ldots,u^d$ are given vectors in $\mathbb{R}^n$, and $\lambda_1,\ldots,\lambda_d$, $\beta_1,\ldots,\beta_d$ are given reals. Note that we could always set $\beta_\ell = 0$ for $\ell = 1,\ldots,d$, because all linear

terms can be collected in the term $\langle \widetilde{c}, x \rangle$ and additive constants do not play a role in the minimization of f. The reason why we, nevertheless, use the more general formulation is that the choice of the vector $\widetilde{c}$ and of the numbers β_ℓ might have an influence on the running times of our algorithms (for further details see the comments below).

From linear algebra it is known that any quadratic function can be always represented in the form (2). One method to arrive at such a representation is to determine a spectral decomposition of A, i.e., to use the non-zero eigenvalues of A as values λ_j and the corresponding eigenvectors as vectors u^j, $j = 1, \ldots, d$, where $d = r$ (recall that r denotes the rank of A). Moreover, all β_ℓ are set to zero. This approach has the disadvantage that it might lead to irrational numbers in the representation (2), even in the case where all entries of A and c are rational. If a rational representation is needed, one can compute a so-called LDU decomposition of A which leads in the symmetric case to a decomposition of A as product LDL^T where L is a lower triangular matrix and D is a diagonal matrix with rank $d = r$ (see textbooks on linear algebra, e.g. Gantmacher (1959), for details).

Since the representation of a quadratic function in the form (2) is not unique, this poses the question of finding the best such representation. Different representations can have different values for d and $\widetilde{c}$, which will influence the running time of our algorithms. For example, by choosing the numbers β_ℓ in a clever way, it might be possible to arrive at $\widetilde{c} = 0$, which, as we will see later, leads to algorithms with a faster running time for the classes considered in this paper. Similarly, a clever choice of $\widetilde{c}$ might allow to arrive at a quadratic part with rank $d < r$. We will not deal with the question of finding the representation which results in the smallest running times of our algorithms in this paper. This is a problem in its own right.

It is well-known and easy to see that problem CR-QP01 is NP-hard already for matrices of rank 1. If the representation (2) is used, one can even moreover assume that $\widetilde{c} = 0$. To see this, consider the well-known SUBSET SUM problem, see Garey and Johnson (1979), which, given nonnegative integers $s_1, \ldots, s_n$ and an integral target value B, asks for the existence of a subset $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} s_i = B$. This question has the answer yes if and only if the optimal value of the instance of the CR-QP01 given by $\min\limits_{x \in \{0,1\}^n} \left(\sum_{i=1}^n s_i x_i - B\right)^2$ is 0.

The following result shows that problem CR-QP01 can be solved in pseudo-polynomial time for rational data.

Proposition 1 *Let an instance of problem (2) be given with $u^1, \ldots, u^d \in \mathbb{Z}^n$, $\beta_1, \ldots, \beta_d \in \mathbb{Z}$, $\widetilde{c} \in \mathbb{Q}^n$ and $\lambda_1, \ldots, \lambda_d \in \mathbb{Q}$. Let $U = 2 \max_{i=1,\ldots,n;\ell=1,\ldots,d}\{|u_i^\ell|, |\widetilde{c}_i|\}$. Then the given instance can be solved in $O\left(dU^{2d+2}n^{2d+3}\right)$ time.*

To prove Proposition 1, we need the following lemma.

Lemma 1 *Let $K = \{k_{ij}\}$ be a $m \times n$ integral matrix, and b an integer vector of dimension m. The problem of deciding whether there exists a vector $x \in \{0,1\}^n$ such that $Kx = b$ can be solved in $O\left(mn^{m+1}\kappa^m\right)$ time where $\kappa = 2 \max_{i=1,\ldots,m, j=1,\ldots,n} \{|k_{ij}|\}$.*

Proof. The proof of this lemma is based on a modification of the dynamic programming approach of Papadimitriou (1981) for the integer linear feasibility problem. Let $k^{(i)}$ denote the i-th column of the matrix K, $i = 1, \ldots, n$. At the j-th stage of the dynamic program, we compute the set W_j of vectors w that can be written as $w = \sum_{i=1}^{j} x_i k^{(i)}$ with $x_i \in \{0,1\}$, $i = 1, \ldots, j$. The cardinality of the set W_j is bounded from above by $(j\kappa + 1)^m$. Hence the set W_n can be computed in $O\left(m \sum_{j=1}^{n-1} (j\kappa+1)^m\right) = O\left(mn^{m+1}\kappa^m\right)$ time. To answer the feasibility question, it suffices to check if the set W_n contains the vector b, which can also be done in $O\left(mn^{m+1}\kappa^m\right)$ time. ■

Proof of Proposition 1. For notational convenience, set $u^0 = \widetilde{c}$. Due to the definition of U we have, $-\frac{nU}{2} \leq \langle u^\ell, x\rangle \leq \frac{nU}{2}$ for all $\ell = 0, \ldots, d$ and $x \in \{0,1\}^n$. Let $v = (v_0, \ldots, v_d)$ be an integral vector in the box $\left[-\frac{nU}{2}, \frac{nU}{2}\right]^{d+1}$. We associate with v the following parametrized minimization problem

$$\begin{array}{ll} \min & g_{v_0,\ldots,v_d}(x) = v_0 + \sum_{\ell=1}^{d} \lambda_\ell \left(\beta_\ell + v_\ell\right)^2 \\ \text{s.t.} & \left\{ \begin{array}{ll} \langle u^\ell, x\rangle = v_\ell & \ell = 0, \ldots, d \\ x_i \in \{0,1\} & i = 1, \ldots, n. \end{array} \right. \end{array} \tag{3}$$

For each choice of v, the set of constraints of the corresponding problem (3) defines a feasibility problem which can be solved in $O\left(dn^{d+2}U^{d+1}\right)$ time applying the approach from Lemma 1. The optimal value of problem (2) is the minimum of $v_0 + \sum_{\ell=1}^{d} \lambda_\ell \left(\beta_\ell + v_\ell\right)^2$ over all vectors $v = (v_0, \ldots, v_d)$ which correspond to a feasible problem. There are $(nU+1)^{d+1} = O\left((nU)^{d+1}\right)$ vectors $(v_0, \ldots, v_d)$ to test, hence the claimed result follows. ■

3 General algorithmic framework

In this section we present the general framework of our approach. In the two subsequent sections we will discuss how polynomial time algorithms can be obtained for the special cases C1 and C2 introduced in the introduction.

In Section 3.1 we introduce some key notations for the generic algorithmic approach which will be sketched in Section 3.2. The generic algorithm consists of three steps. We propose two variants for performing the first step, which are presented in Sections 3.3 and 3.4, respectively. The second and third steps of the algorithm are addressed in Sections 3.5

and 3.6, respectively. In Section 3.7, we introduce some graph theoretical definitions that will be needed in the remainder of this paper.

3.1 Neighborhoods and local minima

A key notion needed in this section is the notion of a neighborhood. The function $\mathcal{N}$ which maps $x \in \{0,1\}^n$ to the set $\mathcal{N}(x) \subseteq \{0,1\}^n \setminus \{x\}$ is called a *neighborhood function*, or *neighborhood* for short. The members of the set $\mathcal{N}(x)$ are called *neighbors* of x. Note that we allow $\mathcal{N}(x) = \emptyset$, i.e., x has no neighbors.

$\tilde{x} \in \{0,1\}^n$ is said to be a *local minimum* of (2) with respect to the neighborhood function $\mathcal{N}$ if $f(\tilde{x}) \leq f(x)$ holds for all $x \in \mathcal{N}(\tilde{x})$. $\tilde{x} \in \{0,1\}^n$ is said to be a *global minimum* of (2) if $f(\tilde{x}) \leq f(x)$ holds for all $x \in \{0,1\}^n$.

In the sequel it will be more convenient to use the following alternative representation of neighborhood functions: For $x \in \{0,1\}^n$ and a subset F of $\{1,\ldots,n\}$, let x^F denote the vector which results from x by flipping the values of the components of x corresponding to indices in F, i.e.,

$$x_i^F = \begin{cases} 1 - x_i & \text{if } i \in F \\ x_i & \text{if } i \notin F \end{cases} \qquad i = 1,\ldots,n.$$

Clearly, to each neighborhood function $\mathcal{N}$, we can associate a set function $\mathcal{F}$ such that $\mathcal{N}(x) = \{x^F : F \in \mathcal{F}(x)\}$ holds for all $x \in \{0,1\}^n$. By a slight abuse of notation, we will in the following also refer to $\mathcal{F}$ as a neighborhood function. We denote by $\mathcal{G}$ the union of the sets $\mathcal{F}(x)$ over all $x \in \{0,1\}^n$, i.e., $\mathcal{G} = \bigcup_{x\in\{0,1\}^n} \mathcal{F}(x)$. We assume that the sets in $\mathcal{G}$ are ordered in some arbitrary way, say $\mathcal{G} = \{F_1, F_2, \ldots, F_g\}$ where $g = |\mathcal{G}|$.

Both specific neighborhood functions which will be used in this paper (in Sections 4 and 5, respectively) are symmetric, i.e., fulfill the property $x \in \mathcal{N}(x') \Leftrightarrow x' \in \mathcal{N}(x)$. Moreover we assume that there exists a constant p such that $|F| \leq p$ for all $F \in \mathcal{G}$. This assumption will be fulfilled in all cases considered in this paper.

We are now going to characterize local minima with respect to a given neighborhood function $\mathcal{F}$.

