Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7, D-14195 Berlin

Christoph Helmberg
Robert Weismantel

Cutting Plane Algorithms for Semidefinite Relaxations

Preprint SC 97–02 (January 1997)

Cutting Plane Algorithms for Semidefinite Relaxations

Christoph Helmberg* Robert Weismantel†

January 24, 1997

Abstract

We investigate the potential and limits of interior point based cutting plane algorithms for semidefinite relaxations on basis of implementations for max-cut and quadratic 0-1 knapsack problems. Since the latter has not been described before we present the algorithm in detail and include numerical results.

Key words. Semidefinite programming, max-cut, quadratic 0-1 knapsack, cutting plane algorithms, interior point methods.

AMS subject classifications (MSC 1991). 90C25, 90C09

1 Introduction

Pioneered by the work of Lovász on the Shannon capacity of graphs [27] the interest in semidefinite relaxations of combinatorial optimization problems has been steadily increasing [16, 28, 7, 34]. With the development of interior point algorithms [23, 32, 1, 37, 21, 25, 2, 33] practical methods for computing these relaxations became available. This encouraged research in the field and, as a consequence, several new approximation results based on semidefinite relaxations appeared within short time [14, 29, 13, 5, 12]. Most semidefinite relaxations can be improved in a canonical way by a cutting plane approach. Computational results show that the corresponding bounds are of high quality [19, 22, 20, 24, 40, 39]. However, semidefinite relaxations are expensive to compute. In this paper we try to explore the potential and the limits of cutting plane algorithms that are based on interior point methods by studying two typical semidefinite relaxations. In particular we will recapitulate the most important steps of the cutting plane algorithm for max-cut as described in [20] and compare this to a cutting plane algorithm for the quadratic 0-1 knapsack

*Konrad Zuse Zentrum für Informationstechnik Berlin, Takustraße 7, D–14195 Berlin, Germany. e-mail: helmberg@zib.de, URL: http://www.zib.de/helmberg.

†Konrad Zuse Zentrum für Informationstechnik Berlin, Takustraße 7, D–14195 Berlin, Germany. e-mail: weismantel@zib.de

problem. The latter algorithm was used but not explained in [22], so we will go into some details.

There is a good reason for comparing these particular problems. In the case of max-cut the combinatorial structure of the feasible set, the cut polytope, does not depend on the data. The max-cut polytope is very well studied in the literature (see e.g. the survey of Poljak and Tuza [35] or the forthcoming book of Deza and Laurent [10]), and many classes of valid inequalities are known. The quadratic 0-1 knapsack problem is strongly related to max-cut. Indeed, it can be interpreted as max-cut with one additional constraint. In spite of this close relationship the underlying polyhedron differs substantially from the cut polytope, because it depends on the data. We will see that this has consequences for cutting plane algorithms.

Cutting plane methods have become a standard technique for solving combinatorial optimization problems. Iteratively one computes an optimal solution of a relaxation. If it is not feasible for the original problem, the separation problem must be solved, i.e. find inequalities that are valid for the original problem but cut off the current point.

In integer linear programming cutting plane algorithms usually employ the simplex method. With the dual simplex method one can incorporate the optimal solution of the previous relaxation in order to compute the optimal solution of the current relaxation. This possibility of restarting is missing in interior point algorithms. However, interior point algorithms offer other possibilities which may compensate for this disadvantage (see e.g. [30]).

In semidefinite programming there is currently no practical alternative to interior point methods. As in linear programming a general scheme for reoptimization is missing, although the design of efficient restart routines seems possible if a large part of the central path associated with the semidefinite relaxation runs inside the underlying combinatorial polytope. This property makes certain strategies work well for max-cut which fail for quadratic knapsack problems.

In solving semidefinite relaxations by interior point methods computation time increases rapidly with the number of cutting planes added. This is due to the fact that all inequalities contribute to the computation of the step direction. Furthermore interior point methods offer little possibility to exploit the sparsity of constraint or cost matrices. Since typically hundreds of violated inequalities can be found, efficiency requirements necessitate the selection of a small efficacious subset. The problem of deciding on the cutting planes to be added can be approached from two points of view. On the one hand we would like to know the strength of cutting planes for the problem in general. How much do certain classes of inequalities improve the initial relaxation? Is there a hierarchy of cuts, i.e. do some hyperplanes guarantee the feasibility of other ones? On the other hand, from a computational point of view it is important to select those cutting planes which improve the current relaxation with respect to the current cost function the most. Some partial answers are given in [22] for the quadratic 0-1 knapsack problem. We address this question here from a practical point of view.

In order to locate the most expensive operations within semidefinite interior point methods we give a concise description of the interior point method in Section 2. In Section 3 we outline the cutting plane algorithm of max-cut. In doing so we will mention possible generalizations and point out several open problems. Section 4 gives a detailed description of our cutting plane algorithm for the quadratic 0-1 knapsack problem. We will be able to build on experience from max-cut, but some fundamental differences will become evident. In Section 5 we summarize the conclusions and point out further directions of research.

Notation

$\mathbb{R}$ ($\mathbb{N}$, $\mathbb{Z}$)	real (natural, integral) numbers
$\mathbb{R}^n$ ($\mathbb{N}^n$, $\mathbb{Z}^n$)	real (natural, integral) column vector of dimension n
$M_{m,n}$, M_n	$m \times n$, $n \times n$ real matrices
S_n	$n \times n$ symmetric real matrices
$A \succeq 0$, $A \succ 0$	A is (symmetric) positive semidefinite, positive definite
I, I_n	identity of appropriate size or of size n
e_i	i-th column of I
e	vector of all ones of appropriate dimension
$\operatorname{tr}(A)$	trace of $A \in M_n$
$\langle A, B\rangle$	inner product in $M_{m,n}$, $\langle A, B\rangle = \operatorname{tr}(B^T A)$
$\operatorname{diag}(A)$	$\operatorname{diag}(A) = [a_{11}, \dots, a_{nn}]^T$
$\operatorname{Diag}(v)$	diagonal matrix with diagonal v
$\mathcal{A}(X)$	$= [\langle A_1, X\rangle, \dots, \langle A_m, X\rangle]^T$ for given $A_i \in S_n$
$\mathcal{A}^T(y)$	$= \sum_{i=1}^m y_i A_i$
$\operatorname{conv}(\cdot)$	convex hull operator

Unless explicitly stated otherwise all matrices that we consider here are symmetric and vectors are interpreted as columns.

