The Bundle Method in Combinatorial Optimization

Ilse Fischer* Gerald Gruber† Franz Rendl* Renata Sotirov*‡

April 29, 2004

Key words: Bundle Method, Semidefinite Programming, Max-Cut.
AMS Subject Classification: 90C22, 90C27.

Abstract

We propose a dynamic version of the bundle method to get approximate solutions to semidefinite programs with a nearly arbitrary number of linear inequalities. Our approach is based on Lagrangian duality, where the inequalities are dualized, and only a basic set of constraints is maintained explicitly. This leads to function evaluations requiring to solve a relatively simple semidefinite program.

Our approach provides accurate solutions to semidefinite relaxations of the Max-Cut and the Equipartition problem, which are not achievable by direct approaches based only on interior-point methods.

1 Introduction

Many combinatorial optimization problems have a natural formulation in 0-1 variables, leading to either linear or quadratic problems in binary variables. To get tractable relaxations, one can use a (partial) description of the convex hull of integer solutions (polyhedral approach), or more recently, semidefinite programming.

After having formulated some basic semidefinite relaxation, it is usually possible to tighten it by adding additional linear inequalities valid for all 0-1 solutions. These combinatorially derived 'cutting planes' pose a serious challenge even to state of the art software for semidefinite programming, because the potential number of these constraints can be prohibitively large.

The most reliable methods to solve semidefinite programs are based on the interior-point idea, which means applying the Newton method to some variant of parameterized optimality conditions. The primal-dual path-following variants seem to yield the most reliable computational results, see [33].

The computational effort for one interior-point iteration is determined by the following factors:

- Maintaining positive definite matrices of some given order n.
- Linear algebra operations with matrices of order n, such as matrix multiplication or forming the inverse explicitly.

*Universität Klagenfurt, Institut für Mathematik, Universitätsstraße 65-67, A–9020 Klagenfurt, Austria, ({ifischer,rendl,rsotirov}@uni-klu.ac.at)

†Carinthia Tech Institute, Department of Geoinformation, Europastrasse 4, A-9524 Villach, Austria, (g.gruber@cti.ac.at)

‡Support from the Austrian Science Foundation FWF Project P12660-MAT is gratefully acknowledged.

- Solving a linear system of order given by the number m of constraints of the problem.

The first two factors can be bounded by $O(n^3)$ operations, independent of the number of constraints. The last factor depends heavily on the number m of constraints, and is at least $O(m^3)$ plus the effort to determine the matrix of the linear system to be solved. The main conclusion is therefore that the number m of constraints has to be limited, if interior-point methods are used to solve the relaxation.

In this paper we investigate alternative algorithmic approaches, where we still assume that the order n of the matrix space is reasonably small ($n \leq 1000$), but the number of constraints can be significantly larger.

Recently, Barahona and Anhil [5] and Barahona and Ladanyi [6] revived the idea of working with the Lagrangian dual to handle constraints $Ax \leq b$ only indirectly. The Lagrangian dual functional is nonsmooth, so one can use the algorithmic machinery from nonsmooth optimization to deal with it. In [5, 6] a simple version of the subgradient iteration scheme of Polyak [29] is refined to carry out the iterations. The key point with these methods is that they rely on a subroutine, which carries out a function and subgradient evaluation of the dual functional. Using the subgradient information, a new trial point is determined, and the process is iterated.

Contrary to the setup in [5, 6], we have a more general framework in mind, where the function evaluation itself may be nontrivial. Therefore it is important to provide an algorithmic tool, which works well even if the number of function evaluations is limited (to a few dozen function evaluations). We will use the bundle method instead of the subgradient scheme, to make efficient use of subgradient information, gathered during the computational process. It was recently shown [3] that the approach from [5, 6] can in fact be viewed as a simple variant of the bundle method.

It is the main purpose of this paper to show that a dynamic version of the bundle method provides an efficient machinery to approximately solve semidefinite problems with a nearly arbitrary number of inequalities. More precisely, we will concentrate on the Max-Cut and the Equipartition problem. For both these problems there exists a rather 'simple' semidefinite relaxation which can be refined by including additional cutting planes ($O(n^3)$ triangle inequalities for Max-Cut or $O(n^2)$ sign constraints for Equipartition). We demonstrate that these problems can be solved rather accurately for problems of medium size (number of vertices of the graphs no more than 500). To the best of our knowledge, such results are currently not available using other methods. We explain the main features of our version of the bundle method, which consists of repeatedly identifying new interesting constraints and removing old unimportant constraints. We also describe other types of problems where we think that the present approach might work well. We finally also show the practical limits of our approach.

Notation: We use standard notation for semidefinite programming. $\mathrm{tr}(A)$ denotes the trace of the square matrix A. The vector of all ones is e.

2 Large Semidefinite programs from combinatorial optimization

We develop our ideas on the following two graph optimization problems. Both are defined on an undirected edge-weighted graph G on vertex set $V := \{1, \ldots, n\}$. The relevant data are conveniently expressed through the weighted adjacency matrix $A = (a_{ij})$, where

$$a_{ij} = a_{ji} = w_e \text{ for edge } e = [ij] \in E,$$

$$a_{ij} = 0 \text{ if } [ij] \notin E.$$

Given A, we also introduce the Laplacian matrix $L = L_A$, associated to A, which is defined as

$$l_{ii} = \sum_k a_{ik}, \ \forall i, \ l_{ij} = -a_{ij} \ \ i \neq j.$$

The **Max-Cut Problem (MC)** asks to partition V into S and $V \setminus S$ in such a way that the total weight of the edges joining S and $V \setminus S$ is maximized. (If $S = \emptyset$ or $S = V$, the weight of the cut is defined to be 0, since no edges are cut in this case.)