Proposition 2 *x is a local minimum with respect to the neighborhood function $\mathcal{F}$ if and only if the following property holds for all $F \in \mathcal{F}(x)$:*

$$\sum_{j\in F}(2x_j - 1)\left(\widetilde{c}_j + 2\sum_{\ell=1}^{d}\lambda_\ell u_j^\ell(\beta_\ell + \langle u^\ell, x\rangle)\right) - \sum_{i,j\in F}(2x_i - 1)(2x_j - 1)a_{ij} \leq 0. \qquad (4)$$

Proof. Compute the difference $\Delta = f(x) - f(x^F)$ and note that $\sum_{\ell=1}^{d}\lambda_\ell u_i^\ell u_j^\ell = a_{ij}$ for all $i, j = 1,\ldots,n$. It is then easy to see that the condition $\Delta \leq 0$ is equivalent to the condition (4). ■

We are now going to reformulate the condition (4). Our goal is to arrive at a polyhedral description. For each set $F \in \mathcal{G}$ we choose a value δ_F such that

$$\sum_{i,j \in F} (2x_i - 1)(2x_j - 1)a_{ij} \leq \delta_F \qquad \text{for all } x \in \{0,1\}^n. \tag{5}$$

Let $\delta = (\delta_{F_1}, \delta_{F_2}, \ldots, \delta_{F_g})$. To each $x \in \{0,1\}^n$ we associate a polyhedron $P_{x,\delta} \subseteq \mathbb{R}^d$ which contains all $y \in \mathbb{R}^d$ such that

$$2\sum_{\ell=1}^{d} \lambda_\ell \left(\sum_{j \in F} (2x_j - 1)u_j^\ell \right) y_\ell \quad \leq \quad \delta_F - \sum_{j \in F} (2x_j - 1)\widetilde{c}_j \qquad \text{for all } F \in \mathcal{F}(x). \tag{6}$$

Proposition 2 implies that for all local minima x with the property $\mathcal{F}(x) \neq \emptyset$ (i.e. x has at least one neighbor) the polyhedron $P_{x,\delta} \subseteq \mathbb{R}^d$ is nonempty (to see that, set $y_\ell := \beta_\ell + \langle u^\ell, x \rangle$ for $\ell = 1, \ldots, d$). In order to write the inequalities defining the polyhedron $P_{x,\delta}$ in a more succinct way, we introduce the terms $r_j(y)$ defined by

$$r_j(y) = \widetilde{c}_j + 2\sum_{\ell=1}^{d} \lambda_\ell u_j^\ell y_\ell \qquad j = 1, \ldots, n. \tag{7}$$

Then $P_{x,\delta}$ can be defined as the set of all $y \in \mathbb{R}^d$ such that for all $F \in \mathcal{F}(x)$ we have

$$\sum_{j \in F} (2x_j - 1)r_j(y) \leq \delta_F. \tag{8}$$

3.2 A generic algorithm

In this section we are going to propose a high-level description of a generic algorithm $\mathcal{A}$ to solve the CR-QP01, stated in the form (2). In subsequent parts of Section 3 we will give more details on how the different steps of the generic algorithm $\mathcal{A}$ can be performed. Further specializations result from the choice of specific neighborhood functions $\mathcal{F}$ in Sections 4 and 5, where the special cases C1 and C2 are treated. We note that algorithm $\mathcal{A}$ is inefficient for the general case of CR-QP01. In Sections 4 and 5, respectively, we will show how $\mathcal{A}$ turns into a polynomial time algorithm for the special cases C1 and C2, respectively.

Generic algorithm $\mathcal{A}$

1. Construct a set Y with the property that Y contains at least one point $y \in P_{x,\delta}$ for all local minima x with $\mathcal{F}(x) \neq \emptyset$.
2. For each $y \in Y$, construct a set $X(y)$ such that $X(y) \supseteq \{x \in \{0,1\}^n : y \in P_{x,\delta}\}$.

3. Compute $f(x)$ for all $x \in X(Y) = \bigcup_{y \in Y} X(y)$ and for all $x \in \{0,1\}^n$ such that $\mathcal{F}(x) = \emptyset$. Let x^* be a point with minimal objective function value among the tested points. Then x^* constitutes an optimal solution of problem CR-QP01.

Clearly the choice of δ has a strong influence on the effectiveness of algorithm $\mathcal{A}$. If δ is badly chosen, then the cardinality of the sets $X(y)$ will be too large to allow an efficient algorithm (recall that in the final step of $\mathcal{A}$ an exhaustive search is done over the union of all sets $X(y)$ for $y \in Y$). Observe that $\sum_{j \in F}(2x_j - 1)r_j(y) = -\sum_{j \in F}(2x_j^F - 1)r_j(y)$ holds for $F \in \mathcal{F}(x)$. This motivates to choose δ_F such that, if possible, not both x and its corresponding neighbor x^F fulfill the inequality (8). This will help in achieving our goal to keep the cardinalities of the sets $X(y)$ sufficiently small. In the following we will make use of the following two different strategies to reach this goal:

S1 Choose $\delta_F < 0$ for all $F \in \mathcal{G}$.

S2 In the case $\widetilde{c} = 0$, there exists the following alternative choice: Set $\delta_F = 0$ for all $F \in \mathcal{G}$. This choice leads to an algorithm with improved running time as we will see later on, but it makes only sense to apply it when $\sum_{j \in F}(2x_j - 1)r_j(y) \neq 0$ holds for a sufficiently large number of sets $F \in \mathcal{G}$ (for details see Section 3.4).

In Section 3.3, we show how to construct the set Y when strategy S1 is used. In Section 3.4, we show how to perturb the problem so that $\sum_{j \in F}(2x_j - 1)r_j(y) \neq 0$ holds for all $x \in \{0,1\}^n$ and a sufficiently large number of sets $F \in \mathcal{G}$. This enables the use of strategy S2. The construction of the set $X(y)$ is discussed in Section 3.5.

3.3 Construction of the set Y using strategy S1 to choose δ

In this section, strategy S1 will be applied to choose δ. Let Γ be the set of all constraints of type (8). Note that a constraint in (8) is defined by a subset $F \in \mathcal{G}$ and a choice for the values x_j, $j \in F$. Thus, we have $|\Gamma| \leq \sum_{j=1}^{p} \binom{n}{j} 2^j = O(n^p)$ (recall that we assume throughout that $|F| \leq p$ for all $F \in \mathcal{G}$). Suppose that the constraints in Γ are ordered, i.e., $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_{|\Gamma|}\}$.

We now construct a tree $\mathcal{T}$ as follows: A node of $\mathcal{T}$ at level h is characterized by h linearly independent constraints of type (8), say $\gamma_{i_1}, \ldots, \gamma_{i_h}$ where $i_1 < i_2 < \ldots < i_h$. The root of the tree (level 0) corresponds to an empty set of constraints. Given a node $N(\gamma_{i_1}, \ldots, \gamma_{i_h})$ at level h, its sons are the nodes $N(\gamma_{i_1}, \ldots, \gamma_{i_h}, \gamma_i)$ for all possible choices of i such that the following three properties are fulfilled: (i) $i > i_h$, (ii) the $h+1$ constraints $\gamma_{i_1}, \ldots, \gamma_{i_h}, \gamma_i$ are linearly independent and (iii) constraint γ_i is compatible with the constraints $\gamma_{i_1}, \ldots, \gamma_{i_h}$ with respect to the choice of the values of the variables x_j involved in these constraints. Clearly the maximal depth of the tree $\mathcal{T}$ is d. For each leaf of the tree, we compute a point of the system of equations associated to the leaf (these equations result if we require that the inequalities characterizing the leaf are all fulfilled with equality). Note that, if a leaf is at level d, this system of equations has a unique solution, which is not the case if the leaf is at a level $< d$. In the latter case

we simply choose one solution of the system of equations corresponding to the leaf under consideration. The points computed in this way constitute the set Y.

It remains to be argued that the set Y constructed above contains at least one point of each polyhedron $P_{x,\delta}$. Let x be fixed and consider a face f of the polyhedron $P_{x,\delta}$ with smallest dimension $d-k$ (if $P_{x,\delta}$ has extreme points, f will be an extreme point). This face f is characterized by k linearly independent constraints of type (8) which are satisfied at equality, say, $\gamma_{j_1}, \ldots, \gamma_{j_k}$ with $j_1 < j_2 < \cdots < j_k$. By definition, the tree $\mathcal{T}$ contains the node $N(\gamma_{j_1}, \ldots, \gamma_{j_k})$. If $N(\gamma_{j_1}, \ldots, \gamma_{j_k})$ is a leaf, then by construction of the algorithm, a point of the face f has been computed. If $N(\gamma_{j_1}, \ldots, \gamma_{j_k})$ is not a leaf, then it has a descendent $N(\gamma_{j_1}, \ldots, \gamma_{j_k}, \gamma_{j_{k+1}}, \ldots, \gamma_{j_t})$ which is a leaf: the point that was computed for this leaf is a point of our face f.

The number of leaves in the tree, and hence the cardinality of Y, is bounded by $\binom{|\Gamma|}{d}$ (observe that the number of leaves is largest if there are no leaves at levels $< d$). The amount of work that has to be done at each node (i.e., either checking that the inequalities of that node are linearly independent, or finding a point of the system) can be bounded by $O(d^3)$, hence the time complexity of computing Y is given by $O\left(\sum_{\ell=1}^{d} \binom{|\Gamma|}{\ell} d^3\right) = O(|\Gamma|^d d^3) = O(d^3 n^{pd})$.

3.4 Implicit construction of the set Y using strategy S2 to choose δ

Strategy S2 will be applied when $\widetilde{c} = 0$. Recall that this means that we set $\delta_F = 0$ for all $F \in \mathcal{G}$. In that case $P_{x,\delta}$ is a polyhedral cone with origin $\Omega = (0, \ldots, 0)$ for all $x \in \{0,1\}^n$. Note that the point Ω itself is not a useful point for inclusion into the set Y because it belongs to all $P_{x,\delta}$. ($\Omega \in Y$ would result in $X(Y) = \{0,1\}^n$, i.e., in an exhaustive search over all feasible solutions of CR-QP01). Instead we consider points that are close to Ω. These points are on extreme rays (or faces of greater dimension, if no extreme rays exist). Since these faces are of dimension ≥ 1, their number is $O(|\Gamma|^{d-1}) = O(n^{p(d-1)})$. This allows us to decrease the time complexity of the procedure for computing Y in comparison to the case of strategy S1 where $O(|\Gamma|^d)$ points had to be investigated. The price we have to pay for this improvement is that we have to cope with problems which result from degeneracy.

Each point $y \in Y$ results from a set of constraints of type (8) which have to be fulfilled at equality. If for a point y and for sets $F \in \mathcal{G}$ that were not used to define y, we have $\sum_{j \in F}(2x_j - 1)r_j(y) = 0$, the point y might not be much more useful than Ω. This means that we have to take care of degeneracy. To that end, a symbolic perturbation method, which is described next, will be applied.

3.4.1 A perturbation method

The perturbation method which we are going to propose is inspired by an approach described in the book by Edelsbrunner (1987, p. 185–191). Let q be the first prime greater than $d+1$ and set $\psi(j,\ell) = q^{d(j+1)-\ell}$. Consider the perturbed vectors $\widehat{u}^\ell$ defined by

$$\widehat{u}^\ell_j = u^\ell_j + \varepsilon^{\psi(j,\ell)} \quad \ell = 1, \ldots, d, \quad j = 1, \ldots, n$$

where ε is a small positive number. Note that this perturbation also affects the problem in (2). We are actually solving a perturbed version which is obtained by replacing the vectors u^ℓ by their perturbed versions $\widehat{u}^\ell$, $\ell = 1, \ldots, d$. Let $\widehat{P}_{x,0}$ be the perturbed version of $P_{x,0}$. The polyhedron $\widehat{P}_{x,0}$ contains all $y \in \mathbb{R}^d$ which fulfill

$$\sum_{j \in F} (2x_j - 1)\widehat{r}_j(y) \leq 0 \qquad \text{for all } F \in \mathcal{F}(x) \tag{9}$$

where $\widehat{r}_j(y) = 2\sum_{\ell=1}^{d} \lambda_\ell \widehat{u}_j^\ell y_\ell$ for $j = 1, \ldots, n$.

If we had to give a specific value to ε, this value would probably have to be exponentially small, which would threaten the polynomiality of our algorithm. It turns out, however, that we can perform Step 1 of algorithm $\mathcal{A}$ in a modified way such that it is not necessary to explicitly compute the candidate points $y \in Y$. This allows us to refrain from choosing a specific value for ε. The key observation is that it suffices to be able to determine the sign of the expressions on the left hand side of the inequalities (9) defining the perturbed polyhedron $\widehat{P}_{x,0}$. In Section 3.4.2, we explain how to construct the systems defining the candidate points $y \in Y$. In Section 3.4.3, we characterize the sets $F \in \mathcal{G}$ for which the expressions $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y)$ are non-zero. Section 3.4.4 explains how to determine the sign of the expressions $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y)$. Section 3.4.5 discusses when the perturbation method should be used.

