Bounds for the Quadratic Assignment Problem Using the Bundle Method

Franz Rendl* Renata Sotirov†

January 21, 2005

Abstract

Semidefinite Programming (SDP) has recently turned out to be a very powerful tool for approximating some NP-hard problems. The nature of the Quadratic Assignment Problem suggests SDP as a way to derive tractable relaxations. We recall some SDP relaxations of QAP and solve them approximately using the Bundle Method. The computational results demonstrate the efficiency of the approach. Our bounds are the currently among strongest ones available for QAP. We investigate their potential for Branch and Bound settings by looking also at the bounds in the first levels of the branching tree.

Key Words. quadratic assignment problem, semidefinite programming relaxation, bundle method, interior point method.
AMS Subject Classifications. 90C22, 90C27, 90C57, 90C51; Secondary 90C06.

1 Introduction

The Quadratic Assignment Problem (QAP) was introduced in 1957 by Koopmans and Beckmann as a model for location problems, that takes into account the cost of placing a new facility on a certain site as well as the interaction with other facilities. Nowadays, the QAP is widely considered as a classical combinatorial optimization problem. The QAP is also known as a generic model for various real–life problems, see Çela [7] for a list of applications.

Let A, B and C be real $n \times n$ matrices, and Π the set of $n \times n$ permutation matrices. (We assume $n \geq 3$ to avoid trivialities.) The QAP can be stated as follows

$$\text{(QAP)} \quad \mu^* := \min_{X \in \Pi} \text{ tr } (AXB^T + C)X^T. \tag{1}$$

*Universität Klagenfurt, Institute für Mathematik, Universitätsstraße 65-67, A–9020 Klagenfurt, Austria, franz.rendl@uni-klu.ac.at

†Department of Mathematics and Statistics, The University of Melbourne, Parkville, 3052, Vic, Australia, rsotirov@ms.unimelb.edu.au

The formulation (1) is called the *trace formulation* and it was introduced by Edwards in 1977. A QAP is called symmetric, if both matrices A and B are symmetric. Throughout we assume that A and B are symmetric. The QAP is well known to be an NP–hard combinatorial optimization problem and even finding an ϵ–approximation of QAP is an NP–hard problem (Sahni and Gonzales [24]).

Branch and Bound (B&B) algorithms are among the most successful approaches to get optimal solutions for a combinatorial optimization problem. The choice of the bounding method is the most important factor in the performance of B&B methods. The first B&B algorithms for QAP utilize the well known Gilmore–Lawler bound that is cheap to compute but in general not very tight. It is known (see [8]) that the time needed to solve a problem using the B&B algorithm based on the Gilmore–Lawler bound increases with a factor four if the problem dimension is increased by one. Stronger lower bounds for the QAP include bounds based on linear programming relaxations and are used by Adams and Johnson [1], by Resende et al. [23], and by Hahn et al. [14, 15]. Eigenvalue–based bounds are investigated by Finke et al. [9], Hadley et al. [13], and Rendl and Wolkowicz [22].

The recent developments in algorithms as well as in computational platforms have resulted in a large improvement in the capability to solve QAPs exactly. Anstreicher et al. [4] made a break–through by solving a number of previously unsolved large QAPs from QAPLIB [6], including the Nug30, Kra30b and Tho30 problems. They incorporated a quadratic programming bound (QPB) that was introduced by Anstreicher and Brixius in [3], into a branch and bound framework and were running their branch and bound algorithm on a computational grid, see [11]. Their computations are considered to be among the most extensive computations ever performed to solve discrete optimization problems. The computational work to solve a problem of size $n = 30$ (Nug30) took the equivalent of nearly 7 years of computation time on a single HP9000 C3000 workstation, see [4]. A summary of recent advances in the solution of QAP by B&B is given in the survey article by Anstreicher [2]. In [5] S. Burer and D. Vandenbussche propose a novel method to apply dual decomposition to the lift–and–project relaxations of binary integer programs and report very tight bounds for various of problems.