Let us encode $S \subseteq V$ by setting $x_i = 1 \ \forall i \in S$ and $x_i = -1 \ \forall i \notin S$. It is a well-known fact, easy to verify, that the weight of the cut, given by $S \subseteq V$, $cut(S, V \setminus S)$ is given by

$$cut(S, V \setminus S) = \frac{1}{4} x^T L x.$$

Hence Max-Cut is equivalent to

$$z_{mc} := \max\{x^T L x : x \in \{-1, 1\}^n\}. \tag{1}$$

In case of the **k-Equipartition Problem (k-EP)** we make the assumption that the number n of vertices is a multiple of a given (integer) parameter k, $n = mk$. The problem now consists in partitioning V into k sets of cardinality m, so as to minimize the total weight of edges joining different partition blocks. Now we represent k-partitions $S_1, \ldots, S_k$ by $0-1$ matrices X of size $n \times k$ with $x_{ij} = 1$ if vertex $i \in S_j$, and $x_{ij} = 0$ otherwise. The problem now amounts to the following, see e.g. [11].

$$z_{ep} = \min\{\mathrm{tr} X^T L X : Xe = e, \ X^T e = \frac{n}{k} e, \ x_{ij} \in \{0, 1\}\}. \tag{2}$$

2.1 Semidefinite Relaxations

The formulation (1) of Max-Cut gives rise to the following relaxation, which has attracted a lot of attention due to the famous approximation analysis of Goemans and Williamson [13]. It is most easily derived through the trivial identity

$$x^T L x = \mathrm{tr} \ L(xx^T).$$

We replace xx^T by a new matrix variable X and observe that $x \in \{-1, 1\}^n$ implies in this case that $X \succeq 0$ and $\mathrm{diag}(X) = e$. Hence we obtain the following basic relaxation of (1):

$$z_{mc-basic} = \max\{\mathrm{tr} \ LX : \mathrm{diag}(X) = e, \ X \succeq 0\}. \tag{3}$$

This is a semidefinite program in the matrix variable X of order n, and n equality constraints. This relaxation was introduced by Delorme and Poljak [10] in a dual form. The primal version (3) can be found in [28]. In [13] it is shown that this relaxation has an error of no more than 13.82 %, i.e.

$$\frac{z_{mc-basic}}{z_{mc}} \leq 1.1382,$$

if all $a_{ij} \geq 0$. Nesterov [27] provides a weaker bound for a larger class of instances with $L \succeq 0$ by showing that

$$\frac{z_{mc-basic}}{z_{mc}} \leq 1.57.$$

It is not too difficult to see that the relaxation (3) can be strengthened by requiring X to satisfy the triangle inequalities, $X \in$ MET, see [28, 17]. By definition, $X \in$ MET if and only if

$$\begin{aligned} x_{ij} + x_{ik} + x_{jk} &\geq -1, \\ x_{ij} - x_{ik} - x_{jk} &\geq -1 \\ -x_{ij} + x_{ik} - x_{jk} &\geq -1, \\ -x_{ij} - x_{ik} + x_{jk} &\geq -1 \ \forall i < j < k. \end{aligned}$$

Hence we get

$$z_{mc-met} = \max\{\text{tr } LX : \text{diag}(X) = e,\ X \in \text{ MET},\ X \succeq 0\}. \tag{4}$$

This is again a semidefinite program, but it has $4\binom{n}{3}$ triangle inequalities in addition to the n equations fixing the main diagonal of X to e. The computational effort to solve this problem is nontrivial, even for small n like $n \approx 100$, see e.g. [16].

Turning to $k-$equipartition, we find a similar situation. Linearizing again using $\text{tr}X^T LX = \text{tr}L(XX^T)$ and introducing a new variable Y for XX^T, we observe, see e.g. [1, 18] that Y has to satisfy the constraints $Y \succeq 0,\ \text{diag}(Y) = e,\ Ye = \frac{n}{k}e$.

This gives rise to the following basic semidefinite relaxation for $k-$EP, introduced in [18].

$$z_{ep-basic} := \min\{\text{tr } LY : Ye = \frac{n}{k}e,\ \text{diag}(Y) = e,\ Y \succeq 0\}. \tag{5}$$

In [18] it is observed that the condition $Ye = \frac{n}{k}e$ can be eliminated by parameterizing Y as $Y = \frac{1}{k}ee^t + VRV^T$. Here V is an arbitrary $n \times (n-1)$ matrix with its columns spanning the orthogonal complement of e, and R is symmetric of order $n-1$. This representation insures that Y has the eigenvector e to the eigenvalue $\frac{n}{k}$ and all the other eigenvectors are orthogonal to e. Since $Le = 0$ by the definition of the Laplacian L, we obtain the following equivalent formulation, see [18]:

$$z_{ep-basic} = \min\{\text{tr } (V^T LV)R : \text{diag}(VRV^T) = \frac{k-1}{k}e,\ R \succeq 0\}.$$

This basic relaxation is again a problem in the matrix variable R of order $n-1$ with n equations.

In [18], several refinements of this model are discussed. The simplest one includes the additional constraint $Y \geq 0$ elementwise.

$$z_{ep-nonneg} := \min\{\text{tr } LY : Ye = \frac{n}{k}e,\ \text{diag}(Y) = e,\ Y \geq 0,\ Y \succeq 0\}. \tag{6}$$

Including all $\binom{n}{2}$ sign constraints $y_{ij} \geq 0$ directly is computationally prohibitive for problems of medium size, like $n \approx 200$.

In both cases we have the situation that the basic semidefinite relaxation can be solved with reasonable effort for problem sizes n rather large. A Matlab routine to compute $z_{mc-basic}$ is for instance contained in [17] [1]. Typical computation times with this routine on randomly generated problems of various sizes are given in Table 1.

A Matlab routine that computes $z_{ep-basic}$ is given in [18]. The computation times are similar to Table 1. The situation is different, once we include additional constraints such as the triangle inequalities for Max-Cut or the sign constraints for k-EP. We conclude this section with a discussion of the limitations of standard interior-point methods to deal with semidefinite relaxations like (4) or (6).

[1] Available at the web-site: `http://www.math.uni-klu.ac.at/or/Software/`)

n	seconds
200	1.84
400	8.92
600	24.10
800	51.45
1000	99.27
1500	314.99
2000	714.21

Table 1: Computation times (seconds) to solve the SDP relaxation (3) on a PC (Pentium 4, 1.7 Ghz).

	$n = 100$	$n = 200$	$n = 300$
$m = 500$	21 (19)	34 (20)	49 (19)
$m = 1000$	103 (21)	136 (22)	164 (21)
$m = 1500$	304 (24)	358 (24)	422 (24)
$m = 2000$	643 (25)	763 (26)	816 (24)
$m = 2500$	1090 (24)	1313 (26)	1360 (24)

Table 2: Computation times (seconds) on a PC (Pentium 4, 1.7 GHz) to compute the semidefinite relaxation of Max-Cut for a graph with n nodes and m triangle inequalities. The number of interior point iterations is given in parentheses.