2 The Interior Point Method

To demonstrate the most expensive operations within one "Newton"-step of the interior point code we briefly sketch the main steps of the computation of the search direction. The search direction is based on the linearization $\Delta XZ + X\Delta Z$ which was introduced in [21, 25]. Consider the primal-dual pair of semidefinite programs

$$\text{(P)} \quad \begin{array}{ll} \min & \langle C, X\rangle \\ \text{s.t.} & \mathcal{A}(X) = b \\ & X \succeq 0 \end{array} \qquad\qquad \text{(D)} \quad \begin{array}{ll} \max & \langle b, u\rangle \\ \text{s.t.} & \mathcal{A}^T(u) + Z = C \\ & Z \succeq 0. \end{array}$$

$\mathcal{A} : S_n \to \mathbb{R}^m$ is a linear operator of the form

$$\mathcal{A}(X) = \begin{bmatrix} \langle A_1, X\rangle \\ \vdots \\ \langle A_m, X\rangle \end{bmatrix},$$

with $A_i \in S_n$, $i = 1, \ldots, m$. Its adjoint operator $\mathcal{A}^T : \mathbb{R}^m \to S_n$, satisfying $\langle \mathcal{A}(X), u \rangle = \langle X, \mathcal{A}^T(u) \rangle$ for all $X \in S_n$ and $u \in \mathbb{R}^m$, is

$$\mathcal{A}^T(u) = \sum_{i=1}^{m} u_i A_i.$$

In the interior point algorithm of [21] a step direction $(\Delta X, \Delta y, \Delta Z)$ is determined by the system of linearized KKT-conditions,

$$\begin{aligned} \mathcal{A}(\Delta X) &= -(\mathcal{A}(X) - b) \\ \mathcal{A}^T(\Delta y) + \Delta Z &= -(\mathcal{A}^T(y) + Z - C) \\ \Delta X Z + X \Delta Z &= \mu I - XZ. \end{aligned}$$

Expressing ΔX and ΔZ in terms of Δy yields

$$\mathcal{A}(X\mathcal{A}^T(\Delta y)Z^{-1}) = b - \mathcal{A}(Z^{-1} - X(\mathcal{A}^T(y) + Z - C)Z^{-1}). \tag{1}$$

The $m \times m$ matrix $M = \mathcal{A}(X\mathcal{A}^T(\cdot)Z^{-1})$ is symmetric and positive definite if X and Z are positive definite and the rank of $\mathcal{A}(\cdot)$ is m. The coefficients of M are determined by

$$M_{ij} = \operatorname{tr}(A_i X A_j Z^{-1}).$$

The system (1) is solved for Δy via a Cholesky or an LDL^T-factorization of M. ΔZ and ΔX are computed by backsubstitution. ΔX is not symmetric in general. In the final step direction we use its symmetric part only.

The two most expensive operations within each iteration of the interior point algorithm are the construction of M and its factorization. For the construction of M one can exploit the structure of the constraint matrices A_i. In particular most of the constraints described in this paper are of the form vv^T or $vw^T + wv^T$ for vectors $v, w \in \mathbb{R}^n$. In addition many of these vectors are sparse. For ease of explanation assume that all A_i are of the form $v_{(i)}v_{(i)}^T$. Then

$$M_{ij} = \operatorname{tr}(A_i X A_j Z^{-1}) = (v_{(i)}^T X v_{(j)})(v_{(j)}^T Z^{-1} v_{(i)}).$$

The vectors $Xv_{(i)}$ and $Z^{-1}v_{(i)}$ have to be computed only once for each row, the remaining elements of the row can be computed in $O(n)$ time each. Neglecting the sparsity of the vectors $v_{(i)}$ the construction of M requires $O(mn^2 + m^2 n)$ operations.

Regarding the work required to factor M one should realize that in general X and Z^{-1}, and hence M, are dense matrices. Therefore the factorization of M requires $m^3/3$ arithmetic operations (see [15]). For cutting plane algorithms m grows quickly. Thus the factorization is the bottleneck of the algorithm. In consequence it is advisable to keep the number of cutting planes small.

We mention that for large m it pays to employ a predictor corrector approach to make better use of the expensive factorization. This issue will not be discussed here.

3 Max-Cut

In the following we explain some typical problems arising in semidefinite cutting plane methods by investigating the combinatorial optimization problem

$$\text{(MC)} \qquad \max\ x^T C x \ \text{ s.t. } x \in \{-1,1\}^n$$

for arbitrary cost matrices $C \in S_n$. If C is the Laplace matrix of a graph, (MC) is equivalent to the max-cut problem of the underlying graph. A detailed description of a semidefinite branch and cut algorithm for (MC) is given in [20]. We outline here the most important issues of this approach. To derive the semidefinite relaxation of (MC) we start with $x^T C x = \langle C, xx^T \rangle$. xx^T is a positive semidefinite matrix of rank one with all diagonal elements equal to one. Ignoring the rank one constraint we relax xx^T in form of a positive semidefinite matrix $X \succeq 0$ satisfying $\operatorname{diag}(X) = e$. We assume that m cutting planes that we code in the linear operator $\mathcal{A}(X) \geq e$ have been added to the initial relaxation. Then the semidefinite relaxation reads

$$\text{(SMC)} \quad \begin{array}{ll} \min & \langle C, X \rangle \\ \text{s.t.} & \operatorname{diag}(X) = e \\ & \mathcal{A}(X) - s = e \\ & X \succeq 0,\ s \geq 0 \end{array} \qquad \text{(DMC)} \quad \begin{array}{ll} \max & \langle e, u_d \rangle + \langle e, u_{\mathcal{A}} \rangle \\ \text{s.t.} & C + Z = \operatorname{Diag}(u_d) + \mathcal{A}^T(u_{\mathcal{A}}) \\ & t = -u_{\mathcal{A}} \\ & Z \succeq 0,\ t \geq 0. \end{array}$$

$u_d \in \mathbb{R}^n$ and $u_{\mathcal{A}} \in \mathbb{R}^m$ are the Lagrange multipliers corresponding to the diagonal and the cutting plane constraints, respectively. The slack variables $s \in \mathbb{R}^m_+$ and $t \in \mathbb{R}^m_+$ are needed when we investigate restart strategies later.