In this paper, we recall semidefinite programming (SDP) relaxations of QAP. Semidefinite programming studies [19, 21, 27] show that it is a very promising method for providing tight relaxations for hard combinatorial problems, notably QAP. In Section 2, we recall and summarize the approach from [27] to derive SDP relaxations for QAP. All relaxations are formulated in the space of symmetric matrices of order $(n-1)^2+1$. The simplest relaxation has n^2+1 equality constraints. Two further refinements of this relaxation are obtained by (first) including $O(n^3)$ additional equations and then $O(n^4)$ sign constraints. Standard interior-point methods are not adequate to solve these latter models.

In Section 3, we propose a variant of the bundle method to solve these relaxations at least approximately with reasonable computational effort. Note that another efficient version of the bundle method for solving large scale semidefinite relaxations, i.e., a spectral bundle method, is proposed in [17, 16]. Using our version of the bundle method, we compute bounds of our relaxations for some of the instances from QAPLIB [6]. The computational results presented in Section 4 demonstrate the efficiency of combining the basic SDP relaxation with the bundle method. The resulting lower bounds are among the currently strongest bounds for QAP, see also [5]. We also show how these bounds behave in the first levels of the Branch and Bound tree. Smaller problems ($n \leq 15$) lead to branching trees with only a few dozen nodes. For larger problems, the reduction of the gap between bound and integer solution going from the root problem to the first level of the branching tree is still significant. This makes the present bounds potential new candidates for use in Branch and Bound methods.

Notation. The space of $k \times k$ real matrices is denoted by $\mathcal{M}_k$, and the space of $k \times k$ symmetric matrices is denoted by $\mathcal{S}_k$. We use $\mathrm{tr}(A)$ to denote the trace of a square matrix A. The space of symmetric matrices is considered with the trace inner product $\langle A, B \rangle = \mathrm{tr}(AB)$. For $A, B \in \mathcal{S}_k$, $A \succeq 0$ (resp. $A \succ 0$) denotes positive semidefiniteness (resp. positive definiteness), and $A \succeq B$ denotes $A - B \succeq 0$. For two matrices $A, B \in \mathcal{M}_k$, $A \geq B$, $(A > B)$ means $a_{ij} \geq b_{ij}$, $(a_{ij} > b_{ij})$ for all i, j.

For $X \in \mathcal{M}_k$, $\mathrm{vec}(X)$ denotes the vector in $\mathbb{R}^{k^2}$ that is formed from the columns of the matrix X. The connection between operators vec and tr is given with the following relation; see e. g. [12],

$$\mathrm{tr}(AB) = (\mathrm{vec}(A^T))^T \mathrm{vec} B, \quad A, B \in \mathcal{M}_k. \tag{2}$$

$\mathrm{Diag}(x)$ is the diagonal matrix with diagonal entries equal to the components of x, and conversely, $\mathrm{diag}(X)$ is the vector of the diagonal elements of the matrix X. $\mathrm{Diag}(x)$ is the adjoint operator of $\mathrm{diag}(X)$.

The Hadamard product of two matrices $U = (u_{ij})$ and $V = (v_{ij})$ of the same size is denoted by $U \circ V$, $(U \circ V)_{ij} = u_{ij} \cdot v_{ij}$ for all i, j. The Kronecker product of matrices A and B is

$$A \otimes B = (a_{ij} b_{kl}) = (a_{ij} B) \quad \forall i, j, k, l,$$

i. e. the matrix formed from all possible products of elements from A and B. The following identity will be used several times, see e. g. [12],

$$\mathrm{vec}(AXB) = (B^T \otimes A)\mathrm{vec}(X). \tag{3}$$

We use e to denote the vector with each component equal to one, and $E = ee^T$ denotes the matrix of ones.