2.2 Limitations of Interior-Point methods

Let us consider solving (4) directly by interior point methods, as proposed e.g. in [14, 16]. The total number of inequalities in (4) is roughly $\frac{2}{3}n^3$, hence impossible to include directly for problems of interest ($n \geq 100$). On the other hand, there are only $\binom{n}{2}$ variables in (4), so the 'right' choice of active inequalities should not contain more than $\binom{n}{2}$ of the triangle inequalities. Unfortunately, this number is still too big. To illustrate how the number of triangle inequalities influences the computation times, we provide in Table 2 computation times to solve (4) with $n \in \{100, 200, 300\}$ vertices and $m \in \{500, 1000, 1500, 2000, 2500\}$ triangle constraints by interior point methods. The computation times clearly indicate that problems with significantly more than 2000 triangle constraints are computationally expensive to solve.

Another reason for potential difficulties with a direct approach lies in the problem of quickly identifying important inequalities, that is to say those which are active at the optimum. In [14] it was proposed to take the constraints most violated by the optimal solution X of (3), and solve the relaxation (4) with those constraints included, and iterate this process. This idea should work well, if a reasonable proportion of constraints violated by X would appear in the list of constraints active at the optimum. Unfortunately, our computational experiments do not indicate that this is likely to happen.

To elaborate on this point, we report results about the following experiment. We first solved the relaxation (4) for the problems g1d, g1s, spin5 and g2d given in the appendix.

Then we took the optimal solution X_{sdp} of the basic relaxation (3) and sorted all the triangle constraints according to violation with respect to X_{sdp}. Thus the first constraint is the most

graph	n	active for (4)	500	1000	2000	5000	inactive
g1d	100	1053	66	120	205	406	20
g1s	100	1371	33	63	118	253	199
spin5	125	1924	21	29	51	116	420
g2d	200	2108	56	94	158	302	24

Table 3: The active constraints of (4) are difficult to identify by looking at the most violated constraints of the basic relaxation. The last column gives the number of active constraints of (4), which are inactive at the basic relaxation.

violated one, and so on. In Table 3 we provide the number of active constraints for these problems at the optimal solution of (4) in column 3. We first observe that this number is much smaller than $\binom{n}{2}$.

We then checked, how many of the most violated constraints of (3) are among the active constraints at the optimal solution to (4). We looked separately at the 500, 1000, 2000 and 5000 most violated constraints (columns 4,5,6 and 7). Thus we see that among the 500 most violated constraints of X_{sdp} for g1d, only 66 are among the final active constraints. The situation is even worse for spin5, where among the 5000 most violated constraints of (3), only 116 are active at (4).

Finally, we also checked, how many of the active constraints of (4) are inactive at the optimal solution of (3) (column 8). Ideally, this number should be quite small, but unfortunately it is not in all our examples.

From these results it should not come as a surprise that solving (4) iteratively by including the most violated constraints of (3) takes a nontrivial computational effort.

As a possible alternative we therefore follow the philosophy of Lagrangian duality and try to approximate the Lagrangian dual using techniques from non-smooth optimization. We recall the necessary mathematical machinery and also the relevant technicalities in the following section.

3 Lagrangian Duality and the Bundle Method

3.1 Lagrangian Duality

To describe the semidefinite problems from before in a general setting, we first reformulate them as follows. It will be irrelevant that our problems are formulated in the space of symmetric matrices, hence we prefer expressing them simply as finite-dimensional (nonlinear) optimization problems. Using $x = vec(X),\ c = vec(L)$, we introduce the set $\mathcal{X} := \{x : x = vec(X), diag(X) = e, X \succeq 0\}$. In this case, (3) can equivalently be written as

$$\max\{c^T x : x \in \mathcal{X}\}. \tag{7}$$

A similar definition would give the basic relaxation of $k-$EP. Including the triangle constraints in (7) amounts to including the constraints

$$Ax \leq b$$

where A is a matrix of order $4\binom{n}{3} \times n^2$, with each row of A representing one of the triangle inequalities. Thus we end up with a problem of the following general form:

$$z = \max\{c^T x : x \in \mathcal{X}, Ax \leq b\}. \tag{8}$$

We assume that the problem without the inequalities is 'easy' to solve, but the inclusion of $Ax \leq b$ makes it computationally expensive, see the discussion in the previous section.

We introduce the Lagrangian

$$L(x, \gamma) := c^T x + \gamma^T (b - Ax) \tag{9}$$

and the dual functional

$$f(\gamma) := \max_{x \in \mathcal{X}} L(x, \gamma) = b^T \gamma + \max_{x \in \mathcal{X}} (c^T - A^T \gamma) x. \tag{10}$$

The following notation will be useful. We call a pair (γ, x) a **matching pair** for f, if $f(\gamma) = L(x, \gamma)$. The Lagrangian dual now is $z = \min_{\gamma \geq 0} f(\gamma)$. Note that we assume $\mathcal{X}$ to be nice in the sense that $\max_{x \in \mathcal{X}} \min_{\gamma \geq 0} L(x, \gamma) = \min_{\gamma \geq 0} f(\gamma)$ holds.

The problem now consists in finding $x^* \in \mathcal{X}$ and $\gamma^* \geq 0$ such that

$$|f(\gamma^*) - c^T x^*| \approx 0 \text{ and } \| \max\{0, Ax^* - b\} \| \approx 0.$$

Prosaically speaking, we would like to find x^* which is nearly feasible and nearly optimal. The dual variables γ^* serve to estimate z from above. We note that evaluating f for some γ amounts to solving a problem of type (7). Our idea is now to get a good approximation of $\min f(\gamma)$ with only a limited number of function evaluations.

The use of the Lagrangian dual to deal with primal constraints indirectly has a long history in combinatorial optimization, see for instance [4]. Traditionally, the dual functional is based on a Linear Programming model, i.e. the set $\mathcal{X}$ is polyhedral. We will see below that in the case of Max-Cut and Equipartition, it is advantageous to use semidefinite relaxations.

3.2 The bundle method: basic ideas

The bundle method goes back at least to the 1970's, see e.g. [20, 19, 31]. It was originally developed to minimize a nonsmooth convex function $f(\gamma)$ over $\gamma \in \mathbb{R}^n$. The function f is assumed to be given by an oracle, which, for some input γ returns the function value $f(\gamma)$ and a vector g contained in the subdifferential of f at γ, $g \in \partial f(\gamma)$. Imposing the sign constraints $\gamma \geq 0$, as required in our situation, does not make the problem much harder. We will use the bundle method taylored for our problem, hence it is useful to briefly recall its basic ideas and practical issues.