3.4.2 Implicit construction of the set Y

Consider again the tree $\mathcal{T}$ introduced in Section 3.3. In the rest of Section 3 we will work with the perturbed problem. A given node of tree $\mathcal{T}$ at level h is thus characterized by a system of equations

$$\sum_{j \in F_{t_\mu}} (2x_j - 1)\widehat{r}_j(y) = 0 \qquad \mu = 1, \ldots, h \tag{10}$$

where $F_{t_\mu} \in \mathcal{G}$ for $\mu = 1, \ldots, h$ and the values x_j, $j \in \bigcup_{\mu=1,\ldots,h} F_{t_\mu}$, are given.

Recall that in the process of constructing the tree $\mathcal{T}$ described in Section 3.3 we repeatedly need to test a given set of inequalities of type (8) for linear independence. Moreover, the explicit construction of the set Y requires that a system of equations is solved. This approach cannot be followed if perturbation is used and no specific value of ε is chosen. In the following we will demonstrate how these difficulties can be circumvented.

Suppose we are given the system of equations (10). We associate with this system the following simplified system of equations in the new variables z_j:

$$\sum_{j \in F_{t_\mu}} z_j = 0 \qquad \mu = 1, \ldots, h. \tag{11}$$

Obviously, the linear dependency of these equations implies the linear dependency of the equations (10). We are going to show that for ε sufficiently small, the converse also holds. We first show this result for the case $h = d - 1$, i.e., for the greatest possible value of h. The general result will be dealt with in Corollary 1.

We start with discussing the case of a leaf at level $h = d-1$. We augment the system (11) by a normalization constraint of the form

$$\sum_{j \in F_{t_0}} \alpha_j z_j = 1 \tag{12}$$

where the set $F_{t_0} \subseteq \{1, \ldots, n\}$ and the coefficients $\alpha_j, j \in F_{t_0}$ are chosen such that the equations given by (11)–(12) are linearly independent (F_{t_0} does not need to belong to $\mathcal{G}$; a possible choice is $F_{t_0} = \{j_0\}$ where $j_0 \notin \bigcup_{\mu=1,\ldots,h} F_{t_\mu}$ and $\alpha_{j_0} = 1$, although this has the disadvantage to require the fixation of an additional variable x_{j_0}). Consider the system in the variables y_ℓ obtained by replacing z_j by $(2x_j - 1)\widehat{r}_j(y)$. We show now that for ε sufficiently small, this system has always a unique solution.

Proposition 3 *Assume that the d equations in the variables z_j given by (11)–(12) are linearly independent. Then for any choice of the values $x_j, j \in \bigcup_{\mu=0,\ldots,d-1} F_{t_\mu}$, and for ε sufficiently small, the system in the variables y_ℓ given by*

$$\begin{aligned} \sum_{j \in F_{t_\mu}} (2x_j - 1)\widehat{r}_j(y) &= 0 \qquad \mu = 1, \ldots, d-1 \\ \sum_{j \in F_{t_0}} \alpha_j (2x_j - 1)\widehat{r}_j(y) &= 1 \end{aligned}$$

has a unique solution.

Proof. Since the equations given by (11)–(12) in the variables z_j are linearly independent, this system of equations can be put in a triangular form, i.e., there exist numbers ν_{ik} for $i = 1, \ldots, d$, and $k = 1, \ldots, n$, satisfying $\nu_{ii} = 1$ for $i = 1, \ldots, d$ and numbers $b_i, i = 1, \ldots, d$, such that the system (11)–(12) is equivalent to $\sum_{k=i}^{n} \nu_{ik} z_{j_k} = b_i$ for $i = 1, \ldots, d$. The corresponding system in the variables y_ℓ has then the following form:

$$2 \sum_{\ell=1}^{d} \lambda_\ell \left(\sum_{k=i}^{n} \nu_{ik} (2x_{j_k} - 1) \left(u_{j_k}^\ell + \varepsilon^{\psi(j_k,\ell)} \right) \right) y_\ell = b_i, \qquad i = 1, \ldots, d.$$

Clearly the determinant of the coefficient matrix of this system is a polynomial in ε. This polynomial contains the term $2^d \left(\prod_{i=1}^{d} \lambda_i \nu_{ii} (2x_{j_i} - 1) \right) \varepsilon^{\sum_{i=1}^{d} \psi(j_i,i)}$ (observe that a cancellation of this term is not possible since due to the construction of the perturbation there cannot be another term with the same power of ε). Hence the polynomial contains at least one

non-zero term. Consequently, the determinant will be non-zero for ε sufficiently small, which implies the claim about the unique solvability. ■

The case where the level of the leaf under consideration is $h < d-1$ is reduced to the case $h = d-1$ by adding $d-1-h$ additional equations of the form (10) such that the equations $\sum_{j \in F_{t_\mu}} z_j = 0$, $\mu = 1, \ldots, d-1$ are linearly independent.(The sets F_{t_μ}, $\mu = h+1, \ldots, d-1$, corresponding to the newly added equations are not required to be members of the set $\mathcal{G}$.)

Corollary 1 *For ε sufficiently small, Equations (10) are linearly independent if and only if Equations (11) are linearly independent.*

Proof. As observed above, the linear independence of (10) clearly implies the linear independence of (11). The converse is a consequence of Proposition 3. ■

We are now prepared to summarize the procedure to construct the set Y implicitly. We again build up the tree $\mathcal{T}$ described in Section 3.3, but there are two essential differences. The first one relates to the fact that instead of computing the members of the set Y explicitly, we will work with systems of equations which define implicitly the points in Y. The second difference concerns the fact that the leaves of the tree have a depth of $\leq d-1$ (in contrast to $\leq d$ in Section 3.3). In the following we distinguish two cases: leaves at level $d-1$ and leaves at level $< d-1$. We start with the first case. A leaf at level $d-1$ is characterized by $d-1$ equalities of type (10). These equalities define a line L passing through the origin Ω, where Ω partitions L into two halflines. The addition of the normalization constraint $\sum_{j \in F_{t_0}} \alpha_j (2x_j - 1)\widehat{r}_j(y) = 1$ has the effect of selecting a point y_1^L lying on one of these two halflines. A point y_2^L on the other halfline is obtained by using the normalization constraint $\sum_{j \in F_{t_0}} \alpha_j (2x_j - 1)\widehat{r}_j(y) = -1$. We add both points y_1^L and y_2^L, defined implicitly by their system of equations, to the set Y.
The second case concerns leaves at level $h < d-1$. In such a case, we first add $d-1-h$ artificial constraints as explained above. We end up with $d-1$ equations which are linearly independent. These equations again define a line L through the origin, but in that case we only need to include in Y one point, for example y_1^L.

It remains to be argued that the set Y constructed above contains at least one point y of each polyhedral cone $\widehat{P}_{x,0}$. Let x be fixed and consider a face f of the polyhedron $\widehat{P}_{x,0}$ with smallest dimension $d-k$ (if Ω is an extreme point of $\widehat{P}_{x,0}$, then $f = \Omega$). This face f is characterized by k linearly independent constraints of type (9) which are satisfied at equality, say, $\gamma_{j_1}, \ldots, \gamma_{j_k}$ with $j_1 < j_2 < \ldots j_k$. We distinguish two cases, depending on whether $k = d$ or $k \leq d-1$. Consider first the case $k = d$. By definition, the tree $\mathcal{T}$ contains the leaf $N(\gamma_{j_1}, \ldots, \gamma_{j_{k-1}})$. A point on the halfline of $\widehat{P}_{x,0}$ defined by the equations $\gamma_{j_1}, \ldots, \gamma_{j_{k-1}}$ was included to the set Y (as well as a point on the other side of the supporting line with respect to Ω). Consider now the case $k \leq d-1$. By definition, the tree $\mathcal{T}$ contains the node $N(\gamma_{j_1}, \ldots, \gamma_{j_k})$. If $N(\gamma_{j_1}, \ldots, \gamma_{j_k})$ is a leaf, then at least one point of the face f was computed (two points if $k = d-1$). If $N(\gamma_{j_1}, \ldots, \gamma_{j_k})$ is not a leaf,

then it has a descendent $N(\gamma_{j_1}, \ldots, \gamma_{j_k}, \gamma_{j_{k+1}}, \ldots, \gamma_{j_t})$ which is a leaf: again at least one point of the face f was computed.

From the discussion above, it follows that the set Y, with which we end up, will indeed have cardinality $O(|\Gamma|^{d-1})$, in contrast to $O(|\Gamma|^d)$ in the case handled in Section 3.3. Recall, however, that the points of Y are defined implicitly. Now in Step 2 of algorithm $\mathcal{A}$, we need to determine for each point $y \in Y$ whether or not it belongs to $\widehat{P}_{x,0}$, i.e., whether or not (9) is satisfied. To that end, we need to be able to determine the sign of $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y)$ without computing explicitly the point y. We propose a method in Section 3.4.4, but first we will characterize the cases for which the sign is defined, i.e., for which $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y) \neq 0$.

3.4.3 Characterizations of sets F with $\sum_{\mathbf{j \in F}}(\mathbf{2x_j - 1})\widehat{\mathbf{r}}_\mathbf{j}(\mathbf{y}) \neq 0$

In this section we show that the proposed perturbation method eliminates the problems caused by degeneracy. Specifically, the perturbation guarantees that the number of sets $F \in \mathcal{G}$ for which $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y) = 0$ holds is sufficiently small. (Recall that this property is required to end up with a set $X(Y)$ of manageable size.)

The following two results characterize the sets F with the desired property $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y) \neq 0$.

Proposition 4 *Let $\tilde{y}$ be a point of Y which is implicitly defined by the system of equations $\sum_{j \in F_{t_\mu}}(2x_j - 1)\widehat{r}_j(y) = 0$ for $\mu = 1, \ldots, d-1$ augmented by a normalization constraint which is not listed here. Let $F_{t_d} \subseteq \{1, \ldots, n\}$ (not necessarily in $\mathcal{G}$). If the equations $\sum_{j \in F_{t_\mu}} z_j = 0$, $\mu = 1, \ldots, d$, are linearly independent, then we have $\sum_{j \in F_{t_d}}(2x_j - 1)\widehat{r}_j(\tilde{y}) \neq 0$.*

Proof. The proof is similar to that of Proposition 3. We first write the system $\sum_{j \in F_{t_\mu}} z_j = 0$, $\mu = 1, \ldots, d$, as a triangular system, then replace the variables z_j by $2(2x_j - 1)\sum_{\ell=1}^{d} \lambda_\ell \left(u_j^\ell + \varepsilon^{\psi(j,\ell)}\right) y_\ell$ and consider the determinant of the resulting system in the variables y_ℓ. This determinant is again a polynomial in ε with at least one non-zero term, hence the determinant is non-zero for ε sufficiently small. Therefore the only solution of the system of d equations is the point $\Omega = (0, \ldots, 0)$. Since $\tilde{y}$ is not equal to Ω and satisfies the first $d-1$ equations, $\tilde{y}$ cannot satisfy the last equation given by $\sum_{j \in F_{t_d}}(2x_j - 1)\widehat{r}_j(\tilde{y}) = 0$ which proves the claim. ■

Recall that when determining a point y, we had to fix the values of some of the x_j. Denote by J_y the set of indices $j \in \{1, \ldots, n\}$ with the property that the values x_j have not been used to define y.