2 SDP Relaxations of QAP

In this section we summarize and simplify the approach from [27] to get SDP relaxations for QAP. The key idea is to reformulate the problem in terms of $x = \mathrm{vec}(X)$ and linearize the quadratic term xx^T in the cost function.

In order to rewrite the cost function from QAP we use (2) and (3) and obtain the following form of the objective function

$$\mathrm{tr}\ (AXB + C)X^T = \langle x, \mathrm{vec}(AXB + C)\rangle = x^T(B \otimes A)x + x^T c,$$

where $x = \mathrm{vec}(X)$ and $c = \mathrm{vec}(C)$. Therefore QAP becomes

$$\min\{x^T(B \otimes A)x + x^T c : \ x = \mathrm{vec}(X), X \in \Pi\}, \tag{4}$$

which is equivalent to

$$\min\{\mathrm{tr}(B \otimes A + \mathrm{Diag}(c))xx^T : \ x = \mathrm{vec}(X), X \in \Pi\},$$

because $c^T x = c^T(x \circ x) = \mathrm{tr}(\mathrm{Diag}(c)(xx^T))$. To derive semidefinite relaxations of QAP we linearize the objective function and obtain the following feasible set of QAP.

$$P := \mathrm{conv}\{xx^T : x = \mathrm{vec}(X), X \in \Pi\}.$$

In order to obtain tractable relaxations for QAP we need to approximate the set P by larger sets containing P. We first impose a semidefiniteness constraint on elements $Y \in P$. The vertices Y of P satisfy the (nonlinear and nonconvex) constraint $Y - \mathrm{diag}(Y)\mathrm{diag}(Y)^T = 0$, which we weaken to $Y - \mathrm{diag}(Y)\mathrm{diag}(Y)^T \succeq 0$. This condition is well known to be equivalent to the convex constraint

$$\begin{pmatrix} 1 & \tilde{y}^T \\ \tilde{y} & Y \end{pmatrix} \succeq 0, \ \tilde{y} = \mathrm{diag}(Y). \tag{5}$$

We next exploit the fact that the row and column sums of permutation matrices are one.

Lemma 1 [13] *Let V be an $n \times (n-1)$ matrix with $V^T e = 0$ and $\mathrm{rank}(V) = n-1$. Then*

$$\left\{X \in \mathcal{M}_n : Xe = X^T e = e\right\} = \left\{\frac{1}{n}ee^T + VMV^T : M \in \mathcal{M}_{n-1}\right\}.$$

■

Matrix V from the previous Lemma could be any basis of $e^\perp$. Our choice for V is

$$V = \begin{pmatrix} I_{n-1} \\ -e_{n-1}^T \end{pmatrix}. \tag{6}$$

The following Lemma gives some more structure of the elements in P.

Lemma 2 *Let $Y \in P$ and*

$$W := \left(\frac{1}{n}e_n \otimes e_n, V \otimes V\right),$$

where V is given in (6). Then there exists a symmetric matrix R of order $(n-1)^2 + 1$, indexed from 0 to $(n-1)^2$, such that

$$R \succeq 0, \ R_{00} = 1 \text{ and } Y = WRW^T.$$

PROOF. (See also [27].) First we look at the extreme points of P. Let Y be one of them, i.e. $Y = xx^T$ for some permutation matrix X. From Lemma 1 it follows that for the permutation matrix X there exists some matrix $M \in \mathcal{M}_{n-1}$ such that $X = \frac{1}{n}ee^T + VMV^T$. With the use of (3), we get

$$x = \mathrm{vec}(X) = \frac{1}{n}(e \otimes e) + (V \otimes V)m = Wz,$$

where $m = \mathrm{vec}(M)$ and $z = \begin{pmatrix} 1 \\ m \end{pmatrix}$. Now

$$Y = xx^T = Wzz^TW^T = WRW^T,$$

with $R = zz^T$. Hence, R is symmetric positive semidefinite and $R_{00} = 1$. The same holds for convex combinations formed from several permutation matrices. ■