If $f(\gamma)$ is given as before, $f(\gamma) = b^T \gamma + \max_{x \in \mathcal{X}} (c - A^T \gamma)^T x$, the bundle method is most conveniently explained as follows. For given $\gamma \geq 0$ we solve (10) returning $f(\gamma)$ and x, such that (γ, x) is a matching pair. A subgradient $g \in \partial f(\gamma)$ is then implicitly given by $g = b - Ax$, as can easily be verified.

We start with some initial $\gamma_{start} \geq 0$, for instance $\gamma_{start} = 0$ and evaluate f. Thus initially we have a matching pair $(\gamma_{start}, x_{start})$ with $f(\gamma_{start}) = L(x_{start}, \gamma_{start})$.

The algorithm is iterative and maintains in each iteration a currently best approximation $\hat{\gamma}$ to the minimizer of f and a sequence $X = (x_1, \ldots, x_k)$ where each $x_i \in \mathcal{X}$ and $(\hat{\gamma}, x_k)$ is a matching pair.

To describe a general step, we assume to have $X = (x_1, \ldots, x_k)$ and $\hat{\gamma}$, with $(\hat{\gamma}, x_k)$ a matching pair. Given X, we compute the following information:

$$g_i := b - Ax_i, \; G = (g_1, \ldots, g_k), \; \phi_i := c^T x_i \text{ and } F = (\phi_1, \ldots, \phi_k)^T. \tag{11}$$

The bundle method now combines two ideas to determine a new trial point γ_{test}. First, the function $f(\gamma)$ is approximated by

$$f_{appr}(\gamma) := \max\{L(x,\gamma) : x \in \mathrm{conv}(x_1,\ldots,x_k)\}.$$

Since $x \in \mathrm{conv}(x_1,\ldots,x_k)$ if and only if $\exists \lambda \in \Lambda := \{\lambda \in I\!R^k : \lambda \geq 0,\ e^T\lambda = 1\}$ with $x = X\lambda$, we can write out the definition of f_{appr} to get

$$f_{appr}(\gamma) = \max_{\lambda\in\Lambda} b^T\gamma + (c - A^T\gamma)^T(X\lambda) = \max_{\lambda\in\Lambda} F^T\lambda + \gamma^T G\lambda,$$

using the definition of F and G, and noting that $G\lambda = b - AX\lambda$.

The second ingredient of the bundle method consists in the proximal point idea, which penalizes displacements from the currently best point $\hat{\gamma}$ with a term proportional to $\|\gamma - \hat{\gamma}\|^2$.

In summary therefore, the bundle method asks to find a new trial point $\gamma_{test} \geq 0$ by minimizing, for some prescribed parameter $t > 0$ the function

$$f_{appr}(\gamma) + \frac{1}{2t}\|\gamma - \hat{\gamma}\|^2 \tag{12}$$

over the set $\gamma \geq 0$. We obtain the following dual problem to be solved.

$$\begin{aligned}
&\min_{\gamma\geq 0}\max_{\lambda\in\Lambda} F^T\lambda + (G\lambda)^T\gamma + \frac{1}{2t}\|\gamma - \hat{\gamma}\|^2 \\
&\qquad = \min_{\gamma}\max_{\lambda\in\Lambda,\eta\geq 0} F^T\lambda + (G\lambda)^T\gamma + \frac{1}{2t}\|\gamma - \hat{\gamma}\|^2 - \gamma^T\eta \\
&\qquad\qquad = \max_{\lambda\in\Lambda,\eta\geq 0}\min_{\gamma} F^T\lambda + (G\lambda)^T\gamma + \frac{1}{2t}\|\gamma - \hat{\gamma}\|^2 - \gamma^t\eta
\end{aligned}$$

The inner minimization problem is convex quadratic in γ, hence we can replace it by insuring $\frac{\partial}{\partial\gamma}(.) = 0$ which is equivalent to $\gamma = \hat{\gamma} + t(\eta - G\lambda)$. Therefore minimizing $f_{appr}(\gamma) + \frac{1}{2t}\|\gamma - \hat{\gamma}\|^2$ is dual to the following maximization

$$\max_{\lambda\in\Lambda,\eta\geq 0} (F + G^T\hat{\gamma})^T\lambda - \frac{t}{2}\|\eta - G\lambda\|^2 - \hat{\gamma}^T\eta, \tag{13}$$

yielding λ and η. The minimizing γ of (12) is then given by $\gamma = \hat{\gamma} + t(\eta - G\lambda)$.

The final problem in λ and η is a convec quadratic optimization problem. We solve it approximately by keeping one set of the variables constant: keeping η constant results in a convex quadratic problem over the set Λ. Keeping λ constant allows to solve for η coordinatewise, see for instance [15]. Thus we start with $\eta = 0$, solve for λ which we then keep constant to solve for η and iterate this process several times to get (approximate) solutions λ, η of (13). Using these estimates λ and η we get the new trial point

$$\gamma_{test} = \hat{\gamma} + t(\eta - G\lambda).$$

As a byproduct of minimizing (12) we also get a 'primal' point

$$\hat{x} := X\lambda \tag{14}$$

in the convex hull of the current bundle X.

Note also that $\lambda_i = 0$ implies that the column x_i of X has no influence on the maximization. We use this observation and remove any columns of X with corresponding $\lambda_i = 0$.

The bundle method now asks to evaluate f at γ_{test}, producing the function value $f(\gamma_{test})$ and the matching pair $(\gamma_{test}, x_{test})$. Using some standard criteria, we decide whether or not γ_{test} becomes the currently best point, we update our information and start a new iteration.

One iteration of the bundle method therefore has the following two significant steps:

- Solve (12) approximately by first solving a sequence of convex quadratic problems in the k variables λ. This gives the new trial point γ_{test}.
- Evaluate f at γ_{test}.

In our applications, the last step is the dominating operation in each iteration. At termination, the bundle method returns $\hat{\gamma}$ as an approximate dual solution, and $\hat{x}$, see (14) as an approximate primal solution. We note, that these do not form a matching pair, but the general convergence theory for the bundle method, see e.g. [21, 31, 12], insures that under appropriate stopping conditions these vectors converge towards an optimal primal-dual solution pair of (8).

We mention that the bundle method or other subgradient based techniques are not the only possibilities to solve the Lagrangian dual. The 'Analytic Center Cutting Plane Method (ACCPM)' [2, 26] could as well be used. For our purposes however, this method is too time-consuming, because inequalities are maintained explicitely.