Corollary 2 *For all $F \subseteq J_y$ and all $x \in \{0,1\}^n$, we have $\sum_{j \in F}(2x_j - 1)\widehat{r}_j(y) \neq 0$.*

Proof. The assumption $F \subseteq J_y$ implies that the equation $\sum_{j\in F} z_j = 0$ only involves variables z_j with indices j which are not appearing in the equations used to determine y. Consequently Proposition 4 can be applied, which yields the desired result. ■

3.4.4 Determination of the sign of $\sum_{\mathbf{j}\in\mathbf{F}}(\mathbf{2x_j-1})\widehat{\mathbf{r}}_\mathbf{j}(\mathbf{y})$

Assume that the point $y \in Y$ is implicitly defined by the system

$$\sum_{\ell=1}^{d} \lambda_\ell \left(\sum_{j\in F_{t_\mu}} (2x_j - 1) \left(u_j^\ell + \varepsilon^{\psi(j,\ell)} \right) \right) y_\ell = 0 \qquad \mu = 1, \ldots, d-1 \tag{13}$$

$$\sum_{\ell=1}^{d} \lambda_\ell \left(\sum_{j\in F_{t_0}} \alpha_j (2x_j - 1) \left(u_j^\ell + \varepsilon^{\psi(j,\ell)} \right) \right) y_\ell = 1. \tag{14}$$

Let $F \subseteq J_y$. We now explain how to determine the sign of $\sum_{j\in F}(2x_j - 1)\widehat{r}_j(y)$.

To simplify the notation, we set $F_{t_d} = F$ (recall, however, that F does not necessarily belong to $\mathcal{G}$). By Cramer's Rule, we have $y_\ell = \frac{\det M_\ell(\varepsilon)}{\det M(\varepsilon)}$ for $\ell = 1, \ldots, d$, where $M(\varepsilon)$ denotes the coefficient matrix of the system of equations given by (13)–(14) and $M_\ell(\varepsilon)$ denotes the matrix obtained from $M(\varepsilon)$ by replacing the ℓ-th column of $M(\varepsilon)$ by the column vector $(0, \ldots, 0, 1)^t$. Note that $\det M(\varepsilon)$ is non-zero by Proposition 3. Let $\eta'_{k\ell} = 2\lambda_\ell \left(\sum_{j\in F_{t_k}} (2x_j - 1) \left(u_j^\ell + \varepsilon^{\psi(j,\ell)} \right) \right)$ for $k = 0, \ldots, d$ and $\ell = 1, \ldots, d$.

We are interested in the sign of the expression $\sum_{j\in F_{t_d}}(2x_j - 1)\widehat{r}_j(y)$. This expression is equal to

$$\Psi = \sum_{j\in F_{t_d}} (2x_j - 1)\widehat{r}_j(y) = \sum_{\ell=1}^{d} \eta'_{d\ell} y_\ell = \sum_{\ell=1}^{d} \eta'_{d\ell} \frac{\det M_\ell(\varepsilon)}{\det M(\varepsilon)}.$$

We now develop the determinant of the matrix $M_\ell(\varepsilon)$ with respect to its ℓ-th column. This leads to $\det M_\ell(\varepsilon) = (-1)^{d+\ell} \det M''_\ell(\varepsilon)$ where $M''_\ell(\varepsilon)$ is the matrix obtained from $M_\ell(\varepsilon)$ by deleting the ℓ-th column and the last row. Hence

$$\Psi = \sum_{\ell=1}^{d} (-1)^{d+\ell} \eta'_{d\ell} \frac{\det M''_\ell(\varepsilon)}{\det M(\varepsilon)} = \frac{\det M''(\varepsilon)}{\det M(\varepsilon)}$$

where $M''(\varepsilon)$ is the matrix with elements $\eta'_{k\ell}$, $k, \ell = 1, \ldots, d$. Proceeding in a similar way as in the proof of Proposition 3, it can be shown that $\det M''(\varepsilon)$ is non-zero for ε sufficiently small. Since both $\det M(\varepsilon)$ and $\det M''(\varepsilon)$ are polynomials in ε, their sign is determined by the sign of their first non-zero coefficient, starting with the terms of smallest exponent. We explain in the following how to determine the sign of $\det M''(\varepsilon)$. The case

of $\det M(\varepsilon)$ is handled analogously. The exponents of ε in $\det M''(\varepsilon)$ are of the form $\sum_{(j,\ell)\in S} \psi(j,\ell) = \sum_{(j,\ell)\in S} q^{jd+d-\ell}$ for all subsets S of $\left(\bigcup_{\mu=1,\ldots,d} F_{t_\mu}\right) \times \{1,\ldots,d\}$ with cardinality $\leq d$. Since p is an upper bound on $|F|$ for all $F \in \mathcal{G}$, it follows that j can take at most pd distinct values which implies that the expression $\phi(j,\ell) = (j+1)d-\ell$, $(j,\ell) \in S$, can take at most pd^2 distinct values. The number of possible values for the exponents of ε is therefore bounded by $\sum_{k=1}^{d} \binom{pd^2}{k}$. For each possible exponent ω of ε, the coefficient of ε^ω is the sum of at most d subdeterminants of $M''(\varepsilon)$, and can thus be computed in $O(d^4)$ time. Since d is a constant, the sign of $\sum_{j\in F}(2x_j - 1)\widehat{r}_j(y)$ can therefore be computed in $O(1)$ time (note, however, that the constants hidden in this asymptotic notation increase rapidly with d and p).

3.4.5 Applicability of the perturbation method

In order to be able to apply the perturbation method, we must have

$$\sum_{i,j\in F} \widehat{a}_{ij} \leq 0 \qquad \text{for all } F \in \mathcal{G} \tag{15}$$

for ε sufficiently small, where $\widehat{A}$ is the perturbed matrix defined by $\widehat{a}_{ij} = \sum_{\ell=1}^{d} \lambda_\ell \widehat{u}_i^\ell \widehat{u}_j^\ell$ for all i, j. (Note that if an inequality in (15) were violated, it would not be possible to choose $\delta = 0$.) The condition (15) is guaranteed to hold for small values of ε, only when $\sum_{i,j\in F} a_{ij} < 0$ holds for all $F \in \mathcal{G}$.

We close the discussion of the perturbation approach by the remark that in principle this approach can also be applied in the general case, i.e., for $\widetilde{c} \neq 0$. The candidate set Y can be computed in $O(|\Gamma|^{d-1})$ time for the case in which $\widetilde{c} \neq 0$ and the number of indices i such that $\widetilde{c}_i \neq 0$ is bounded by a constant. In that case, we distinguish between polyhedra $P_{x,\delta}$ that are polyhedral cones vertexed at Ω, and polyhedra that contain at least one extreme point different from Ω. For polyhedral cones, we compute (implicitly) points on faces of dimension ≥ 1. For polyhedra with at least one extreme point different from Ω, we compute candidate extreme points that satisfy at equality at least one inequality corresponding to a set F containing an index i such that $\widetilde{c}_i \neq 0$. Since the number of these inequalities is bounded by a constant, these candidate extreme points can also be computed in $O(|\Gamma|^{d-1})$ time.

Although the perturbation method can also be applied in the case $\widetilde{c} \neq 0$, it is not recommendable to apply it for at least 3 reasons: removing the degeneracy results in an increase of the cardinality of Y; the hidden constants in the complexity bound increase when perturbation is used; and finally, perturbation may destroy a possible symmetry in the objective function, implying that algorithm $\mathcal{A}$ cannot any longer be used to obtain the set of *all* optimal solutions.

In the remaining part of this paper, we generally assume that the perturbation method is used if and only if $\widetilde{c} = 0$.

3.5 Construction of the set $\mathbf{X}(y)$

In order to be able to handle the cases with and without application of the perturbation method in a unified way, we introduce the expressions $\rho_j(y)$ for $j = 1, \ldots, n$ and $y \in Y$, where $\rho_j(y)$ equals $\widehat{r}_j(y)$ if perturbation is used and equals $r_j(y)$ otherwise.

Let $y \in Y$ be given explicitly or implicitly (by its set of defining equations). In order to construct the set $X(y)$ we need to compute the set of all $x \in \{0,1\}^n$ such that $y \in P_{x,\delta}$ (or its perturbed version $\widehat{P}_{x,0}$). This task amounts to finding all points $x \in \{0,1\}^n$ which satisfy

$$\sum_{j \in F} (2x_j - 1)\rho_j(y) \leq \delta_F \qquad \text{for all } F \in \mathcal{F}(x)$$

where $\delta_F = 0$ for all $F \in \mathcal{G}$ in the perturbed case.

Our method for solving this task is largely dependent on the choice of the neighborhood function $\mathcal{F}$. We therefore postpone the further discussion of the computation of the sets $X(y)$ to Sections 4 and 5, where specific neighborhood functions for the cases C1 and C2 will be introduced.

3.6 Construction of the set of optimal solutions

Let $X(Y) = \left(\bigcup_{y \in Y} X(y) \right)$ and $X = X(Y) \cup \{x : \mathcal{F}(x) = \emptyset\}$: the optimal solutions to problem (2) are obtained by evaluating the objective function f for all points of X, and keeping the points of smallest value. The complexity of this phase is $O(|X|nd)$.

Note that we can also construct the set of all local minima for the neighborhood function $\mathcal{F}$ under consideration by testing all points in X and listing those which are local minima. The running time of this approach is $|X|$ times the time needed to check if a given point is a local minimum with respect to $\mathcal{F}$.

A word of caution is in order when perturbations are used. We then have no guarantee to obtain all global optima, or all local minima, and will in general have to be satisfied with a single global optimum.

3.7 Some graph theoretical definitions

The following definitions will be needed in the remainder of the paper (for further details see e.g. Berge, 1976).

A *hypergraph* $H = (V(H), E(H))$ is defined by a set $V(H)$ of vertices and a collection $E(H)$ of subsets of $V(H)$ called *edges*. For notational convenience assume $V(H) = \{1, \ldots, n\}$. The *size* of an edge $F \in E(H)$ is the cardinality of F, i.e., $|F|$. An edge of size 1 is called a *loop*. Observe that a hypergraph H becomes a graph if all edges are of size 1

or 2. A hypergraph is said to be *of bounded edge size* if there exists a constant k such that $|F| \leq k$ for all $F \in E(H)$.