Lemma 2 and condition (5) suggest the following set $\hat{P}$ containing P.

$$\begin{aligned}\hat{P} := \{Y \in \mathcal{S}_{n^2} : \exists R \text{ s.t. } \quad & R \succeq 0,\ R_{00} = 1,\ Y = WRW^T, \\ & \tilde{y} = \mathrm{diag}(WRW^T), \begin{pmatrix} 1 & \tilde{y}^T \\ \tilde{y} & WRW^T \end{pmatrix} \succeq 0\}.\end{aligned}$$

In [27] it is shown that $\hat{P}$ has interior points. For instance

$$\hat{R} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{n^2(n-1)}(nI_{n-1} - E_{n-1}) \otimes (nI_{n-1} - E_{n-1}) \end{pmatrix} \succ 0$$

is such that $W\hat{R}W^T$ is the barycenter of P, i. e.

$$W\hat{R}W^T = \frac{1}{n!}\sum_{x \in \Pi} \left(\ xx^T \ \right).$$

We arrive at the *basic SDP relaxation* of QAP

$$(\mathrm{QAP}_{R_1}) \quad \min\{\mathrm{tr}\ (B \otimes A + \mathrm{Diag}(c))Y :\ Y \in \hat{P}\}.$$

We can eliminate the matrix variable Y and formulate QAP_{R_1} with the matrix variable R. For that purpose, we define the following set:

$$\mathcal{R} = \{R \in \mathcal{S}_{(n-1)^2+1} :\ R \succeq 0,\ R_{00} = 1,\ \tilde{y} = \mathrm{diag}(WRW^T), \begin{pmatrix} 1 & \tilde{y}^T \\ \tilde{y} & WRW^T \end{pmatrix} \succeq 0\}.$$

Note that this set is defined by n^2+1 equality constraints of very simple structure in addition to the semidefiniteness constraint. If we define

$$L := W^T(B \otimes A + \mathrm{Diag}(c))W \in \mathcal{M}_{(n-1)^2+1}, \tag{7}$$

then QAP_{R_1} is equivalent to

$$(\mathrm{QAP}_{R_1}) \quad \mu_1^* := \min\{\mathrm{tr}\ LR :\ R \in \mathcal{R}\}.$$

Table 1: Solutions of relaxation QAP_{R_1} for Nugent instances and corresponding computation times

	Nug12	Nug15	Nug20	Nug25	Nug30
μ_1^*	-216	-823	-2073	-4683	-10965
time (seconds)	1.1	4.16	19.4	69.3	198.8

Unfortunately this relaxation is in general very weak. In Table 1 we give solutions of this relaxation for some Nugent instances from QAPLIB [6] computed by a primal-dual path-following interior–point method, and corresponding running times. The computation times are obtained using an Athlon XP 1800. Since all data for these problems are nonnegative, a trivial bound on μ^* is $\mu^* \geq 0$. In view of this, μ_1^* can not be considered a serious approximation of QAP.

QAP_{R_1} exploits the fact that matrices of order n with constant row and column sums have essentially only $(n-1)^2$ degrees of freedom (see Lemma 1), and that $x_{ij} \in \{0,1\}$ gives (5).

To improve the relaxation we need to include further constraints, which are valid for permutation matrices. We next exploit the fact that

$$x_{ij}x_{ik} = x_{ji}x_{ki} = 0 \quad \text{for} \quad j \neq k,$$

holds for the permutation matrix $X = (x_{rs})$.