A characteristic property of (MC) is that the combinatorial structure of the feasible set is the same for all problem instances. The semidefinite program (SMC) is a relaxation of the cut polytope,

$$P_C = \operatorname{conv}\left\{ xx^T : x \in \{-1,1\}^n \right\},$$

that has been well studied in the literature (see e.g. the survey by [35] or the forthcoming book [10]). Its center of gravity is the identity, $X = I$, which coincides with the analytic center of (SMC) without cutting planes. A well centered feasible point is available for all problem instances.

Cutting planes.

A general family of valid inequalities for P_C reads

$$b^T X b \geq \min\left\{ b^T xx^T b : x \in \{-1,1\}^n \right\},$$

where $b \in \mathbb{Z}^n$. If the right hand side value is strictly positive, the corresponding inequality is called a *gap inequality*. Since the positive semidefiniteness of X implies $b^T X b \geq 0$, gap inequalities strengthen the positive semidefinite relaxation. If $\min\left\{ b^T xx^T b : x \in \{-1,1\}^n \right\}$ is equal to one, the gap inequality is also called *hypermetric*. If $b\,\{-1,0,1\}^n$ consists of an odd number of $+1$ and -1 entries, hypermetric inequalities are called *clique inequalities*. Clique inequalities always define facets of P_C [4]. Clique inequalities with three nonzero elements specialize to the well known *triangle inequalities*

$$\begin{aligned} x_{ij} + x_{ik} + x_{ik} &\geq -1 \\ x_{ij} - x_{ik} - x_{ik} &\geq -1 \\ -x_{ij} + x_{ik} - x_{ik} &\geq -1 \\ -x_{ij} - x_{ik} + x_{ik} &\geq -1. \end{aligned}$$

The set of all points satisfying all the triangle inequalities defines the so called metric polytope. It is conjectured in [9] that among all valid inequalities for P_C, the triangle inequalities are those with the least distance to the center I of P_C; the conjecture is proved for all valid inequalities having coefficients in $\{-1, 0, 1\}$.

Separation and selection.

Since there are $4\binom{n}{3}$ triangle inequalities and since one iteration of the interior point method requires $O(m^3)$ operations with $m > n$, enumeration of all triangle inequalities is computationally feasible. Routines for separating clique and hypermetric inequalities usually start with triangle inequalities and increase or decrease coefficients of the associated vector b by heuristic rules.

There is another idea for separating hypermetric inequalities that has been around for some time. Intuitively good hypermetric cutting planes should be violated as much as possible, i.e. b should be in the null space of the current solution X^*. To find such vectors b one tries, for instance, to approximate some random vectors from the null space of X^* based on simultaneous Diophantine approximation techniques. Although this idea is appealing we could not make it work in practice.

Usually the number of cutting planes that separation routines return exceeds the threshold on the number of inequalities one is willing to include. In this situation it becomes important to rank inequalities. A traditional rule for selecting inequalities is to rank them by their violation with respect to a normalized representation. In the case of (SMC) a geometric rule seems to work better. We connect the current solution X^* of (SMC) by a straight line segment with the center of gravity. Violated inequalities intersect this line segment. We choose those inequalities whose intersection is closest to the center of gravity. It might be interesting to experiment with rules that try to analyze the influence of the new constraints on the dual problem (DMC).

Early separation and restart

If the semidefinite relaxation wraps the polytope under consideration in a "uniform" way, chances stand that the central path starts within the polytope and leaves it just before reaching the optimal solution of the relaxation. In this case it is an excellent strategy to keep looking for violated inequalities with respect to all iterates instead of calling the separation routines only for the optimal solution of the current relaxation. As soon as violated inequalities are detected, they are added to the problem formulation and the algorithm is restarted. Indeed, in some cases this approach turns out to be a major advantage of interior point methods over the simplex method. In particular this strategy works very well for (SMC). In practice we keep looking for violated triangle inequalities only. A violation of these triggers the complete separation machinery.

Problems of this type may also allow for very efficient restart routines. For example it is possible to store all previous iterates, primal and dual variables. As soon as the new inequalities are selected, determine the last iterate feasible with respect to all new inequalities and restart from there. What are good choices for the new variables corresponding to the new constraints? The primal slack variables are determined via their defining equations. For the dual variables we explain our approach with respect to (DMC). Because we introduced dual slack variables, we can set the new variables $u_{\mathcal{A}}$ to zero and their corresponding slack variables t to one. This causes dual infeasibility on the conditions $t = -u_{\mathcal{A}}$. The next iterates are valid upper bounds as long as the $u_{\mathcal{A}}$ variables remain nonpositive. As the Newton step tries to bring t and $-u_{\mathcal{A}}$ together while keeping t positive, the next step usually yields negative values for $u_{\mathcal{A}}$. For all negative values of $u_{\mathcal{A}}$ we can reset the t values correctly. Even without resetting t, infeasibilities disappear after roughly three iterations.

For machines with rather stringent memory constraints it might not be possible to store all previous iterates if n is large. To overcome this difficulty we construct a new X that remains close to the old iterate. This can be achieved by selecting a point on the straight line segment connecting the current iterate and the center of gravity in such a way that all new inequalities are satisfied strictly. Since the straight line segment is close to the central path and the early separation approach stops the algorithm as soon as a violated inequality is detected, the current iterate is at most "one interior point step" away from feasibility. This makes it likely that the new X stays close to the current iterate. Therefore the pair consisting of the new X and the dual variables associated with the current iterate remains sufficiently close to the central path. Regarding the new variables associated with the new inequalities we apply the approach outlined above. Computational results show that about three iterations suffice to remove infeasibilities without loss of quality in primal and dual solutions.

Elimination of inequalities.

A negative side effect of an early restart is that the dual variables do not have time to converge to zero. Therefore no indicator is available for recognizing redundant inequalities. To avoid massive accumulation of inequalities, it is necessary to compute the optimal solution of the current relaxation after applying an early separation routine ten times, say. Having computed the optimal solution, all inequalities are sorted with respect to their dual cost and all inequalities with "small" dual variables are eliminated (this makes sense if the right hand sides are normalized to one as in (DMC)).

The elimination of inequalities makes it difficult to use the the restart procedure that works with the stored previous iterates, because the dual variables may not be feasible any more. On the other hand it is also difficult to apply a restart procedure that works without storing previous iterates because we can expect that violated inequalities push the new X quite far away from the current optimal solution. Thus the primal dual pair consisting of the new X and the dual variables of the previous iterate will be of poor quality. Indeed, after having computed the optimal solution of a relaxation we always restart from the center of P_C. Better procedures for recognizing redundant inequalities early might remedy these difficulties. In linear programming there is the theoretical concept of Tapia indicators [11]. We are not aware of any practical experience with these indicators for interior point cutting plane algorithms. None the less it may be worth to investigate whether an equivalent indicator exists in semidefinite programming. On the other hand there is still hope that efficient restart procedures for interior point methods will be developed in the future.