Let $H = (V(H), E(H))$ be a hypergraph and let $W \subseteq V(H)$. The set W induces a subhypergraph of H, the so-called *induced subhypergraph* $H[W] = (W, E(H[W]))$ with vertex set W and edge set $E(H[W])$ which only contains those edges $F \in E(H)$ which are subsets of W. As a special case the notion of a *induced subgraph* arises: let $G = (V(G), E(G))$ be an undirected graph and $W \subseteq V(G)$, then $G[W] = (W, E(G[W]))$ with $E(G[W]) = \{\{i, j\} \in E(G) : i, j \in W\}$ is called the subgraph of G induced by the vertex set W.

A *stable set* or *independent set* of $H = (V(H), E(H))$ is a subset W of $V(H)$ such that no subset of W belongs to $E(H)$. Note that a subset of a stable set is still a stable set.

4 A polynomial time algorithm for special case C1

In this section, we consider the special case C1 of problem CR-QP01 (cf. Section 1). This case arises for matrices A of rank d which additionally satisfy the following property

$$\sum_{i,j \in F} a_{ij} < 0 \qquad \text{for all } F \in E(H) \tag{16}$$

where $H = (V(H), E(H))$ is a hypergraph with $|V(H)| = n$. Our main result is the following:

Theorem 1 *Let $\mathcal{H}$ be a class of hypergraphs satisfying the following conditions: for any $H \in \mathcal{H}$,*

(a) H is a hypergraph of bounded edge size.
(b) The largest stable set in H is of size $O(\log n)$.
(c) The number of maximal stable sets in H is polynomial in n.

Then the CR-QP01 stated in the form (2) can be solved in polynomial time when restricted to the class of matrices fulfilling property (16) with $H \in \mathcal{H}$.

Theorem 1 will be proved in the course of this section.

4.1 Definition of the neighborhood function used for case C1

For dealing with case C1, we need a neighborhood function $\mathcal{F}$. To define $\mathcal{F}$ we proceed as follows. Let $H = (V(H), E(H))$ be a hypergraph and let $x \in \{0,1\}^n$. Let $H_0^x = H[V_0^x]$ and $H_1^x = H[V_1^x]$, respectively, denote the subhypergraphs of H which are induced by the vertex sets V_0^x and V_1^x, respectively, where $V_0^x = \{i \in V(H) : x_i = 0\}$ and $V_1^x = \{i \in V(H) : x_i = 1\}$. To each $x \in \{0,1\}^n$ we associate the set $\mathcal{F}(x)$ which defines the neighbors of x by taking $\mathcal{F}(x)$ to be the union of the edges of the subhypergraphs H_0^x and H_1^x. In

other words, $x' \in \{0,1\}^n$ is a neighbor of x if it can be obtained from x by selecting an edge $F \in E(H)$ such that the components x_i, $i \in F$, have the same value and then flipping the value of these components. Observe that $\mathcal{G} = \bigcup_{x\in\{0,1\}^n} \mathcal{F}(x) = E(H)$.

Let δ be chosen either according to strategy S1, i.e., such that $\sum_{i,j\in F} a_{ij} \leq \delta_F < 0$ holds for all $F \in \mathcal{G}$, or according to strategy S2, i.e., $\delta_F = 0$ for all $F \in \mathcal{G}$ (this strategy is applied for $\widetilde{c} = 0$).

Using the neighborhood function $\mathcal{F}$ introduced above, the defining inequalities (8) of the polyhedron $P_{x,\delta}$ (or the inequalities (9) defining its perturbed version) simplify to

$$\sum_{j\in F} \rho_j(y) \leq \delta_F \qquad \text{for all } F \in \mathcal{F}(x) \text{ such that } \; x_i = 1 \text{ for all } i \in F \tag{17}$$

$$-\sum_{j\in F} \rho_j(y) \leq \delta_F \qquad \text{for all } F \in \mathcal{F}(x) \text{ such that } \; x_i = 0 \text{ for all } i \in F \tag{18}$$

where again $\rho_j(y)$ equals $\widehat{r}_j(y)$ or $r_j(y)$ depending on whether or not perturbation has been applied (cf. Section 3.5).

4.2 Construction of the set $\mathbf{X}(y)$ for case C1

We assume that the set Y has already been computed (either explicitly or implicitly, see Sections 3.3 and 3.4 respectively). In order to compute the set $X(y)$ for a given $y \in Y$, we need to consider all points $x \in \{0,1\}^n$ such that $y \in P_{x,\delta}$ (cf. Section 3.2), which in our case means the set of all $x \in \{0,1\}^n$ such that the system of inequalities given by (17)–(18) is satisfied. Recall that in the course of computing y, the values of some x_j have already been fixed to either 0 or 1. Let J_y again denote the set of the indices $j \in \{1,\ldots,n\}$ for which the value of x_j has not yet been fixed. For $j \in \{1,\ldots,n\} \setminus J_y$, let x_j^y denote the already fixed value of the j-th component of x. Clearly, we do not have any freedom in choosing the values x_j^y. Thus the task of computing the set $X(y)$ amounts to finding all possibilities for choosing the values of x_j for $j \in J_y$ such that y belongs to $P_{x,\delta}$.

Let $H[J_y]$ denote the subhypergraph of H which is induced by the vertex set $J_y \subseteq V(H)$. If $x \in \{0,1\}^n$ satisfies the system of inequalities (17)–(18), then it also satisfies the following set of conditions

$$\sum_{i\in F} \rho_i(y) \leq 0 \qquad \text{for all } F \in E(H[J_y]) \text{ such that } x_i = 1 \text{ for all } i \in F \tag{19}$$

$$\sum_{i\in F} \rho_i(y) > 0 \qquad \text{for all } F \in E(H[J_y]) \text{ such that } x_i = 0 \text{ for all } i \in F. \tag{20}$$

To prove this claim, we distinguish two cases depending on which strategy has been used to choose δ. If S1 has been applied, we have $\delta_F < 0$ for all $F \in \mathcal{G}$, so the claim follows directly from (17)–(18). If S2 has been applied, the strict inequality in (20) follows from Corollary 2.

Consequently, we reduce the task to compute $X(y)$ to the search for all partitions (O_y, Z_y) of the set J_y such that

$$\sum_{i\in F} \rho_i(y) \leq 0 \text{ for all } F \in E(H[O_y]) \quad \text{and} \quad \sum_{i\in F} \rho_i(y) > 0 \text{ for all } F \in E(H[Z_y]). \tag{21}$$

Such partitions will be called *feasible partitions* of J_y. Each feasible partition leads to a point $x \in \{0,1\}^n$ in the following way:

$$x_j = \begin{cases} 0 & \text{for } j \in Z_y \\ 1 & \text{for } j \in O_y \\ x_j^y & \text{for } j \in \{1,\ldots,n\} \setminus J_y. \end{cases} \tag{22}$$

(The names O_y and Z_y have been chosen to reflect that x_j is set to one for $j \in O_y$, and to zero for $j \in Z_y$.)

It is easy to see that the set of feasible partitions is nonempty since the partition $(\widetilde{O}_y, \widetilde{Z}_y)$ with $\widetilde{O}_y = \{i \in J_y : \rho_i(y) \leq 0\}$ and $\widetilde{Z}_y = \{i \in J_y : \rho_i(y) > 0\}$ is clearly feasible (note that if perturbation is used, we have to use the technique described in Section 3.4.4 to determine the sign of $\rho_i(y)$).

Our problem now is to find all feasible partitions of J_y. The following lemma turns out to be helpful in solving this problem.

Lemma 2 *Let $(\widetilde{O}_y, \widetilde{Z}_y)$ be the initial feasible partition defined above and let (O_y, Z_y) be an arbitrary partition of J_y. Then for (O_y, Z_y) to be a feasible partition, the following two conditions have to be fulfilled*

(i) $U_y^{0\to 1} = \widetilde{Z}_y \cap O_y$ *is a stable set in the induced hypergraph* $H[\widetilde{Z}_y]$.

(ii) $U_y^{1\to 0} = \widetilde{O}_y \cap Z_y$ *is a stable set in the induced hypergraph* $H[\widetilde{O}_y]$.

Proof. We prove the statement in (i). The statement in (ii) is proved analogously. Assume that $U_y^{0\to 1}$ is not a stable set in $H[\widetilde{Z}_y]$, i.e., it contains an edge F of the hypergraph $H[\widetilde{Z}_y]$. Then by the feasibility of $(\widetilde{O}_y, \widetilde{Z}_y)$ it follows that $\sum_{i\in F} \rho_i(y) > 0$. Therefore, we cannot have $\sum_{i\in F} \rho_i(y) \leq 0$, which shows that (O_y, Z_y) cannot be a feasible partition since the first condition in (21) would be violated (note that $U_y^{0\to 1} \subseteq O_y$). We thus arrived at a contradiction which implies the claim (i). ■

Lemma 2 and the discussion above motivate the following approach for computing a set $X(y)$ containing the set of $x \in \{0,1\}^n$ such that $y \in P_{x,\delta}$:

Algorithm $\mathcal{B}$ to compute $\mathbf{X}(y)$:

1. Compute the initial feasible partition $(\widetilde{O}_y, \widetilde{Z}_y)$. Compute the point $\tilde{x}$ associated with $(\tilde{O}_y, \tilde{Z}_y)$ according to (22). Add $\tilde{x}$ to $X(y)$.

2. Enumerate the sets $\mathcal{S}(H[\widetilde{Z}_y])$ and $\mathcal{S}(H[\widetilde{O}_y])$ which denote the sets of all stable sets in the induced hypergraphs $H[\widetilde{Z}_y]$ and $H[\widetilde{O}_y]$ respectively. (Note that $\mathcal{S}(H[\widetilde{Z}_y])$ and $\mathcal{S}(H[\widetilde{O}_y])$ are subsets of the set of all stable sets of the hypergraph H.)
3. With each $(S_0, S_1) \in \mathcal{S}(H[\widetilde{Z}_y]) \times \mathcal{S}(H[\widetilde{O}_y])$, we associate the new partition (O_y, Z_y) with $O_y = \widetilde{O}_y \cup (S_0 \setminus S_1)$ and $Z_y = \widetilde{Z}_y \cup (S_1 \setminus S_0)$. Compute the point x associated with (O_y, Z_y) according to (22). Add x to $X(y)$.

Note that the running time of algorithm $\mathcal{B}$ depends heavily on the time needed by the second step in which all stable sets of two subhypergraphs of H need to be enumerated. When H is a graph, a large number of papers are available which present algorithms for listing all its maximal stable sets in time polynomial in the size of the output, see, e.g., Bron and Kerbosch, 1973; Chiba and Nishizeki, 1985; Dahlhaus and Karpinski, 1988; Johnson, Yannakakis and Papadimitriou, 1988; Mulligan and Corneil, 1972; Tsukiyama, Ide, Ariyoshi and Shirakawa, 1977. For the general case of a hyperplane of bounded edge size, Eiter and Gottlob (1995) have proposed an algorithm which lists all maximal stable sets in time polynomial in the size of the output (the existence of such an algorithm for general hypergraphs is an open question; see Boros *et al.* (2004) for some step in this direction). The approach of Eiter and Gottlob (1995) can be applied in our case, but in order to arrive at a polynomial overall running time for the procedure for computing $X(y)$, we need to make sure that the size of the output depends polynomially on the size of the input. This leads to the following sufficient condition for the polynomiality of algorithm $\mathcal{B}$.