To express the zero pattern, we index the elements of the matrix $Y \in P$ by $y_{r,s} = Y_{(i,j)(k,l)}$ for $r,s \in \{1,\ldots,n\} \times \{1,\ldots,n\}$, $i,j,k,l \in \{1,\ldots,n\}$. The zero pattern is covered by the following equalities:

$$y_{rs} = 0 \quad \text{for} \quad r = (i,j), s = (i,k), \quad \text{or} \quad r = (j,i), s = (k,i), \;\; j \neq k.$$

We collect all these equalities in the constraint $G(WRW^T) = 0$ which is represented by the set

$$\mathcal{G} := \{R : R \in \mathcal{S}_{(n-1)^2+1}, \;\; G(WRW^T) = 0\}.$$

We strengthen the relaxation QAP_{R_1} by adding this new set of equalities and arrive at the tighter model

$$(\mathrm{QAP}_{R_2}) \quad \mu_2^* := \min\{\mathrm{tr}\; LR : \;\; R \in \mathcal{R} \cap \mathcal{G}\},$$

that contains additional $O(n^3)$ equations, $n^3 - n^2$ to be precise. Model QAP_{R_2} is introduced in [27] as the *Gangster model*. In Table 2 we give results of some numerical experiments. The first column lists some of the larger Nugent instances from QAPLIB [6]. The number in the name of the problem refers to the size of the problem. The second column contains the value of the optimal solution of QAP. In the third column we provide the solutions of the relaxation QAP_{R_2} using an interior-point method. The number of constraints is too big to be manageable by a standard PC. These results were obtained in collaboration

Table 2: Solutions of relaxation QAP_{R_2} obtained by the interior-point method (using NEOS) and by the bundle method with corresponding computation times for one iteration of the algorithms. The interior–point method needs about 20 iterations, the bundle method about 300 iterations.

		interior–point		bundle	
	exact	μ_2^*	time	bound on μ_2^*	time
Nug20	2570	2386	1 h 7'	2380	15.11 "
Nug21	2438	2253	1 h 45'	2244	18.56 "
Nug22	3596	3396	2 h 41'	3372	22.01 "
Nug24	3488	3235	6 h	3217	35.44 "
Nug25	3744	3454	8 h 48'	3438	44.49 "
Nug30	6124	5695	39 h	5651	122.35 "

with Henry Wolkowicz in 2001 by use of the NEOS Server for Optimization. The machine that was used at NEOS was a Sun E6500 server with 24 processors and 24 GB of memory. All processors were 400 MHz Sparc2. The fourth column contains the running times required for one single interior–point iteration of the algorithm. Nug30 was solved with the CSDP solver and the algorithm needed 36 iterations. The solution was obtained after about 1400 hours.

The results show that QAP_{R_2} provides very tight approximations of μ^*, but it becomes also quite clear that the interior–point method is not appropriate for solving this relaxation.

The relaxation QAP_{R_2} can be further tightened by adding nonnegativity constraints

$$(WRW^T)_{rs} \geq 0, \quad \forall r, s = 1, \ldots, n^2. \tag{8}$$

We collect the inequalities (8) which are not yet covered by $G(WRW^T) = 0$ in the constraint $N(WRW^T) \geq 0$. Let us define the set

$$\mathcal{N} := \{R : R \in \mathcal{S}_{(n-1)^2+1},\ N(WRW^T) \geq 0\}.$$

We arrive at the final relaxation, also introduced in [27]:

$$(\mathrm{QAP}_{R_3}) \quad \mu_3^* := \min\{\mathrm{tr}\ LR :\ R \in \mathcal{R} \cap \mathcal{G} \cap \mathcal{N}\}.$$

The resulting SDP has $O(n^4)$ sign constraints and $O(n^3)$ equality constraints. The relaxation QAP_{R_3} can not be solved straightforward by interior–point methods for interesting instances ($n \geq 15$).