3.3 A dynamic version of the bundle method

Contrary to the "classical" use of the bundle method, where the function $f(\gamma)$ to be minimized is considered to be fixed, we have a more flexible version in mind, where f will change in the course of the algorithm, see also [4, 5]. To get the initial function $f(\gamma)$ we solve

$$\max\{c^T x : x \in \mathcal{X}\}$$

yielding an initial maximizer x^*. In principle we could now define $f(\gamma)$ by dualizing all the constraints $Ax \leq b$. Recall however, that the number m of constraints may be substantially larger than the dimension of the problem. To maintain efficiency, we are interested only in those constraints, which are likely to be active at the optimum. Therefore we look at $r^* := b - Ax^*$ and set $r_{\min} := \min\{r_i^*\}$. In the unlikely event that $r_{\min} \geq 0$, x^* is optimal for (8) and we are done. Otherwise $r_{\min} < 0$ and we consider now only those constraints from $Ax \leq b$ which are 'badly' violated by x^* to define f.

Specifically, let $0 < \alpha < 1$ and set

$$I := \{i : r_i^* \leq \alpha\ r_{\min}\}.$$

We denote by A_I the submatrix of A with rows indexed by I, a similar definition holds for b_I. For notational convenience we define

$$f_I(\gamma_I) := \max_{x \in \mathcal{X}} c^T x + (b_I - A_I x)^T \gamma_I.$$

Thus we start by minimizing f_I using the bundle method described above. In order to maintain computational efficiency, we include as an additional stopping condition an upper bound on the number of iterations. Therefore, when the method stops, returning some $\hat{\gamma}_I$ and some $\hat{x}$, it is likely that $\hat{\gamma}_I$ is still far from the true minimizer of f_I. To continue, we need to decide on the following two issues:

- Which of the constraints in I should be kept?
- Are there additional constraints that should be added to I?

We use $\hat{\gamma}_I$ to answer to the first question. A large value $\hat{\gamma}_i$ indicates that the constraint i is binding and hence should not be removed. $\hat{\gamma}_i = 0$ indicates that constraint i may be inactive, and therefore could be removed. In summary, we use $\hat{\gamma}_I$ to purge I by removing constraints i with a value $\hat{\gamma}_i$ smaller than a prescribed fraction of $\max(\hat{\gamma}_I)$.

It is less obvious to decide which new constraints should be added to I. In our experiments we found that using a convex combination of $\hat{x}$ and the previous point $\hat{x}_{old}$ to identify violated constraints gives the most satisfactory behaviour of the algorithm.

4 Solving the relaxations with the bundle method

We now turn to computational experiments with our version of the bundle method. We consider Max-Cut and Equipartition problems in the following two subsections separately.

4.1 Solving the semidefinite relaxation (4) of Max-Cut

To make our experiments reproduceable and accessible to other researchers, we generate our test instances using the graph generator rudy, introduced by Giovanni Rinaldi. It is available under `http://www-user.tu-chemnitz.de/~helmberg/rudy.tar.gz`.

We generate three types of instances, all with edge weights c_{ij} randomly chosen from $\{1, -1\}$: dense graphs, sparse graphs with edge density of 10 %, and three-dimensional grid graphs. Finally we take some graphs from the literature, see [16].

The calls to rudy to generate these graphs are given in the appendix. We first give some statistics about these graphs in Table 4, including the number of vertices (n), the number of edges ($|E|$) and the value of the best cut known to us. We are mostly interested in difficult instances, where neither the approximation results of Goemans and Williamson ($a_{ij} \geq 0$) nor the theory of Nesterov ($L \succeq 0$) applies. We will show that even for this difficult class of Max-Cut instances, the relaxation (4) provides a good (a-posteriori) approximation to the true value of the Max-Cut problem, and in particular constitutes a nontrivial improvement of the basic relaxation (3).

We apply the bundle method with the modifications previously described to these graphs. For problems with $n \leq 512$ we have set a limit of 100 function evaluations, for the bigger problems ($729 \leq n \leq 1000$) we allowed 50 function evaluations. We use the graphs on 2000 vertices to indicate also the limits of the present approach. For these graphs we allow only 30 function evaluations to estimate the optimum of (4).

In Table 4 we summarize our computational results. The table identifies the graphs (in column 1), then gives the initial bound (optimal value of (3)) and the final bound found after the maximum number of function evaluations was reached. We also include the gap (in %) of the bounds relative to the value of the best known cut, $gap = 100(bound - cut)/cut$. The last two columns contain the number of $\gamma_i > 0$, i.e. the number of inequalities actually active at the stop, and the total computation times in minutes on a PC (Pentium 4, 1.7 Ghz).

Accuracy: The first question is to estimate, how good our bounds approximate the correct optimal value of (4). To estimate the quality of our bounds, we use the final set of active constraints of the bundle method, to solve the relaxation with these constraints by interior-point methods. We kept adding violated constraints, until the largest violation of all triangle constraints was less than 0.02. We collect the results in Table 5 for some of the smaller problems. Comparing the bound obtained after 100 iteration of the bundle method with the true optimal value of (4), we conclude that the bundle method gives reasonably accurate approximations of (4). We may therefore safely assume that the results of Table 4 give tight approximations to (4).

Bound versus Bundle Iterations: We now take a closer look at the development of the bounds during the iterations of the bundle method. In Figure 1 we provide a graphical development of the results for the three-dimensional grid graphs. The figure shows that the biggest improvement is obtained in the first few iterations with a flat tail. This is an attractive feature, because it allows the user to trade quality against computation time.

Big graphs: Looking at the graphs on 2000 nodes, we also see the limits of the present approach. The function evaluation itself is already quite expensive, see also Table 1, leading to computation times of several hours. We note however that even for these large problems, the inclusion of triangle inequalities provides a significant improvement of the initial basic semidefinite programming bound.

LP versus SDP: For curiosity, we also compare solving (4) with and without the semidefiniteness constraint on X. Optimizing over the triangle inequalities alone amounts to solving a Linear Program, so one might think that this is a straight-forward task. Unfortunately, this is not the case. In the last column of Table 5 we provide this LP bound for all the instances where we managed to compute the optimal solution of the triangle relaxation. We note that the linear relaxation is quite strong in case of the grid graphs, but tends to be very weak for all other instances. We find it surprising that solving (at least approximately) the relaxation (4) seems easier than solving the relaxation with the semidefiniteness condition omitted. One reason for this may be the fact that $X \succeq 0$ with $diag(X) = e$ bounds the largest violation of the triangle inequalities in a nontrivial way. Without any constraint on X except $-1 \leq x_{ij} \leq 1$, we get $x_{ij} + x_{ik} + x_{jk} + 1 \geq -2$. If $X \succeq 0$ holds, then $x_{ij} + x_{ik} + x_{jk} + 1 \geq -0.5$ can easily be shown to hold.