Condition 1 *The sum of the cardinality of all stable sets of H is polynomial in n.*

We now show how this condition relates to the conditions of Theorem 1. Since the sum of the cardinalities of all subsets of a stable set of cardinality m is given by $\sum_{k=1}^{m} \binom{m}{k} k = m2^{m-1}$, the sum of the cardinalities of all stable sets is bounded by $\sigma(N, \overline{m}) = N\overline{m}2^{\overline{m}-1}$, where N denotes the number of maximal stable sets and $\overline{m}$ denotes the size of the largest maximal stable set. The number $\sigma(N, \overline{m})$ is bounded from above by a polynomial in n provided that N is a polynomial in n and $\overline{m} = O(\log n)$. This shows that Condition 1 is implied by the conditions of Theorem 1.

4.3 Enumeration of the points of empty neighborhood for class C1

Now the proof of Theorem 1 is almost completed. We have already argued in Sections 3.3 and 3.4 that the computation of the set Y, i.e., the first step of the generic algorithm $\mathcal{A}$ presented in Section 3.2, can be implemented to run in polynomial time. In Section 4.2, we showed that the second step of algorithm $\mathcal{A}$ can also be implemented in polynomial time. It remains to discuss the complexity of the third and last step of algorithm $\mathcal{A}$. To arrive at an overall polynomial time algorithm we need to make sure that the set of points x such that $\mathcal{F}(x) = \emptyset$ can be constructed in polynomial time. Observe that $\mathcal{F}(x) = \emptyset$ for some $x \in \{0,1\}^n$ if and only if $V(H)$, the vertex set of H can be partitioned into two stable sets. Moreover, each such pair of stable sets gives rise to two points x such that $\mathcal{F}(x) = \emptyset$.

Consequently Condition 1 also ensures that the set of points x such that $\mathcal{F}(x) = \emptyset$ can be constructed in polynomial time.

5 A polynomial time algorithm for special case C2

In this section we are going to deal with the special case C2 of problem CR-QP01 (cf. Section 1). Recall that the special case C2 arises for matrices A of rank d which additionally satisfy the following conditions:

$$a_{ii} + a_{jj} - 2|a_{ij}| < 0 \qquad \text{for all } \{i,j\} \in E(G) \tag{23}$$

where $G = (V(G), E(G))$ is an undirected graph with n vertices and without loops.

The main result of this section is the following:

Theorem 2 *Let $\mathcal{G}$ be a class of graphs satisfying the condition that the number of stable sets in the graph G is polynomial in n for all $G \in \mathcal{G}$. Then the CR-QP01 stated in the form (2) can be solved in polynomial time when restricted to the class of matrices fulfilling property (23) for $G \in \mathcal{G}$.*

We rewrite condition (23) as follows:

$$a_{ii} + a_{jj} - 2a_{ij} < 0 \qquad \text{for all } \{i,j\} \in E^+(G) \tag{24}$$

$$a_{ii} + a_{jj} + 2a_{ij} < 0 \qquad \text{for all } \{i,j\} \in E^-(G) \tag{25}$$

for a partition $E^+(G) \cup E^-(G)$ of $E(G)$ such that $E^+(G) \supseteq \{\{i,j\} \in E(G) : a_{ij} > 0\}$ and $E^-(G) \supseteq \{\{i,j\} \in E(G) : a_{ij} < 0\}$.

5.1 Definition of the neighborhood function used for case C2

The neighborhood function $\mathcal{F}$ associated with class C2 is implicitly defined as follows. $x' \in \{0,1\}^n$ is a neighbor of $x \in \{0,1\}^n$ if x and x' differ in exactly two components i and j, with $\{i,j\} \in E(G)$ and x_i and x_j satisfying the following condition: if $\{i,j\} \in E^+(G)$ then $x_i = 1 - x_j$ $(= 1 - x'_i = x'_j)$; if $\{i,j\} \in E^-(G)$ then $x_i = x_j$ $(= 1 - x'_i = 1 - x'_j)$. Note that $\mathcal{G} = \bigcup_{x \in \{0,1\}^n} \mathcal{F}(x) = E(G)$.

Let δ_F for $F = \{i,j\} \in \mathcal{G}$ be such that

$$0 > \delta_F \geq \begin{cases} a_{ii} + a_{jj} - 2a_{ij} & \text{if } \{i,j\} \in E^+(G) \\ a_{ii} + a_{jj} + 2a_{ij} & \text{if } \{i,j\} \in E^-(G). \end{cases}$$

When $\widetilde{c} = 0$, we also allow the choice $\delta_F = 0$ for all $F = \{i,j\} \in \mathcal{G}$.

Using the neighborhood function $\mathcal{F}$ introduced above, the defining inequalities (8) of the polyhedron $P_{x,\delta}$ (or the inequalities (9) defining its perturbed version $\widehat{P}_{x,0}$) simplify to

$$\rho_i(y) + \rho_j(y) \leq \delta_F \quad \text{for all } F = \{i,j\} \in E^-(G) \text{ such that } x_i = x_j = 1 \tag{26}$$

$$-\rho_i(y) - \rho_j(y) \leq \delta_F \quad \text{for all } F = \{i,j\} \in E^-(G) \text{ such that } x_i = x_j = 0 \tag{27}$$

$$\rho_i(y) - \rho_j(y) \leq \delta_F \quad \text{for all } F = \{i,j\} \in E^+(G) \text{ such that } x_i = 1, x_j = 0. \tag{28}$$

5.2 Construction of the set $\mathbf{X}(y)$ for case C2

Assume that Y has already been computed and let $y \in Y$. To compute the set $X(y)$, we need to consider all points $x \in \{0,1\}^n$ such that $y \in P_{x,\delta}$. From the definition of $P_{x,\delta}$ by the inequalities (26)–(28), it follows that the set $\{x \in \{0,1\}^n : y \in P_{x,\delta}\}$ is equal to the set of points $x \in \{0,1\}^n$ satisfying

$$x_i + x_j \leq 1 \qquad \text{for all } F = \{i,j\} \in E^-(G) : \rho_i(y) + \rho_j(y) > \delta_F \tag{29}$$

$$x_i + x_j \geq 1 \qquad \text{for all } F = \{i,j\} \in E^-(G) : \rho_i(y) + \rho_j(y) < -\delta_F \tag{30}$$

$$x_i \leq x_j \qquad \text{for all } F = \{i,j\} \in E^+(G) : \rho_i(y) - \rho_j(y) > \delta_F. \tag{31}$$

Recall that the values of some components of x were fixed when computing y. Using the inequalities (29)–(31), the value of some other components may be determined. If a contradiction occurs, $X(y) = \emptyset$; otherwise let J_y denote the set of the indices $j \in \{1,\ldots,n\}$ for which the value of x_j has not been fixed. For $j \in \{1,\ldots,n\} \setminus J_y$, let x_j^y denote the already fixed value of the j-th component of x. We define the set $X(y)$ as the set of points $x \in \{0,1\}^n$ satisfying

$$x_i + x_j \leq 1 \qquad \text{for all } \{i,j\} \in E^-(G[J_y]) : \rho_i(y) + \rho_j(y) \geq 0 \tag{32}$$

$$x_i + x_j \geq 1 \qquad \text{for all } \{i,j\} \in E^-(G[J_y]) : \rho_i(y) + \rho_j(y) \leq 0 \tag{33}$$

$$x_i \leq x_j \qquad \text{for all } \{i,j\} \in E^+(G[J_y]) : \rho_i(y) \geq \rho_j(y) \tag{34}$$

$$x_j = x_j^y \qquad \text{for all } j \in \{1,\ldots,n\} \setminus J_y. \tag{35}$$

The inclusion $\{x \in \{0,1\}^n : y \in P_{x,\delta}\} \subseteq X(y)$ is obvious if $\delta_F < 0$ for all $F \in \mathcal{G}$. In the case $\delta_F = 0$ for all $F \in \mathcal{G}$, we additionally need to observe that $\rho_i(y) + \rho_j(y) \neq 0$ holds for all $\{i,j\} \in E^-(G[J_y])$ and that $\rho_i(y) \neq \rho_j(y)$ holds for all $\{i,j\} \in E^+(G[J_y])$ by Corollary 2. The following observation turns out to be helpful to compute the set $X(y)$.

Observation 1 *Let U be a subset of J_y. Let a (respectively b) be the vertex of U with minimum value $\rho_a(y)$ (respectively maximum value $\rho_b(y)$).*
If $\rho_a(y) + \rho_b(y) \leq 0$, fixing x_a to 0 forces the value of all x_i, $i \in U$, such that $\{i,a\} \in E(G[J_y])$.
If $\rho_a(y) + \rho_b(y) > 0$, fixing x_b to 1 forces the value of all x_i, $i \in U$, such that $\{i,b\} \in E(G[J_y])$.

Indeed, assume that $\rho_a(y) + \rho_b(y) \leq 0$. Then by definition of a and b, for all $j \in U$ we have $\rho_a(y) + \rho_j(y) \leq 0$ and $\rho_a(y) \leq \rho_j(y)$. By (33), we therefore conclude that $x_j = 1$ for all $j \in U$ such that $\{a,j\} \in E^-(G[J_y])$ and by (34) we conclude that $x_j = 0$ for all $j \in U$ such that $\{a,j\} \in E^+(G[J_y])$. Similarly if $\rho_a(y) + \rho_b(y) > 0$, fixing x_b to 1 results in the fixation of x_i to 0 for all $i \in U$ such that $\{i,b\} \in E^-(G[J_y])$ and in the fixation of x_i to 1 for all $i \in U$ such that $\{i,b\} \in E^+(G[J_y])$.

Using Observation 1, the set $X(y)$ can be computed by calling the recursive procedure enumStable, described below, with parameters $(x, J_y, \emptyset)$. We will use the notion *partial*

binary vector for a vector whose components have three possible entries: 0, 1 or "not yet defined". Recall that for the vector x in the initial call all entries x_j with $j \in J_y$ are equal to "not yet defined", while we have $x_j = x_j^y$ for $j \in \{1, 2, \ldots, n\} \setminus J_y$.

Moreover, we assume that the components of x are renumbered in such a way that the corresponding values of $\rho_i(y)$ are ordered non-decreasingly, i.e., $\rho_1(y) \leq \rho_2(y) \leq \ldots \leq \rho_n(y)$.

enumStable $(\bar{x}, L, S)$
{$\bar{x}$: a partial binary vector }
{L: set of nonfixed components in $\bar{x}$ }
{S: this parameter is not necessary; it will be useful when analyzing the complexity }

1. If $L = \emptyset$
2. add $\bar{x}$ to $X(y)$
3. Else
   $a = \arg\min_{i \in L} \rho_i(y)$; $b = \arg\max_{i \in L} \rho_i(y)$
4. if $\rho_a(y) + \rho_b(y) \leq 0$, choose $v = a$; otherwise choose $v = b$
   $\bar{x}' = \bar{x}$; if $v = a$, set $\bar{x}'_v$ to 0; otherwise set $\bar{x}'_v$ to 1
   determine $\bar{x}'_w$ for $w \in L$ such that $\{v, w\} \in E(G[J_y])$
   according to Observation 1
5. enumStable $(\bar{x}', (L \setminus \{v\}) \setminus \{w \in L : \{v, w\} \in E(G[J_y])\}, S \cup \{v\})$
   $\bar{x}' = \bar{x}$; if $v = a$, set $\bar{x}'_v$ to 1; otherwise set $\bar{x}'_v$ to 0
6. enumStable $(\bar{x}', L \setminus \{v\}, S)$
7. End If.