Finally, we mention that further refinements of our approximations to μ^* are possible. The fact that P is generated by $0-1$ vectors $x = \mathrm{vec}(X)$ would suggest to include the triangle inequalities

$$0 \leq y_{rs} \leq y_{rr},\ y_{rr} + y_{ss} - y_{rs} \leq 1,$$

$$-y_{tt} - y_{rs} + y_{rt} + y_{st} \le 0, \quad y_{tt} + y_{rr} + y_{ss} - y_{rs} - y_{rt} - y_{st} \le 0,$$

which hold for all distinct triples (r, s, t). This gives an additional $O(n^6)$ constraints. Since we find it already extremely difficult to approximate QAP_{R_3}, we will not pursue this latest relaxation any further, and leave it for future research.

Table 2 shows two things. First the bound QAP_{R_2} yields a drastic improvement compared to QAP_{R_1} and secondly classical interior–point methods are highly inefficient to compute this bound. We now show how we can avoid straight interior–point methods by introducing the bundle method to deal with $G(WRW^T) = 0,\ N(WRW^T) \ge 0$.

3 The Bundle Method to solve the Relaxations

Interior–point methods are very useful and reliable solution methods for semidefinite programs of moderate size, but we have just seen that for QAP_{R_2} and QAP_{R_3} they are not practical. In order to efficiently compute lower bounds of these relaxations, we need a method that is capable to deal with a huge number of constraints. The bundle method turns out to be a convenient method for this purpose. It dates to the 1970's (see e. g. [18, 25, 28]) and 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 vector g contained in the subdifferential of f at $\gamma, g \in \partial f(\gamma)$.

To define f, we dualize the "hard constraints"

$$G(WRW^T) = 0 \quad \text{and} \quad N(WRW^T) \ge 0,$$

and maintain explicitly only the constraints from $\mathcal{R}$. Introducing Lagrange multipliers γ' and $\gamma'' \ge 0$ for the equations and nonnegativity constraints respectively, the Lagrangian is

$$\mathcal{L}(R, \gamma) = \mathrm{tr}\ LR + (\gamma')^T G(WRW^T) - (\gamma'')^T N(WRW^T),$$

where $\gamma = (\gamma', \gamma'')$.

Now we define

$$f(\gamma) := \min_{R \in \mathcal{R}} \mathcal{L}(R, \gamma) = \min_{R \in \mathcal{R}} \langle L + W^T (G^T(\gamma') - N^T(\gamma''))W, R\rangle, \tag{9}$$

and the relaxation QAP_{R_3} is equivalent to

$$\max_{\gamma \in \Gamma} f(\gamma), \tag{10}$$

where $\Gamma := \{(\gamma', \gamma'') : \gamma'' \ge 0\}$. The problem (10) is also difficult to solve directly, but weak duality shows that for any $\gamma \in \Gamma$ we have $f(\gamma) \le \mu_3^* \le \mu^*$, hence any feasible solution γ gives a lower bound on μ^*. (It is our goal to approximate μ_3^* as close as possible.) Note that for some γ the evaluation of $f(\gamma)$ amounts to solving an SDP of the form QAP_{R_1}, which can be done reasonably fast.

We follow now the idea of the bundle method from [10]. For the start of the algorithm we take some initial γ, for instance $\gamma = 0$, and compute R from (9). A pair (γ, R) is called a *matching pair* for f, if $f(\gamma) = \mathcal{L}(R, \gamma)$. Let $\gamma^* = (\gamma'^*, \gamma''^*)$. If (γ^*, R^*) is a matching pair for f then $(g^G(\gamma'^*), g^N(\gamma''^*)) := (G(WR^*W^T), -N(WR^*W^T))$ is a subgradient of $f(\cdot)$ at (γ'^*, γ''^*). We denote a currently best approximation to the maximizer of f with $\hat{\gamma} = (\hat{\gamma}', \hat{\gamma}'')$.