Improvement of (3) using the triangle inequalities: We now take a closer look at the question of how much the triangle inequalities help to improve the basic semidefinite relaxation (3). Comparing the initial gap and the final gap we note that the improvement is particularly drastic for the highly structured 3-dimensional grid graphs spin$x$ and for G11, which is a 2-dimensional grid. It is also quite remarkable in case of G18 and G39, which are sums of two planar graphs. The improvement is only marginal in case of larger unstructured graphs, like g4d, g5d, g5s, G1, G6, G22 and G27.

Generating Cuts: Once we have solved our relaxation up to the desired accuracy we have available a primal matrix X which ideally should be a rank-one matrix and hence represent a cut. In general this is not the case. We use X for the Goemans-Williamson hyperplane rounding technique [13] and report the best cut x found this way. This cut x is locally improved by checking all possible moves of a single vertex to the opposite partition block. We iterate now with the matrix $.5X + .5xx^T$ and repeat as long as we find better cuts. The final cut value obtained this way is given in Table 4 in the column labeled 'cut'.

Comparison to published work: There are several papers dealing with the basic relaxation (3). The primal-dual interior-point path-following method is investigated for instance in [14, 17]. In [8, 9] the authors investigate methods that are taylored also for large-scale instances, dealing with instances where n is up to 10000. The approximate solution of (4) has found less attention in the literature. Interior-point methods are applied in [14, 16] to tackle smaller prob-

problem	n	$\|E\|$	cut	initial bd	gap (%)	final bd	gap (%)	m	time
g1d	100	4901	324	396.1	22.25	353.6	9.14	1299	0.44
g2d	200	19701	1050	1268.9	20.85	1170.4	11.47	2427	2.49
g3d	300	44402	1953	2359.6	20.82	2218.3	13.58	2996	6.63
g4d	400	79002	3012	3670.8	21.87	3481.1	15.57	3644	14.20
g5d	500	123503	3987	4937.3	23.84	4728.6	18.60	4431	25.50
g1s	100	495	126	144.6	14.77	130.8	3.81	1816	0.42
g2s	200	1990	318	377.7	18.77	344.2	8.24	2934	2.36
g3s	300	4485	555	678.5	22.25	635.7	14.54	3158	6.25
g4s	400	7980	920	1118.3	21.55	1055.7	14.75	4433	13.20
g5s	500	12475	1252	1557.6	24.41	1487.6	18.82	4321	24.57
spin5	125	375	108	125.3	16.02	111.1	2.87	3714	0.73
spin6	216	648	182	211.8	16.37	186.9	2.69	6223	3.16
spin7	343	1029	304	347.9	14.44	308.4	1.45	11323	10.55
spin8	512	1536	454	524.3	15.48	463.1	2.00	18375	33.44
spin9	729	2187	648	744.7	14.92	666.7	2.89	18265	42.75
spin10	1000	3000	890	1032.6	16.02	921.4	3.53	27830	106.1
G1	800	19176	11612	12083.2	4.06	12005.4	3.39	7372	51.76
G6	800	19176	2172	2656.2	22.29	2566.2	18.15	6983	43.11
G11	800	1600	564	629.2	11.56	572.7	1.54	15946	60.20
G14	800	4694	3054	3191.6	4.51	3140.7	2.84	8973	59.68
G18	800	4694	985	1166.0	18.38	1063.4	7.96	17635	69.19
G22	2000	19990	13321	14135.9	6.12	14045.8	5.44	18325	278.06
G27	2000	19990	3293	4141.7	25.77	4048.4	22.94	15178	406.66
G39	2000	11779	2373	2877.7	21.27	2672.7	12.63	26471	533.36

Table 4: Results for Max-Cut. The initial bound is the solution of (3). The final bound is obtained after 100 function evaluations with the bundle method in case $n \leq 512$, after 50 iterations in case $729 \leq n \leq 1000$ and after 30 iterations for $n = 2000$. m denotes the number of active constraints at the stop of the bundle method, (times in minutes).

lems ($n \approx 100$). In [15], the spectral bundle method is applied with a limited and fixed set of triangle inequalities. Several papers deal with heuristics for Max-Cut. In Burer et al. [7], heuristic solutions for some of the larger Gx instances are reported, and compare extremely well with other heuristic procedures to generate good cuts. Three instances of the G-set from Table 4 were also considered in [7]: G11 (554), G14 (3053), G22 (13331). The best cut values that we found are: G11 (564), G14 (3054), G22 (13321).

4.2 Computational results for Equipartitioning

We now turn to the equipartition problem. Here we use the data from [22]. These are graphs with edge density of 80 % and edge weights drawn randomly from $\{1, \dots, 100\}$ and are available from `http://www.math.uni-klu.ac.at/or/Software`. We have scaled the cost elements by multiplying with 10^{-3}.

The function evaluation of the bundle method amounts now to solving the basic relaxation (5).

problem	n	z_{mc-met}	100 bundle it.	LP
g1d	100	352.375	353.6	786.3
g2d	200	1167.992	1170.4	n.a.
g3d	300	2215.257	2218.3	n.a.
g1s	100	130.008	130.8	137.5
g2s	200	343.156	344.2	410.0
g3s	300	635.040	635.7	854.4
spin5	125	109.344	111.1	110.3
spin6	216	185.662	186.9	187.1

Table 5: Exact solution of (4) for smaller instances, compared to 100 iterations of the bundle method. The final column contains the pure Linear Programming bound with all the triangles included but without the semidefiniteness constraint on X. We indicate with n.a. those problems where we did not manage to solve the LP.

problem	feas sol.	initial bd	gap %	final bd	(gap %)	m	total time
r100	168.712	155.1	8.78	165.3	2.06	2238	0.52
r200	680.633	641.4	6.11	665.3	2.30	6204	3.35
r300	1558.767	1485.9	4.91	1525.3	2.20	10938	9.08
r400	2768.879	2659.6	4.11	2712.9	2.06	16083	19.12
r500	4354.869	4195.7	3.79	4267.7	2.04	22421	34.51
r200	765.728	698.4	9.64	757.5	1.08	15489	4.32
r300	1739.303	1617.9	7.50	1719.1	1.18	32642	11.04
r400	3078.133	2895.9	6.30	3039.9	1.26	54996	22.45
r500	4825.907	4568.7	5.63	4762.9	1.32	83030	40.01

Table 6: Results for equipartition, k=10 and k=50, (times in minutes).