Reduced cost fixing.

If the goal is to prove optimality of a given solution and the relaxation is not strong enough one has to resort to branch and cut. An important technique within this framework is reduced cost fixing of variables. A linear programming relaxation of (MC) would include the constraints $-1 \leq x_{ij} \leq 1$. The dual variable corresponding to an active constraint, say $x_{ij} \leq 1$, may suffice to prove that all optimal solutions must satisfy $x_{ij} = 1$. Then we can fix this variable and decrease the dimension of our problem by one. This leads to serious speedups of the algorithm. In (SMC) the constraints $-1 \leq x_{ij} \leq 1$ are implied by the semidefiniteness constraint. The corresponding dual variables are not directly available. In [18] we propose an algorithm for constructing approximate Lagrange multipliers for these constraints. The approach requires the computation of an extremal eigenvalue of the matrix $Z + \alpha(e_i e_j^T + e_j e_i^T)$ for each constraint $x_{ij} \geq -1$ or $x_{ij} \leq 1$. Clearly, it suffices to consider candidates x_{ij} which are close to 1 in absolute values. Yet for some instances the number of candidates is still too large. Is there a way to further reduce the number of candidates?

4 Quadratic Knapsack

In this section we concentrate on the cutting plane algorithm for the quadratic knapsack problem that was used, but not described, in [22]. A good starting point for our discussions is the close relationship between max-cut and unconstrained quadratic 0-1 programming. The latter asks for an optimal solution of

$$\text{(QP)} \qquad \max\ y^T By \ \text{ s.t. } y \in \{0,1\}^n$$

It is well known that (QP) is equivalent to (MC) in $n+1$ variables [8]. To derive a semidefinite relaxation of (QP) we append a component with value 1 and index 0 to y, $\bar{y} = \left[1, y^T\right]^T$. For $y \in \{0,1\}^n$ the matrix $\bar{y}\bar{y}^T$ is positive definite of rank one and its first row and column are equal to its diagonal. Ignoring the rank one constraint and relaxing yy^T to Y we obtain the semidefinite relaxation

$$
\text{(SQ)} \qquad \begin{array}{rl} \max & \langle B, Y \rangle \\ \text{s.t.} & \bar{Y} = \begin{bmatrix} 1 & \operatorname{diag}(Y)^T \\ \operatorname{diag}(Y) & Y \end{bmatrix} \succeq 0. \end{array}
$$

By the Schur complement theorem $\bar{Y} \succeq 0$ can be reformulated as

$$
Y - \operatorname{diag}(Y)\operatorname{diag}(Y)^T \succeq 0.
$$

Again it is well known that (SQ) and (SMC) are equivalent [19, 26]. In [18] we show that the canonical equivalence transformation is a scaling (see e.g. [33, 36]) of the problem by the invertible matrix

$$
Q = \begin{bmatrix} 1 & 0 \\ \frac{1}{2}e & \frac{1}{2}I_n \end{bmatrix}.
$$

Indeed,

$$
\bar{Y} = QXQ^T
$$

transforms feasible solutions of (SMC) for $n+1$ variables into feasible solutions of (SQ) for n variables and vice versa. The adjoint to the inverse of this transformation can be used to transform the constraint matrices of (SMC) into constraint matrices of (SQ) without affecting the right hand sides. This yields an equivalence transformation between the duals of the two representations of the feasible sets. If A is a constraint matrix of (SMC), then

$$
\bar{B} = Q^{-T}AQ^{-1}
$$

is the corresponding constraint matrix of (SQ). This transformation preserves most of the structure of A. For instance, rank one matrices $A = vv^T$ transform to rank one representations of $\bar{B}$. Since

$$
Q^{-1} = \begin{bmatrix} 1 & 0 \\ -e & 2I_n \end{bmatrix},
$$

sparsity properties of v are also carried over to $Q^{-T}v$. If (SQ) is modelled in accordance with these transformations all statements and techniques with respect to (SMC) apply to (SQ) as well.

We will now consider the "easiest" case of constrained quadratic 0-1 programming by adding one linear constraint to (QP). The quadratic knapsack problem reads

$$
\text{(QK)} \quad \begin{array}{rl} \max & y^T By \\ \text{s.t.} & a^T y \le b \\ & y \in \{0,1\}^n. \end{array}
$$

Without loss of generality we may assume that $0 < a_i \le b$ for $i = 1, \dots, n$. For $a_i < 0$ we can flip y_i to $1 - y_i$, for $a_i \in \{0\} \cup (b, \infty)$ we can set the corresponding variable to zero. Here, we assume that $a_i < b$, since $a_i = b$ allows to decompose the problem. The semidefinite relaxation of (QK) is based on (SQ), it remains to specify a good representation of the linear knapsack constraint $a^T y \leq b$ within the quadratic model. Because $y_i = y_i^2$ the linear constraint can be modeled on the diagonal,

$$\text{(SQK1)} \qquad \begin{array}{rl} \text{Maximize} & \langle C, Y \rangle \\ \text{subject to} & \langle \text{Diag}(a), Y \rangle \leq b \\ & Y - \text{diag}(Y)\text{diag}(Y)^t \succeq 0. \end{array}$$

A second representation is based on the observation that for all feasible points $|a^T y| \leq b$, therefore $a^T y y^T a \leq b^2$. Replacing yy^T by Y we obtain the *square representation*,

$$\text{(SQK2)} \qquad \begin{array}{rl} \text{Maximize} & \langle C, Y \rangle \\ \text{subject to} & \langle aa^t, Y \rangle \leq b^2 \\ & Y - \text{diag}(Y)\text{diag}(Y)^t \succeq 0. \end{array}$$

Finally, since $y_i \in \{0, 1\}$ the constraint must remain valid under multiplication with y_i or $(1 - y_i)$,

$$y_i a^T y \leq b y_i \qquad \text{or} \qquad (1 - y_i) a^T y \leq b(1 - y_i).$$

Replacing the mixed terms $y_i y_j$ by y_{ij} and the linear terms y_j by y_{jj} yields an inequality that we refer to as the y_i*-representation* or $(1 - y_i)$*-representation* of $a^T x \leq b$. Using these representations we can form a third relaxation.

$$\text{(SQK3)} \qquad \begin{array}{rll} \text{Maximize} & \langle C, Y \rangle & \\ \text{subject to} & \sum_{j=1}^n a_j y_{ij} \leq b y_{ii} & i = 1 \ldots n \\ & \sum_{j=1}^n a_j (y_{jj} - y_{1j}) \leq b(1 - y_{11}) & \\ & Y - \text{diag}(Y)\text{diag}(Y)^t \succeq 0. & \end{array}$$

The $y_i/(1 - y_i)$-representations are well known to yield powerful relaxations (see [28, 3]). Indeed, there is a clear hierarchy between relaxations (SQK1) to (SQK3).