In a general step, we assume to have $\bar{R} = (R_1, \ldots, R_k)$ and $\hat{\gamma} := \gamma_k$, with $(\hat{\gamma}, R_k)$ a matching pair. For each R_i we calculate the corresponding subgradients g_i^G and g_i^N, and form matrices $G^G = (g_1^G, \ldots, g_k^G)$ and $G^N = (g_1^N, \ldots, g_k^N)$. Let $\lambda = (\lambda_1, \ldots, \lambda_k)^T$, $\Lambda = \{\lambda : \lambda \geq 0,\ e^T\lambda = 1\}$, and $F = (\mathrm{tr}(LR_1), \ldots, \mathrm{tr}(LR_k))^T$. The goal is to approximate the function $f(\gamma)$ in the neighborhood of the current iterates reasonably well. The function $f(\gamma)$ is approximated by

$$\begin{aligned}
f_{appr}(\gamma) &= \min_{\lambda \in \Lambda} \langle L + W^T(G^T(\gamma') - N^T(\gamma''))W,\ \sum_{i=1}^k \lambda_i R_i \rangle \\
&= \min_{\lambda \in \Lambda} \sum_{i=1}^k \lambda_i \ \langle L,\ R_i \rangle + \langle \gamma', \sum_{i=1}^k \lambda_i G(WR_iW^T) \rangle \\
&\quad - \langle \gamma'', \sum_{i=1}^k \lambda_i N(WR_iW^T) \rangle \\
&= \min_{\lambda \in \Lambda} F^T\lambda + (\gamma')^T G^G \lambda + (\gamma'')^T G^N \lambda. \qquad (11)
\end{aligned}$$

Since f_{appr} is built of local information from the previous iterates, in order to preserve a reasonable quality of the approximations we should stay in the vicinity of the current point $\hat{\gamma}$. Therefore we use the *proximal point* idea and add a penalty term for the displacement from the current point. We now determine a new candidate $\gamma = (\gamma', \gamma'') \in \Gamma$ from the current iterate $\hat{\gamma} = (\hat{\gamma}', \hat{\gamma}'')$ by solving the concave problem

$$\max_{\gamma \in \Gamma} f_{appr}(\gamma) - \frac{1}{2t}||\gamma - \hat{\gamma}||^2, \qquad (12)$$

where $t > 0$ is a parameter that has to be chosen by the user. Substituting (11) into the maximization problem (12), we obtain the optimization problem

$$\begin{aligned}
&\max_{\gamma \in \Gamma} \min_{\lambda \in \Lambda} F^T\lambda + (\gamma')^T G^G \lambda + (\gamma'')^T G^N \lambda - \frac{1}{2t}||\gamma - \hat{\gamma}||^2 \\
&= \min_{\lambda \in \Lambda,\ \eta \geq 0} \max_{\gamma} F^T\lambda + (\gamma')^T G^G \lambda + (\gamma'')^T G^N \lambda + (\gamma'')^T \eta - \frac{1}{2t}||\gamma - \hat{\gamma}||^2. \qquad (13)
\end{aligned}$$

First–order optimality conditions for the inner maximization in (13) are

$$\frac{\partial}{\partial \gamma'}(\cdot) = 0 \Leftrightarrow G^G\lambda - \frac{1}{t}(\gamma' - \hat{\gamma}') = 0 \Leftrightarrow \gamma' = \hat{\gamma}' + tG^G\lambda, \tag{14}$$

$$\frac{\partial}{\partial \gamma''}(\cdot) = 0 \Leftrightarrow G^N\lambda - \frac{1}{t}(\gamma'' - \hat{\gamma}'') + \eta = 0 \Leftrightarrow \gamma'' = \hat{\gamma}'' + t(\eta + G^N\lambda). \tag{15}$$

We now insert equations for γ' and γ'' obtained in (14) and (15) respectively, into (13) and obtain the optimization problem

$$\min_{\substack{\lambda \in \Lambda \\ \eta \geq 0}} \ \frac{t}{2}||G^G\lambda||^2 + \frac{t}{2}||G^N\lambda + \eta||^2 + \langle F + (\hat{\gamma}')^T G^G + (\hat{\gamma}'')^T G^N, \lambda\rangle + \langle \hat{\gamma}'', \eta\rangle. \tag{16}$$