We dualize the sign constraints, and summarize the results in Table 6. Again, the bundle method provides a strong method to deal with many inequalities efficiently. Looking at the number of active constraints (column 7), it is clear that a direct approach to solve any of the relaxations using interior-point methods is hopeless. We also note that including the sign constraints constitutes a significant improvement of the initial basic relaxation. We provide a graphical comparison of the improvement by comparing the initial gap with the final gap in Figure 2. We point out that the improvement using the bundle method as compared to a direct approach based entirely on interior-point methods, see [22], is impressive.

In summary, the results provided here should give enough computational evidence that the bundle method in combination with simple semidefinite programs is a good way to approximate more complicated semidefinite programs.

5 Extensions and Conclusion

We briefly sketch some other combinatorial optimization problems, where our approach might provide an attractive alternative to pure interior-point based methods.

We start out with the **Quadratic Assignment Problem (QAP)**, which can be expressed

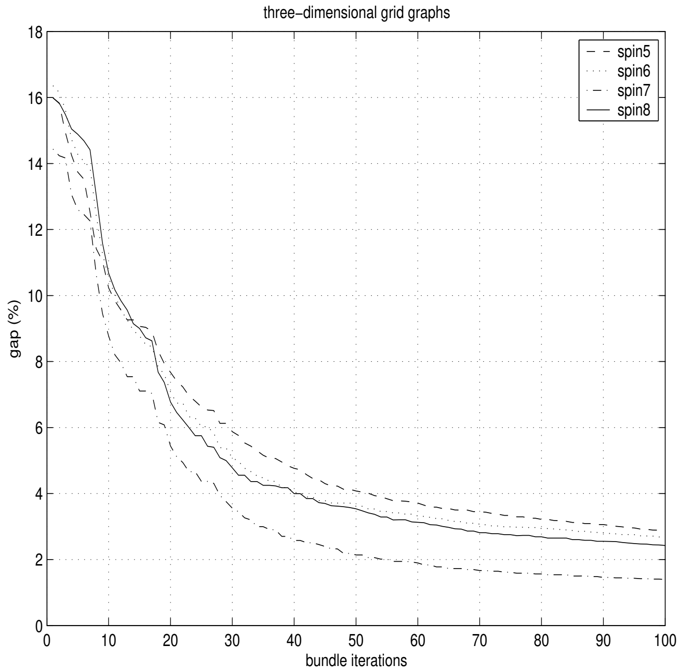

Figure 1: Reduction of gap (in %) for the grid graphs

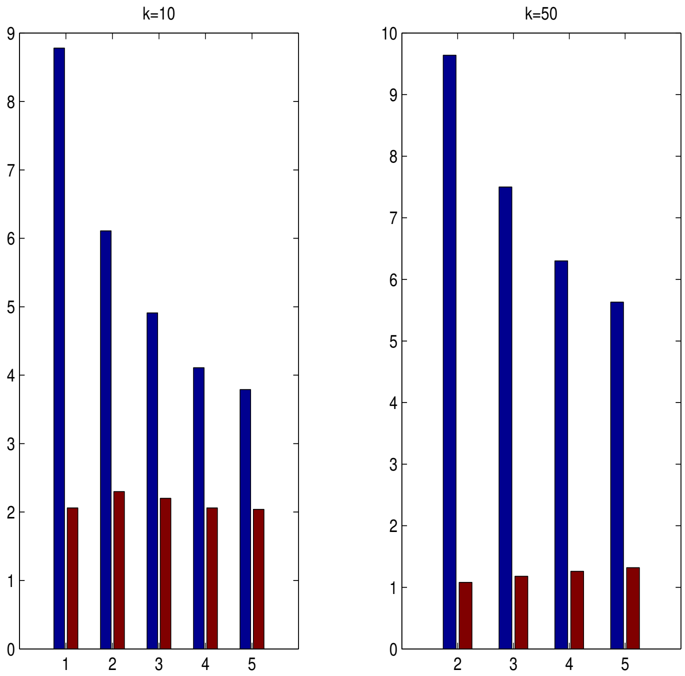

Figure 2: Reduction of gap (in %) for equipartition

as follows:

$$\min \operatorname{tr}(AXB + C)X^T$$

over the set of permutation matrices X. In the recent dissertation [32], an approach similar to the one presented here is applied to obtain bounds for QAP. The resulting bounds are currently by far the best ones available. Since the derivation of the relaxations includes many technicalities that would go beyond the limits of an introductory paper like this one, we refer to [32, 30] for further details and computational results. The technical report [30] is available at `http://www.math.uni-klu.ac.at/or/Forschung/`.

Another interesting source of problems of the type considered in this paper is given by the ϑ-function of a graph. The number $\vartheta(G)$ associated to a graph G separates the clique number $\omega(G)$ from the chromatic number $\chi(G)$,

$$\omega(G) \leq \vartheta(G) \leq \chi(G),$$

see the seminal paper [23]. $\vartheta(G)$ is the optimal value of a semidefinite program of rather simple structure. This semidefinite program can now be strenghtened to give better approximations to $\omega(G)$, see e.g. [24], or to get tigher approximations of the chromatic number, see [25]. These tighter models again are computationally intractable by standard interior-point methods, hence they are another class of problems, where our bundle approach might be a practical alternative.

To summarize, we draw the following conclusions from our computational experiments presented in this paper:

- The bundle method in combination with interior-point methods is a good way to approximate semidefinite programs with a huge number of combinatorial cutting planes.
- The present approach provides the currently strongest bounds at reasonable computational cost for several NP-hard optimization problems like Max-Cut, Equipartitioning and also the Quadratic Assignment Problem.
- The number of function evaluations to reach good approximations is surprisingly small.
- It is also surprising that solving (4) seems less difficult than solving (4) with the semidefiniteness condition omitted.