In line 4, a not yet fixed component v is selected to be fixed. The two possible values for this component are considered. For one of these values, Observation 1 allows to fix all nonfixed components corresponding to a vertex that is connected to v by an edge in the graph G. The procedure enumStable is then recursively called to select and fix a new component. When all components have been fixed (line 1), x is a completely defined binary vector that is added to the set $X(y)$ in line 2. The correctness of the computation of the set $X(y)$ should be obvious. Let us now evaluate its complexity. Notice that if we remove all unnumbered lines of the procedure enumStable and if we print the set S at line 2, we will get the list of all stable sets, maximal or not, of the graph $G[J_y]$, and by consequence, a sublist of all stable sets of the graph G. For each stable set listed at line 2, we have at most n calls to procedure enumStable, and each call to enumStable requires $O(n)$ time due to the computation of the second parameter in the recursive call to enumStable at line 5 if we assume that the graph G is given by an adjacency matrix. Hence the overall complexity of the computation of the set $X(y)$ is $O\left(n^2 \cdot ind(G)\right)$ where $ind(G)$ denotes the number of independent sets (or stable sets) in the graph G.
It follows that the set $X(y)$ can be computed in polynomial time if the conditions (24)–(25) are satisfied.

5.3 Enumeration of the points of empty neighborhood for class C2

In order to complete the proof of Theorem 2, it remains to be shown that the set of points $x \in \{0,1\}^n$ such that $\mathcal{F}(x) = \emptyset$ can be computed in polynomial time. From the definition of the neighborhood function $\mathcal{F}$, it follows that $\mathcal{F}(x) = \emptyset$ if and only if the following equalities are satisfied:

$$x_i = x_j \qquad \text{for all } \{i,j\} \in E^+(G) \tag{36}$$

$$x_i = 1 - x_j \qquad \text{for all } \{i,j\} \in E^-(G). \tag{37}$$

It follows from (36)–(37) that for each connected component CC of G, if suffices to fix the value x_{i_0} for some $i_0 \in CC$ to determine all other x_j, $j \in CC$. If for some CC a contradiction occurs, we conclude that there exists no point with an empty neighborhood. We assume from now on that the system (36)–(37) is feasible. Since for each connected component CC, there correspond 2 sets of values for the corresponding $x_j, j \in CC$ the number of points with an empty neighborhood is $2^{k(G)}$ where $k(G)$ is the number of connected components in the graph G. Now by selecting one vertex of each connected component, we get a stable set S of G of cardinality $k(G)$. Since each subset of S is itself a stable set of G, we have

$$ind(G) \geq 2^{k(G)}$$

where $ind(G)$ is the total number of stable sets in the graph G. By the assumption of Theorem 2, $ind(G)$ is bounded from above by a polynomial in n, hence this is also the case for the number of 0-1 points with an empty neighborhood. Therefore the points of empty neighborhood can be found in polynomial time, which concludes the proof of Theorem 2.

The condition on the polynomial number of stable sets of G in Theorem 2 can be replaced by the following weaker condition which is easier to check.

Condition 2 *The degree of any vertex in G is at least $n - c\log n$, where c is a constant.*

Indeed the number of stable sets containing the vertex $i \in V(G)$ is bounded by $2^{c\log n} = n^c$ which implies that the total number of stable sets is bounded by $n \cdot n^c = n^{c+1}$.

6 Comparison with the algorithm of Allemand, Fukuda, Liebling, and Steiner

In their paper, Allemand, Fukuda, Liebling and Steiner (2001) propose a polynomial algorithm for solving problem (2) when there is no linear term (i.e., $c = 0$) and the matrix A is negative semidefinite (i.e., $\lambda_\ell < 0$ for $\ell = 1, \ldots, d$ if a spectral decomposition of A is used, see the explanation in Section 1 for further details).

The algorithm of Allemand et al. (2001) involves the enumeration of the extreme points of a special polytope, called *zonotope*. The reader interested into the practical

implementation of the method of Allemand et al. is recommended to read the recent paper of Ferrez, Fukuda and Liebling (2004) where an improved method for enumerating the extreme points of the zonotope is proposed.

6.1 Method of Allemand, Fukuda, Liebling, and Steiner

In this section we briefly describe the method of Allemand et al (2001) in a slightly more general framework. We are going to consider the problem:

$$\min_{x\in\{0,1\}^n} \quad f(x) = \beta_0 + \langle u^0, x\rangle + \sum_{\ell=1}^{d} \lambda_\ell \left(\beta_\ell + \langle u^\ell, x\rangle\right)^2.$$

The case treated by Allemand et al. arises by setting $u^0 = 0$ and $\beta_\ell = 0$ for all $\ell = 0, \ldots, d$.

Consider the mapping T from $\mathbb{R}^n$ to $\mathbb{R}^{d+1}$ that transforms a point x into the point $T(x) = (\beta_0 + \langle u^0, x\rangle, \ldots, \beta_d + \langle u^d, x\rangle)$. The image of the hypercube $[0,1]^n$ is a special polytope Q_z of $\mathbb{R}^{d+1}$, called *zonotope*. The crucial observation is that Q_z has $O(n^d)$ extreme points, which can be computed in $O(n^d)$ time (see Allemand et al., 2001); note that in the special case treated in Allemand et al. (2001) the zonotope is d-dimensional rather than $(d+1)$-dimensional). The algorithm in Allemand et al. (2001) evaluates the expression $z_0 + \sum_{\ell=1}^{d} \lambda_\ell (z_\ell)^2$ for each extreme point $z = (z_0, \ldots, z_d)$ of Q_z and keeps the points of smallest value. Observe that, while each extreme point of Q_z is the image of some point $x \in \{0,1\}^n$, not all points in $\{0,1\}^n$ are transformed into an extreme point of Q_z. Therefore, the algorithm works correctly only if the optimal solution can be shown to be among the points $x \in \{0,1\}^n$ corresponding to an extreme point of Q_z. Allemand et al. observed that this property is true when the matrix A is negative semidefinite by exploiting the concavity of the objective function. The next lemma shows that the approach of Allemand et al. works for a larger class of instances of the CR-QP01.

Proposition 5 *Let $\mathcal{I}$ be an instance of the problem CR-QP01 with the property that all optimal solutions of the continuous relaxation of instance $\mathcal{I}$ are integral. Then the algorithm of Allemand et al. solves the instance $\mathcal{I}$ to optimality.*

Proof. We are going to show that if the algorithm of Allemand et al. fails, then there exists an optimal solution of the continuous relaxation that is fractional, contradicting the assumption of the proposition.

Let x^* be an optimal solution of problem CR-QP01 that is not found by the algorithm of Allemand et al. It follows that the image $z^* = T(x^*)$ of x^* under the mapping T is not an extreme point of the zonotope Q_z. Therefore, z^* can be written as a convex combination of $t \geq 2$ extreme points of Q_z, say $z^{(1)}, \ldots, z^{(t)}$. Let $\xi^{(j)} \in \{0,1\}^n$ be such that $z^{(j)}$ is the image of $\xi^{(j)}$ under T, i.e., $z^{(j)} = T(\xi^{(j)})$ for $j = 1, \ldots, t$. Hence, there exists a real vector

$\nu = (\nu_1, \ldots, \nu_t) \geq 0$ with $\sum_{j=1}^{t} \nu_j = 1$ such that

$$\begin{aligned} z_\ell^* &= \sum_{j=1}^{t} \nu_j z_\ell^{(j)} = \sum_{j=1}^{t} \nu_j \left(\beta_\ell + \langle u^\ell, \xi^{(j)} \rangle\right) \\ &= \beta_\ell + \left\langle u^\ell, \sum_{j=1}^{t} \nu_j \xi^{(j)} \right\rangle \qquad \ell = 0, \ldots, d. \end{aligned}$$

But then $\sum_{j=1}^{t} \nu_j \xi^{(j)}$ is a feasible solution of the continuous relaxation with the same objective function value than x^*. It follows that the continuous relaxation has at least one optimal solution that is fractional. ■

A large class of instances of the CR-QP01 to which Proposition 5 applies, results from the class $\mathcal{N}$ of all $n \times n$ matrices A of constant rank with strictly negative entries on the main diagonal. Indeed, suppose the contrary. Let $\mathcal{I}'$ be an instance resulting from a matrix $A' \in \mathcal{N}$. Let $x^* = (x_1^*, \ldots, x_n^*)$ be an optimal solution of the continuous relaxation of $\mathcal{I}'$, and assume that x_j^* is fractional for some j. Then x_j^* is the optimal solution of a one-dimensional quadratic optimization problem of the form

$$\min_{0 \leq x_j \leq 1} \left\{ a'_{jj} x_j^2 + B(x_1^*, \ldots, x_{j-1}^*, x_{j+1}^*, \ldots, x_n^*) x_j + C(x_1^*, \ldots, x_{j-1}^*, x_{j+1}^*, \ldots, x_n^*) \right\}$$

for some quadratic functions B and C. Since $a'_{jj} < 0$, the optimum cannot be attained at a fractional value, contradicting the assumption.

Note that the class of instances resulting from matrices $A \in \mathcal{N}$ is a special case of the class C1 considered in Section 4. This special case is obtained by using the hypergraph $H = (V(H), E(H))$ where $E(H)$ contains only edges of size 1 (i.e. H is a graph all of whose edges are loops). It is not difficult to verify that the conditions of Theorem 1 are satisfied for H.

Finally observe that the class of instances resulting from the class of negative semidefinite matrices with constant rank (for which the algorithm of Allemand et al. was proposed) can also be solved in polynomial time by our approach. Indeed, it follows from the definition of negative semidefiniteness, that any negative semidefinite matrix A satisfies $a_{ii} \leq 0$ for $i = 1, \ldots, n$. Moreover if $a_{ii} = 0$ for some i, then $a_{ij} = 0$ for any $j = 1, \ldots, n$. In this later case, the optimal value of x_i can be easily determined by looking to the sign of its coefficient in the linear term. Hence the problem is reduced to one with matrices in $\mathcal{N}$, which, as noted earlier, can be solved in polynomial time by our first algorithm. It is to be pointed out, however, that the complexity of the algorithm of Allemand et al. is better by an order n than the complexity of our approach, for this class of problems.

6.2 Non-dominance

We now compare our approach with the approach of Allemand et al. We show that none of the two approaches dominates the other with respect to the class of instances of the CR-QP01 which can be solved in polynomial time.