Lemma 4.1 *[22] The feasible set of (SQK3) is contained in the feasible set of (SQK2), the feasible set of (SQK2) is contained in the feasible set of (SQK1).*

From a computational point of view (SQK2) is not only better than (SQK1) but also faster to compute when there are many inequalities (because of its representation as a dyadic product). Although (SQK3) is a better relaxation than (SQK2), we choose (SQK2) as initial relaxation, because it involves only one additional constraint in contrast to $n + 1$. Moreover, numerical results indicate that (SQK2) is not much worse than (SQK3) (see Table 1).

Computational experiments reveal that with the addition of a few y_i-representations to (SQK2) we achieve the quality of the relaxation (SQK3). We point out that y_i and $(1-y_i)$-representations can be written as rank two and rank four

			relative gap (%)			time(sec)		
dim	rhs	feas.	SQK1	SQK2	SQK3	SQK1	SQK2	SQK3
30	450	1580	41	17	13	1	1	2
	512	1802	39	20	17	1	1	2
	600	2326	24	12	11	1	1	2
45	450	2840	16	8.7	8.4	3	3	6
	512	3154	30	12.7	12.7	3	3	6
	600	3840	22	8.2	8.2	3	3	6
47	450	1732	7	5.9	5.8	3	3	7
	512	1932	30	12	11	3	3	7
	600	2186	31	18	17	3	3	7
61	450	26996	3.7	2.4	2.4	8	9	17
	512	29492	2.9	2.0	2.0	9	9	18
	600	32552	2.6	1.9	1.9	8	9	17

Table 1: Comparison of relaxations (SQK1), (SQK2), and (SQK3). The instances are based on compiler design problems taken from [17]. *dim* is the number of 0-1 variables, each problem was examined for the three capacities of column *rhs*. *feas.* gives the best solution we know. The relative gap is computed via $100 \cdot (\frac{\text{upper bound}}{\text{feas.}} - 1)$. All times refer to a Sun SPARCstation 10 Model 41.

matrices by exploiting the structure of $\bar{Y}$. Multiplication with y_i corresponds to

$$\left\langle \frac{(a - be_i)e_i^T + e_i(a - be_i)^T}{2}, \bar{Y} \right\rangle \leq 0$$

and multiplication with $(1 - y_i)$ to

$$\left\langle \frac{ae_0^T + e_0 a^T - [(a - be_i)e_i^T + e_i(a - be_i)^T]}{2}, \bar{Y} \right\rangle \leq b.$$

For all indices i and j with $a_i + a_j > b$ we include the constraint $y_{ij} = 0$ in the relaxation. This is not only a strengthening of the relaxation but also ensures the existence of strictly feasible solutions throughout the cutting plane algorithm. Suppose we would not include one such constraint $y_{ij} = 0$. Then the separation routines might return the triangle inequality $y_{ij} \geq 0$ and the knapsack specific cutting plane $y_{ij} \leq 0$. In this case there would be no strictly feasible solution satisfying both inequalities, the interior point algorithm might fail.

Since we included these constraints, the convex combination of feasible 0-1 solutions comprising up to two elements yields a point in the (relative) interior of the convex hull of all feasible 0-1 solutions which is also strictly feasible for all relaxations building on this initial relaxation. This point is usually not close to the center of the polytope and hence not well suited for selecting "good" violated inequalities. To generate a point closer to the center we include a number of solutions with large support in the convex combination. These are constructed, starting from any pair of one or two items, by iteratively adding elements with cyclically increasing indices. The resulting point is used to start the interior point algorithm and to choose among several violated inequalities by the geometric approach that we introduced for max-cut.

Generic inequalities

A generic way to improve initial relaxations is to add valid inequalities for unconstrained 0-1 quadratic programming. These are certainly valid for constrained cases as well. In practice we do this with constraints corresponding to the triangle inequalities in the max-cut setting,

$$\begin{array}{rcl} y_{ij} & \geq & 0 \\ y_{ij} & \leq & y_{ii} \\ y_{ii} + y_{jj} & \leq & 1 + y_{ij} \\ y_{ik} + y_{jk} & \leq & y_{kk} + y_{ij} \\ y_{ij} + y_{ik} + y_{jk} + 1 & \geq & y_{ii} + y_{jj} + y_{kk}. \end{array}$$

This generic approach already leads to significant improvements of the initial relaxation (see columns *SQK2* and $\triangle$-*ineq.* of Table 2).

These inequalities alone do in general not suffice to close the gap between feasible solution and upper bound. One also needs problem specific cutting planes.

Linear Cutting planes.

For $a \in \mathbb{N}^n$, let

$$P_{LK} \; := \; \operatorname{conv}\left\{ y \in \{0,1\}^n : \sum_{i \in N} a_i y_i \leq b \right\}$$

be the linear knapsack polyhedron. A first class of inequalities is constructed by choosing a subset T of all elements that fit into the knapsack. These retain their weights as coefficients. Each remaining element is assigned, as a new weight, the amount by which the weight of the item and the subset exceeds the knapsack capacity. This value coincides with the coefficient that one obtains from lifting the inequality $\sum_{i \in T} a_i y_i \leq \sum_{i \in T} a_i$ according to any ordering of the items not contained in T.