6 Appendix: rudy-calls to generate the test graphs

```
%% rudy call
% 3d-spinglass random 1,-1 weights
rudy -spinglass3pm 5 5 5 50 5551 > spin5
rudy -spinglass3pm 6 6 6 50 6661 > spin6
rudy -spinglass3pm 7 7 7 50 7771 > spin7
rudy -spinglass3pm 8 8 8 50 8881 > spin8
rudy -spinglass3pm 9 9 9 50 9991 > spin9
rudy -spinglass3pm 10 10 10 50 1010101 > spin10

% dense graphs: random 1,-1 weights
rudy -rnd_graph 100 99 1001 -random 0 1 1001 -times 2 -plus -1 > g1d
rudy -rnd_graph 200 99 2001 -random 0 1 2001 -times 2 -plus -1 > g2d
rudy -rnd_graph 300 99 3001 -random 0 1 3001 -times 2 -plus -1 > g3d
rudy -rnd_graph 400 99 4001 -random 0 1 4001 -times 2 -plus -1 > g4d
rudy -rnd_graph 500 99 5001 -random 0 1 5001 -times 2 -plus -1 > g5d

% sparse (10% density) graphs: random 1,-1 weights
rudy -rnd_graph 100 10 1001 -random 0 1 1001 -times 2  -plus -1 > g1s
rudy -rnd_graph 200 10 2001 -random 0 1 2001 -times 2  -plus -1 > g2s
rudy -rnd_graph 300 10 3001 -random 0 1 3001 -times 2  -plus -1 > g3s
rudy -rnd_graph 400 10 4001 -random 0 1 4001 -times 2  -plus -1 > g4s
rudy -rnd_graph 500 10 5001 -random 0 1 5001 -times 2 -plus  -1 > g5s
```

References

[1] F. ALIZADEH. Interior Point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5:13–51, 1995.

[2] D.S. ATKINSON and P.M. VAIDYA. A cutting plane algorithm for convex programming that uses analytic centers. *Mathematical Programming*, 69:1–43, 1995.

[3] L. BAHIENSE, N. MACULAN, and C. SAGASTIZABAL. The volume algorithm revisited: relation with bundle methods. *Mathematical Programming*, 94:41–69, 2002.

[4] E. BALAS and N. CHRISTOFIDES. A restricted Lagrangian approach to the traveling salesman problem. *Mathematical Programming*, 21:19–46, 1981.

[5] F. BARAHONA and R. ANBIL. The Volume algorithm: producing primal solutions with a subgradient method. *Mathematical Programming*, 87:385–399, 2000.

[6] F. BARAHONA and L. LADANYI. Branch and Cut based on the Volume Algorithm: Steiner trees in graphs and Max-Cut. Technical report, IBM Report RC 22221, 2001.

[7] S. BURER, R.D.C. MONTEIRO, and Y. ZHANG. Rank-two relaxation heuristics for Max-Cut and other binary quadratic programs. *SIAM Journal on Optimization*, 12:503–521, 2002.

[8] S. BURER, R.D.C. MONTEIRO, and Y. ZHANG. A computational study of gradient-based log-barrier algorithms for a class of large-scale sdps. *Mathematical Programming*, 95:359–379, 2003.

[9] C. CHOI and Y. YE. Solving sparse semidefinite programs using the dual scaling algorithm with an iterative solver. Technical report, University of Iowa, 2000.

[10] C. DELORME and S. POLJAK. Laplacian eigenvalues and the maximum cut problem. *Mathematical Programming*, 62:557–574, 1993.

[11] W.E. DONATH and A.J. HOFFMAN. Lower bounds for the partitioning of graphs. *IBM J. of Research and Developement*, 17:420–425, 1973.

[12] A. FRANGIONI. Generalized bundle methods. *SIAM Journal on Optimization*, 13:117–156, 2002.

[13] M.X. GOEMANS and D.P. WILLIAMSON. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

[14] C. HELMBERG. *An interior point method for semidefinite programming and max-cut bounds*. PhD thesis, Graz University of Technology, Austria, 1994.

[15] C. HELMBERG, K.C. KIWIEL, and F. RENDL. Incorporating inequality constraints in the spectral bundle method. In E.A. Boyd R.E. Bixby and R.Z. Rios-Mercado, editors, *Integer Programming and combinatorial optimization*, pages 423–435. Springer Lecture Notes 1412, 1998.

[16] C. HELMBERG and F. RENDL. Solving quadratic (0,1)-problems by semidefinite programming and cutting planes. *Mathematical Programming*, 82:291–315, 1998.

[17] C. HELMBERG, F. RENDL, R. VANDERBEI, and H. WOLKOWICZ. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6:342–361, 1996.

[18] S.E. KARISCH and F. RENDL. Semidefinite Programming and Graph Equipartition. *Fields Institute Communications*, 18:77–95, 1998.

[19] K.C. KIWIEL. *Methods of descent for nondifferentiable optimization*. Springer, Berlin, 1985.

[20] C. LEMARECHAL. Nonsmooth optimization and descent methods. Technical report, International Institute for Applied Systems Analysis, 1978.

[21] C. LEMARECHAL, A. NEMIROVSKII, and Y. NESTEROV. New variants of bundle methods. *Mathematical Programming*, 69:111–147, 1995.

[22] A. LISSER and F. RENDL. Graph partitioning using Linear and Semidefinite Programming. *Mathematical Programming (B)*, 95:91–101, 2002.

[23] L. LOVÁSZ. On the shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25:1–7, 1979.

[24] L. LOVÁSZ and A. SCHRIJVER. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991.

[25] P. MEURDESOIF. Strengthening the lovasz bound for graph coloring. *Mathematical Programming (forthcoming).*

[26] Y. NESTEROV. Complexity estimates of some cutting plane methods based on the analytic barrier. *Mathematical Programming*, 69:149–176, 1995.

[27] Y. NESTEROV. Quality of semidefinite relaxation for nonconvex quadratic optimization. Technical report, CORE Université Catholique de Louvain, 1997.

[28] S. POLJAK and F. RENDL. Nonpolyhedral relaxations of graph bisection problems. *SIAM Journal on Optimization*, 5:467–487, 1995.

[29] B.T. POLYAK. A general method for solving extremum problems. *Soviet Mathematics*, 8:593–597, 1967.

[30] F. RENDL and R. SOTIROV. Bounds for the quadratic assignment problem using the bundle method. Technical report, University of Klagenfurt, Austria, 2003.

[31] H. SCHRAMM and J. ZOWE. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM J. Optimization*, 2:121–152, 1992.

[32] R. SOTIROV. *The Bundle Method in combinatorial optimization*. PhD thesis, University of Klagenfurt, Austria, 2003.

[33] M.J. TODD. A study of search directions in primal-dual interior-point methods for semidefinite programming. *Optimization Methods and Software*, 11:1–46, 1999.