We first present an instance $\mathcal{I}_1$ of the CR-QP01 which is solvable by our approach, but not by the approach of Allemand et al. Consider the quadratic function

$$f(x) = (n^2 x_1 + x_2)^2 - \left(x_1 + 2n^2 x_2 + \sum_{i=3}^{n} (2n^2 + i) x_i \right)^2 .$$

It is easy to check that the matrix A corresponding to the quadratic part of f satisfies the condition $a_{ii} + a_{jj} + 2|a_{ij}| < 0$ for $1 \leq i < j \leq n$. The resulting class of instances belongs to both C1 and C2, and can hence be solved in polynomial time by our algorithms.

We are now going to argue that the approach of Allemand et al. fails. Since the variables $x_i, i = 3, \ldots, n$, appear only in the second term, they must take the value 1 in an optimal solution of the continuous relaxation. Solving the 2-dimensional problem in the remaining variables x_1 and x_2 shows that the unique minimum is attained for $x = (\lambda, 1, \ldots, 1)$ with $\lambda = \frac{n^2 + (n-2)\left(2n^2 + \frac{n+3}{2}\right)}{n^4 - 1}$ (see the appendix for more details). Therefore the instance $\mathcal{I}_1$ cannot be solved by the method of Allemand et al.

Next we give an example of an instance $\mathcal{I}_2$ which can be solved by the approach of Allemand et al., but not by our approach. Consider the quadratic function given by

$$f(x) = \left(\sum_{i=1}^{n} (i+1) x_i + 1 \right)^2 - \left(\sum_{i=1}^{n} \frac{x_i}{i} \right)^2 .$$

This instance results from the matrix A with entries $a_{ij} = (i+1)(j+1) - \frac{1}{ij}$. It is easy to see that A does neither belong to class C1 nor to class C2 (note that $a_{ii} + a_{jj} - 2|a_{ij}| \geq 0$ for all i, j). Consequently our methods do not apply. On the other hand, $f(x)$ is positive for all $x \in [0,1]^n$, therefore the optimal solution of the continuous relaxation must be integral (see Hammer, Hansen, Pardalos and Rader, 2002), and hence this problem can be solved in polynomial time by the method of Allemand et al. (and also by the method of Hammer et al.).

7 Conclusions

In this paper, we derived two new polynomially solvable special cases of the CR-QP01. Our generic algorithm works by enumerating a superset of the set of local minima of the objective function f with respect to a suitably chosen neighborhood.

Due to the high running time of our algorithms, our results are mainly of theoretical interest. Although it is possible to reduce the running times for special cases by exploiting

the special structure of the underlying hypergraph (case C1) or graph (case C2), it is unlikely that the reductions are large enough to result in running times which are acceptable for practical applications. It is, however, conceivable that heuristics obtained from the general idea of our approach lead to promising results. For example, one could think of developing local search heuristics based on the neighborhoods used in this paper. Another way to arrive at a heuristic is to refrain from computing the full set Y (recall that the running times of the proposed algorithms depend on the cardinality of the set Y) and be instead satisfied with a set Y' of randomly selected points of $\mathbb{R}^d$. Instead of searching for the best solution in $X(Y)$, we then search for the best solution in $X(Y')$.

Finally note that the classes presented in this paper are special cases of the more general class defined by:

$$\sum_{i,j\in F_1} a_{ij} + \sum_{i,j\in F_2} a_{ij} - 2\sum_{i\in F_1}\sum_{j\in F_2} a_{ij} < 0 \qquad \text{for all } (F_1,F_2)\in E(HH)$$

where $HH = (V(HH), E(HH))$ is a "hyperhypergraph" whose edges are pairs $\{F_1, F_2\}$ of subsets of $V(HH)$ (a hypergraph can then be considered as the special case of a hyperhypergraph with all edges of the form $\{F, \emptyset\}$ where F is a subset of $V(HH)$). In particular, the class considered in Section 4 corresponds to the hyperhypergraphs with edges $\{F_1, F_2\}$ satisfying $|F_2| = 0$ and the class considered in Section 5 corresponds to the hyperhypergraphs with edges $\{F_1, F_2\}$ satisfying $|F_1| + |F_2| = 2$. This suggests the following question: What conditions on HH ensure that the associated instances of the CR-QP01 can be solved in polynomial time?

Appendix

In this appendix we provide more details on the solution of the continuous relaxation of the instance $\mathcal{I}_1$ of the CR-QP01 which has been investigated in Section 6.2. The resulting quadratic programming problem QP is given by:

$$\min_{x\in[0,1]^n} f(x) = (n^2x_1 + x_2)^2 - \left(x_1 + 2n^2x_2 + \sum_{i=3}^{n}(2n^2+i)x_i\right)^2.$$

Set $h_1(x) = n^2x_1 + x_2$ and $h_2(x) = x_1 + 2n^2x_2 + \sum_{i=3}^{n}(2n^2+i)x_i$. Hence we can write $f(x) = (h_1(x))^2 - (h_2(x))^2$. Since the variables x_i, $i = 3,\ldots,n$ only appear in the second term and since $h_2(x) \geq 0$, we will have $x_i = 1$, $i = 3,\ldots,n$ in any optimal solution. Consequently we are left with a function in 2 variables:

$$g(x_1,x_2) = (n^2x_1 + x_2)^2 - \left(x_1 + 2n^2x_2 + (n-2)\left(2n^2 + \frac{n+3}{2}\right)\right)^2.$$

Assume for a moment that the value of the function $h_2(x)$ at the optimum is known, and let this value be denoted by h_2^*. Then the optimal solution x^* of the QP can be obtained as solution of the following continuous knapsack problem (see Hammer et al., 2002):

$$\begin{array}{ll} \min & n^2x_1 + x_2 \\ \text{s.t.} & \left\{ \begin{array}{l} x_1 + 2n^2x_2 = h_2^* - (n-2)\left(2n^2 + \frac{n+3}{2}\right) \\ x_1, x_2 \in [0,1]. \end{array} \right. \end{array}$$

Since $\frac{n^2}{1} > \frac{1}{2n^2}$, it is well known that the optimum solution is either of the form $(x_1, x_2) = (0, \lambda)$ or $(x_1, x_2) = (\lambda, 1)$ with $0 \leq \lambda \leq 1$. The minimum of

$$g(0,\lambda) = \lambda^2 - \left(2n^2\lambda + (n-2)\left(2n^2 + \frac{n+3}{2}\right)\right)^2$$

on $[0,1]$ is attained for $\lambda = 1$ (observe, for example, that the derivative of $g(0,\lambda)$ with respect to λ is negative).

On the other hand, we have

$$g(\lambda,1) = (n^2\lambda + 1)^2 - \left(\lambda + 2n^2 + (n-2)\left(2n^2 + \frac{n+3}{2}\right)\right)^2.$$

By setting the derivative of $h(\lambda) = g(\lambda, 1)$ equal to 0, we obtain that the minimum of g is attained for

$$\tilde{\lambda} = \frac{n^2 + (n-2)\left(2n^2 + \frac{n+3}{2}\right)}{n^4 - 1}.$$

Note that $0 < \tilde{\lambda} < 1$ for $n \geq 2$. Observing that $g(\tilde{\lambda}, 1) < g(0,1)$, we conclude that the minimum of f over $[0,1]^n$ is obtained for $x = (\tilde{\lambda}, 1, \ldots, 1)$, as claimed in Section 6.2.

References

K. Allemand, K. Fukuda, T.M. Liebling, and E. Steiner. "A polynomial case of unconstrained zero-one quadratic optimization". *Mathematical Programming*, 91(1):49–52, 2001.

F. Barahona. "A solvable case of quadratic 0-1 programming". *Discrete Applied Mathematics*, 13(1):23–26, 1986.

L.W. Beineke. "Biplanar graphs: a survey". *Computers & Mathematics with Applications*, 34(11):1–8, 1997.

C. Berge. Graphs and Hypergraphs. North-Holland Publishing Company, second edition, 1976.

H. L. Bodlaender. "A partial k-arboretum of graphs with bounded treewidth". *Theoretical Computer Science*, 209(1-2):1–45, 1998.

E. Boros and P.L. Hammer. "The max-cut problem and quadratic 0-1 optimization; polyhedral aspects, relaxations and bounds". *Annals of Operations Research*, 33(1-4):151–180, 1991.

E. Boros, K. Elbassioni, V. Gurvich and L. Khachiyan. "Generating maximal independent sets for hypergraphs with bounded edge-intersections". LATIN 2004: Theoretical Informatics: 6th Latin American Symposium.(Farach-Colton, Martin, ed., Buenos Aires, Argentina, April 5-8, 2004), *Lecture Notes in Computer Science* 2976:488–498, 2004.

C. Bron and J. Kerbosch. "Finding all cliques of an undirected graph". *Communications of the ACM*, 16:575–577, 1973.

N. Chiba and T. Nishizeki. "Arboricity and subgraph listing algorithms". *SIAM Journal on Computing*, 14:210–223, 1985.

Y. Crama, P. Hansen, and B. Jaumard. "The basic algorithm for pseudo-Boolean programming revisited". *Discrete Applied Mathematics*, 29(2-3):171–185, 1990.

E. Dahlhaus and M. Karpinski. "A fast parallel algorithm for computing all maximal cliques in a graph and related problems". In: *Proceedings of the 1st Scandinavian Workshop on Algorithm Theory*, *Lecture Notes in Computer Science*, Volume 318, pp. 139–144, Springer Verlag, Berlin, Heidelberg, New York, 1988.

H. Edelsbrunner. Algorithms in Combinatorial Geometry, EATCS Monographs on Theoretical Computer Science, Volume 10. Springer Verlag, Berlin, Heidelberg, New York, 1987.

T. Eiter and G. Gottlob. "Identifying the minimal transversals of a hypergraph and related problems". *SIAM Journal on Computing*, 24(6):1278–1304, 1995.

J.-A. Ferrez, K. Fukuda and T. M. Liebling. Solving the fixed rank convex quadratic maximization in binary variables by a parallel zonotope construction algorithm. *European Journal of Operational Research*, 166(1):35–50, 2005.

F.R. Gantmacher. The Theory of Matrices, AMS Chelsea Publishing, 1959.

M.R. Garey and D.S. Johnson. Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman, San Francisco, CA, 1979.

P.L. Hammer, P. Hansen, P.M. Pardalos, and D.J. Rader. "Maximizing the product of two linear functions in 0-1 variables". *Optimization*, 51:511–537 (2002).

D. S. Johnson, M. Yannakakis, and C.H. Papadimitriou. "On generating all maximal independent sets". *Information Processing Letters*, 27:119–123, 1988.

G. D. Mulligan and D. G. Corneil. "Corrections to Bierstone's algorithm for generating cliques". *Journal of the Association on Computing Machinery*, 19:244–247, 1972.

C. H. Papadimitriou. "On the complexity of integer programming". *Journal of the Association on Computing Machinery*, 28(4):765–768, 1981.

P. M. Pardalos and S. Jha. "Graph separation techniques for quadratic zero-one programming". *Computers & Mathematics with Applications*, 21(6/7):107–113, 1991.

J. C. Picard and H. D. Ratliff. "Minimum cuts and related problems". *Networks*, 5(4):357–370, 1975.

S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa. "A new algorithm for generating all the maximal independent sets". *SIAM Journal on Computing*, 6:505–517, 1977.