Lemma 4.2 *[38]***(weight inequalities)** *Let $T \subseteq \{1, \ldots n\}$ with $r := b - \sum_{i \in T} a_i \geq 0$ be given. The* weight inequality with respect to T

$$\sum_{i \in T} a_i y_i + \sum_{i \in \{1,\ldots,n\} \setminus T} \max\{0, (a_i - r)\} y_i \leq \sum_{i \in T} a_i$$

is valid for P_{LK}.

dim	rhs	feas. sol.	SQK2 %	$\triangle$-ineq. %/mm:ss	lin. cuts %/mm:ss	matching %/mm:ss
30	450	1580	17	0.23	$\triangleright$ 00:19	$\triangleright$ 00:24
	512	1802	20	5.60	0.35	$\triangleright$ 26:10
	600	2326	12	2.64	$\triangleright$ 1:55	$\triangleright$ 1:31
45	450	2840	8.7	2.90	$\triangleright$ 13:54	$\triangleright$ 13:43
	512	3154	12.7	3.07	1.61	1.58
	600	3840	8.2	$\triangleright$ 20:16	$\triangleright$ 3:45	$\triangleright$ 3:02
47	450	1732	5.9	2.51	$\triangleright$ 15:02	$\triangleright$ 31:10
	512	1932	12	1.30	$\triangleright$ 16:59	$\triangleright$ 8:20
	600	2186	18	8.89	6.02	4.09
61	450	26996	2.4	0.40	$\triangleright$ 3:17	$\triangleright$ 3:23
	512	29492	2.0	1.34	$\triangleright$ 28:53	0.02
	600	32552	1.9	1.04	0.42	0.33

Table 2: Improvement of the bound by adding cutting planes for the examples of Table 1. The initial relaxation is (SQK2). For each solution we eliminate inequalities with small dual variables, add n violated inequalities, and iterate. Relative gaps refer to the bound after 30 minutes of computation time. If the relative gap was closed to $5 \cdot 10^{-6}$ within this time limit, we mark this by $\triangleright$ and give the computation time instead. In $\triangle$-*ineq.* the inequalities of (SQK3) and the triangle inequalities are used as cutting planes. In *lin. cuts* the knapsack specific linear cutting planes are separated, as well. In *matching* we consider all inequalities together, including the matching constraints.

The idea of weight inequalities can be extended to more general cases. Instead of working with the original weights of the items, we introduce "relative" weights and derive an analogon of weight inequalities for these relative weights.

Lemma 4.3 *[38]***(extended weight inequalities)** *Let I and T be disjoint subsets of $\{1,\dots n\}$ satisfying $a_t \le a_i \le \sum_{j\in T} a_j$ for all $t \in T, i \in I$ and $r := b - \sum_{t\in T\cup I} a_i \ge 0$. Define relative weights*

$$\begin{aligned} w_t &= 1 && \forall t \in T,\\ w_i &= \min\left\{|S| : S \subset T, \textstyle\sum_{t\in S} a_t \ge a_i\right\} && \forall i \in I. \end{aligned}$$

For $z \in \{1,\dots,n\}\setminus(T\cup I)$ with lifted coefficient

$$w_z = \min\left\{\sum_{s\in S} w_s : S \subset (T\cup I), \sum_{s\in S} a_s \ge a_s - r\right\}$$

the extended weight inequality with respect to $T\cup I\cup\{z\}$

$$\sum_{i\in(T\cup I\cup\{z\})} w_i y_i \le \sum_{i\in(T\cup I)} w_i$$

is valid for P_{LK}.

This inequality can be extended by standard lifting techniques. For a given ordering of the items in $\{1,\dots,n\}\setminus(T\cup I\cup\{z\})$ one can step by step compute the maximum coefficient so that the lifted inequality remains valid in polynomial time.

Separation of weight and extended weight inequalities.

The most difficult part in the separation process is the construction of promising sets T (and I). Once these are fixed the coefficients can be computed by a lifting procedure. Each of these inequalities can be represented in the Y-space by the techniques discussed before. On account of Lemma 4.1 it suffices to consider the representations arising from multiplication with y_i or $(1-y_i)$.

It is quite unlikely that quadratic representations of weight or extended weight inequalities are violated by some Y^* if all the variables in the support of the inequality attain small values. This motivates the search for inequalities whose support is contained in the set of variables with large Y_{ij}-values. In a first step we interpret the variables Y_{ii} as variables $y_1,\dots,y_n$ of a linear problem, see lemmas 4.2 and 4.3. We sort them by value in nonincreasing order. For later reference we denote this order by the symbol $\succ_0$. We first determine the maximal number k_0 such that the first k_0 elements (with respect to $\succ_0$) have weight not greater than b. These k_0 indices form the set T of a weight inequality.

In order to construct extended weight inequalities we choose for every $k > k_0$ the k first elements with respect to $\succ_0$. This forms the set S_k. We sort the elements of S_k with respect to nondecreasing weights and determine the maximal number h such that the first h elements have weight not greater than b. This defines the set $S_k^{\subset}$. If $k > k_0 + 1$ and $S_k^{\subset}$ is contained in $S_{k-1}^{\subset}$, we proceed to index $k+1$. Otherwise every partition of $S_k^{\subset}$ into feasible sets T and I according to Lemma 4.3 gives rise to an extended weight inequality: We lift the elements not in $S_k^{\subset}$ following the order $\succ_0$; the item z is the first element of this order.

To generate promising starting sets for y_j-representations of linear knapsack inequalities it seems reasonable to work with variables Y_{ij} rather than Y_{ii}. The procedures outlined before can be easily adapted to this situation. One replaces $\succ_0$ by an appropriate order. Experimental results show the efficacy of this approach (see column *lin. cuts* of Table 2).

Quadratic cutting planes

The linear knapsack polytope P_{LK} is a convenient starting point for deriving cutting planes because it is well studied in the literature. However, the correct object to study is the corresponding quadric polytope

$$P_{QK} := \mathrm{conv}\{yy^T \,:\, y \in \{0,1\}^N, a^T y \le b\}.$$

The following lemma introduces a family of polytopes which are relaxations of P_{QK}.

Lemma 4.4 *[22] Let $N_1, \ldots, N_k$ be a partition of $\{1, \ldots, n\}$. For every $v \in \{1, \ldots, k\}$ we choose a spanning tree (N_v, T_v) in the complete graph $K(N_v)$ on the node set N_v. By deg_i^v we denote the degree of node i in the tree (N_v, T_v). The polyhedron*

$$\mathrm{conv}\left\{ Y \in S_n \,:\, y_{ij} \in \{0,1\}, \sum_{v=1}^{k} (\sum_{i \in N_v} a_i)[\sum_{ij \in T_v} y_{ij} + \sum_{i \in N_v} (1 - deg_i^v) y_{ii}] \le b \right\}$$

contains all the feasible points of P_{QK}.

These polytopes may be interpreted as linear 0-1 knapsack polytopes defined over the upper triangle of Y. Hence we can use the same machinery as in the linear case to derive valid inequalities for any of these polytopes. Since these polytopes are relaxations of P_{QK}, their valid inequalities remain valid for P_{QK}.

In the special case when the sets N_i of Lemma 4.4 contain either one or two elements, we may associate every spanning tree with an isolated node or an edge. Collecting all isolated nodes in the set V and all edges in the set E, the inequality of Lemma 4.4 reads

$$\sum_{ij \in E} (a_i + a_j) y_{ij} + \sum_{i \in V} a_i y_{ii} \le b. \tag{2}$$

We call an inequality of this form a *matching knapsack constraint.*

Separation of quadratic cutting planes

Our implementation includes a separation routine for matching knapsack constraints only. It is uses the following ingredients: We compute a maximum weight matching[1] on a principal submatrix of Y. The support of the principal submatrix is determined by the indices in S_k, $k > k_0$, see Subsection "Separation of weight and extended weight inequalities." For each S_k, $k > k_0$, our routine returns the maximum weight matching in form of sets E and V of (2). This defines the knapsack polyhedron

$$\operatorname{conv}\left\{x \in \{0,1\}^{|E|+|V|} : \sum_{ij \in E} (a_i + a_j)x_{ij} + \sum_{i \in V} a_i x_i \leq b\right\}.$$

We apply the separation routines for weight and extended weight inequalities for the point with components Y_{ij}, $ij \in E$ and Y_{ii}, $i \in V$.

Early separation and restart fails.

Although we know a feasible starting point which seems to be reasonably centered with respect to P_{QK}, the early separation approach described for max-cut fails in the case of the quadratic knapsack problem. The reason for this failure is that the analytic center of an initial relaxation is in general not in the interior of P_{QK}. Even when starting from a feasible point inside P_{QK} the iterates leave P_{QK} early on their way to the central path. At this early stage the central path may guide the iterates into a direction that differs substantially from the cost function. The inequalities cutting off the current iterate may not cut off the optimal solution of the relaxation. In consequence we only call our separation procedures for the optimal solution of the relaxation.

For the same reason the two approaches for restarting that we suggested for max-cut do not work for the quadratic knapsack problem. Since violated inequalities may cut off large parts of the old central path the stored iterates may not be feasible or may not be close to the new central path any longer. Similarly, the straight line towards the feasible starting point will have little in common with the central path either. Indeed, the most successful strategy was to restart every time from scratch.

5 Conclusions

We presented interior point based implementations of cutting plane algorithms for semidefinite relaxations of max-cut and quadratic 0-1 knapsack. Both problems are strongly related yet exhibit significant differences in their algorithmical behavior. The conclusions we can draw from these two implementations are ambivalent.

[1] We solve the maximum weight matching problem using routines from the LEDA package [31].

On the one hand semidefinite relaxations of constrained quadratic 0-1 programming problems are of high quality in practice. This is true even for generic relaxations without additional problem specific knowledge. The improvement obtained by adding the problem independent triangle inequalities to the initial semidefinite relaxation of the quadratic knapsack problem indicates that any progress with respect to the separation of max-cut will be of general use. However, it is already unlikely that, for instance, the clique inequalities can be separated efficiently.

On the other hand the solution of semidefinite relaxations via interior point methods is very expensive because each iteration requires the factorization of a dense positive definite matrix whose dimension equals the number of constraints. Therefore the problem of selecting a small efficient set of cutting planes from the large number of violated inequalities is of special importance. Although the heuristic rules that we presented seem to work surprisingly well, there is still a need for theoretical investigations of the quality of cutting planes and relaxations.

If the central path remains within the underlying combinatorial polyhedron for a long time the possibility of early separation and fast restart makes interior point methods an attractive choice in spite of the high computational cost. However, the generic semidefinite relaxation for constrained quadratic 0-1 programming does not have this property.

The knapsack specific cutting planes are of general value since they can be applied whenever some constraint with positive coefficients appears in some constrained quadratic 0-1 programming problem. If several inequalities are present it may be difficult or impossible to keep all coefficients positive. What are good initial relaxations in this case? The presence of several inequalities offers a number of other possibilities, as well.

For constrained quadratic 0-1 programming in general it will not be possible to construct an initial feasible solution. In this case an infeasible method or a skew symmetric embedding [6] has to be used. Is it possible to develop good restart procedures for these methods?

Real world applications of quadratic 0-1 programming typically involve thousands of variables and have sparse cost matrices. As of today interior point methods do not allow to exploit structure or sparsity of the cost matrix in semidefinite relaxations and time limits bound the number of constraints. Therefore the solution of such problems with interior point methods is still far away. For these applications other algorithms, yielding rough approximations to the optimal solutions only, may turn out to be more successful. None the less interior point methods form an ideal basis for studying the relationship between a combinatorial optimization problem, its semidefinite relaxation, and the corresponding cutting planes.

References

[1] F. Alizadeh. Interior point methods in semidefinite programming with ap-

plications to combinatorial optimization. *SIAM J. Optimization*, 5(1):13–51, 1995.

[2] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton. Primal–dual interior–point methods for semidefinite programming. Draft presented at the XV symposium on mathematical programming, Computer Science Department, Courant Institute of Mathematical Sciences, New York University, New York, 1994.

[3] E. Balas, S. Ceria, and G. Cornuejols. A lift-and-project cutting plane algorithm for mixed 0/1 programs, *Mathematical Programming*, 58:295–324, 1993.

[4] F. Barahona, M. Grötschel, and R. Mahjoub. Facets of the bipartite subgraph polytope, *Math. Oper. Res.*, 10:340–358, 1985.

[5] B. Chor and M. Sudan. A geometric approach to betweenness. In *ESA '95 Proceedings*, volume 979 of *Lecture Notes in Computer Science*, pages 227–237. Springer, 1995.

[6] E. De Klerk, C. Roos, and T. Terlaky. Initialization in semidefinite programming via a self-dual skew-symmetric embedding. Report 96-10, Faculty of Technical Mathematics and Informatics, Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands, 1996.

[7] C. Delorme and S. Poljak. Laplacian eigenvalues and the maximum cut problem. *Mathematical Programming*, 62:557–547, 1993.

[8] C. De Simone. The cut polytope and the boolean quadric polytope. *Discrete Applied Mathematics*, 79:71–75, 1989.

[9] M. Deza, M. Laurent, and S. Poljak. The cut cone III: On the role of triangle facets. *Graphs and Combinatorics*, 8:125–142, 1992.

[10] M. Deza and M. Laurent. Cut Polyhedra and Metrics. Manuscript, LIENS - Ecole Normale Supérieure, 1996. To appear at Springer.

[11] A. S. El-Bakry, R. A. Tapia, and Y. Zhang. A study of indicators for identifying zero variables in interior-point methods. *SIAM Review*, 36(1):45–72, 1994.

[12] U. Feige and M. X. Goemans. Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT. In *Proceedings of the Third Israel Symposium on Theory of Computing and Systems*, pages 182–189, Tel Aviv, Israel, 1995.

[13] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization*, volume 920 of *Lecture Notes in Computer Science*, pages 1–13. Springer, May 1995.

[14] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42:1115–1145, 1995.

[15] G. H. Golub and C. F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, 2^{nd} edition, 1989.

[16] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 2^{nd} edition, 1988.

[17] E. Johnson, A. Mehrotra, and G. L. Nemhauser. Min-cut clustering, *Mathematical Programming*, 62:133–152, 1993.

[18] C. Helmberg. Fixing variables in semidefinite relaxations. Preprint SC-96-43, Konrad-Zuse-Zentrum Berlin, 1996.

[19] C. Helmberg, S. Poljak, F. Rendl, and H. Wolkowicz. Combining semidefinite and polyhedral relaxations for integer programs. In E. Balas and J. Clausen, editors, *Integer Programming and Combinatorial Optimization*, volume 920 of *Lecture Notes in Computer Science*, pages 124–134. Springer, May 1995.

[20] C. Helmberg and F. Rendl. Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. Preprint SC-95-35, Konrad-Zuse-Zentrum Berlin, 1995.

[21] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz. An interior–point method for semidefinite programming. *SIAM J. Optimization*, 6(2):342–361, May 1996.

[22] C. Helmberg, F. Rendl, and R. Weismantel. Quadratic Knapsack Relaxations Using Cutting Planes and Semidefinite Programming. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization*, volume 1084 of *Lecture Notes in Computer Science*, pages 175–189. Springer, June 1996.

[23] F. Jarre. An interior–point method for minimizing the maximum eigenvalue of a linear combination of matrices. *Siam J. Control and Optimization*, 31(5):1360–1377, Sept. 1993.

[24] S. E. Karisch and F. Rendl. Semidefinite programming and graph equipartition. Technical Report 302, Department of Mathematics, Graz University of Technology, Graz, Austria, Dec. 1995.

[25] M. Kojima, S. Shindoh, and S. Hara. Interior–point methods for the monotone linear complementarity problem in symmetric matrices. Research Report B–282, Department of Information Sciences, Tokyo Institute of Technology, Apr. 1994. Revised April 1995.

[26] M. Laurent, S. Poljak, and F. Rendl. Connections between semidefinite relaxations of the max-cut and stable set problems. CWI Report BS-R9502, CWI Amsterdam, The Netherlands, Jan. 1995.

[27] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, IT-25(1):1–7, Jan. 1979.

[28] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM J. Optimization*, 1(2):166–190, May 1991.

[29] D. K. R. Motwani and M. Sudan. Approximate graph coloring by semidefinite programming. In *FOCS 94*, pages 2–13, 1994.

[30] J. E. Mitchell and B. Borchers. Solving real-world linear ordering problems using a primal-dual interior point cutting plane method. *Annals of Operations Research*, 63:253–276, 1996.

[31] S. Näher and C. Uhrig. The LEDA User Manual Version R 3.3 beta. Max-Planck-Institut für Informatik, Im Stadtwald, Building 46.1, D–66123 Saarbrücken, Germany. (`http://www.mpi-sb.mpg.de/LEDA/leda.html`)

[32] Y. Nesterov and A. Nemirovskii. *Interior–Point Polynomial Algorithms in Convex Programming.* SIAM Studies in Applied Mathematics, Philadelphia, 1993.

[33] Y. Nesterov and M. J. Todd. Self-scaled barriers and interior-point methods for convex programming. Technical Report TR 1091, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York 14853, Apr. 1994. Revised June 1995, to appear in Mathematics of Operations Research.

[34] S. Poljak and F. Rendl. Nonpolyhedral relaxations of graph-bisection problems. *SIAM J. Optimization*, 5(3):467–487, 1995.

[35] S. Poljak and Z. Tuza. Maximum cuts and large bipartite subgraphs. *Dimacs Series in Discrete Mathematics and Theoretical Computer Science*, 20:181–244, 1995.

[36] L. Tunçel. Primal-dual symmetry and scale invariance of interior-point algorithms for convex optimization. CORR Report 96–18, University of Waterloo, Ontario, Canada, Nov. 1996.

[37] L. Vandenberghe and S. Boyd. A primal–dual potential reduction method for problems involving matrix inequalities. *Mathematical Programming, Series B*, 69(1):205–236, 1995.

[38] R. Weismantel. On the 0/1 knapsack polytope, Preprint SC-94-01, Konrad-Zuse-Zentrum Berlin, 1994. To appear in Mathematical Programming.

[39] H. Wolkowicz and Q. Zhao. Semidefinite programming relaxations for the graph partitioning problem. CORR Report, University of Waterloo, Ontario, Canada, Oct. 1996.

[40] Q. Zhao, S. E. Karisch, F. Rendl, and H. Wolkowicz. Semidefinite programming relaxations for the quadratic assignment problem. CORR Report 95/27, University of Waterloo, Ontario, Canada, Sept. 1996.

dim	rhs	knapsack	triangle	weight	ext. weight	matching
30	450	0	58	3	14	1
	512	5	155	14	12	0
	600	4	85	9	5	2
45	450	0	167	9	22	6
	512	3	281	6	24	0
	600	0	93	6	6	2
47	450	6	133	5	29	7
	512	5	71	23	22	0
	600	4	166	3	3	14
61	450	0	89	0	0	3
	512	2	202	10	16	1
	600	3	184	4	12	2

Table 3: Cutting planes used. The numbers correspond to the solution given in column *matching* of Table 2. *knapsack* refers to y_i-representations of the original knapsack constraint. *triangle* gives the number of triangle inequalities. *weight* is the number of weight inequalities in y_i-representation. *ext. weight* refers to the y_i-representations of extended weight inequalities, and *matching* to the number of matching inequalities.