The minimization problem (16) can be easily solved if one set of the variables is kept constant, see [10, 26]. Keeping η constant results in a convex quadratic problem in λ, which can be easily solved by an interior–point method. Keeping λ constant in the minimization problem (16) results in

$$\min_{\eta \geq 0} \ \frac{t}{2}\langle \eta, \eta\rangle + t\langle \eta, G^N\lambda\rangle + \langle \hat{\gamma}'', \eta\rangle.$$

This problem can be solved coordinatewise. 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 η, λ of (16). Using these estimates λ and η in (14) and (15) we arrive with the next trial point $\gamma_{test} = (\gamma'_{test}, \gamma''_{test})$. To finish one iteration, we need to evaluate the function f at the new point γ_{test}, which amounts to solving an SDP of the form QAP_{R_1}. This is in fact the most time-consuming operation in each iteration of the bundle method. Finally, it should be mentioned that the asymptotic convergence of this approach is rather slow, so we set as an additional stopping condition a maximum number of bundle iterations, which we have set somewhat arbitrarily to 300. The final bound is therefore only a lower approximation to either μ_2^* or μ_3^*. For a more detailed survey of the bundle method see [10, 18, 26].

To see how good the bundle method approximates μ_2^*, we provide some representative results in Table 2. In the fifth column of Table 2 we give the bound of relaxation QAP_{R_2} for different Nugent instances computed with the bundle method. The sixth column shows the running time required for one single iteration of the bundle algorithm (on our PC). We conclude that the bundle method approximates the true value μ_2^* reasonably well, at significantly smaller computational cost. We do not have a similar comparison for QAP_{R_3}, because we do not know how to solve this relaxation exactly for problems of interesting size. But, in the following section we present the computation times required for computing QAP_{R_3} bounds with the bundle method (see Table 6).

4 Computational Results

In this section we present computational results. First, we compare the lower bounds QAP_{R_2} and QAP_{R_3} obtained with the bundle method, with several existing bounding strategies. We use the same test problems as in [3] and [27]. All instances have no linear term, i. e. they are pure quadratic and they are taken from the current version of QAPLIB [6]. We also investigate the lower bounds for some QAPLIB instances in the first and second level of the branching tree. The implementation of our bounds was done in MATLAB and performed on a PC (Athlon XP 1800).

4.1 Comparison With Other Bounds

Tables 3 and 4 collect some instances from QAPLIB [6], their optimum values, lower bounds from the literature, and our bounds. More precisely, the Tables 3 and 4 read as follows. The first column gives the problem instances and their sizes, e. g. Had30 refers to the Hadley instance of the size 30. In the second column we provide the optimum value for each instance. The remaining columns give lower bounds in the following order: GLB is the Gilmore–Lawler bound; KCCEB is the dual LP–based bound from [20]; PB is the projected eigenvalue bound from Hadley, Rendl and Wolkowicz [13], and QPB1 is the quadratic programming bound from Anstreicher and Brixius [3]. The last two columns present the bounds $\mathrm{QAP}_{\mathrm{R}_2}$ and $\mathrm{QAP}_{\mathrm{R}_3}$ that are described in Section 2 and computed by the bundle method. 'n. a.' means that the value of the bound is not available for a particular problem. All bounds are rounded up to the next integer.

Tables 3 and 4 demonstrate the efficiency of the relaxations $\mathrm{QAP}_{\mathrm{R}_2}$ and $\mathrm{QAP}_{\mathrm{R}_3}$. These two relaxations were already proposed in [27]. Here we propose a practical way to approximate them within reasonable computation time. The tables show that the relaxation $\mathrm{QAP}_{\mathrm{R}_3}$ is currently among the strongest bound available for QAP. The last column also gives the relative gap of this bound in %. This gap is often quite small, only a few percentage points. We also point out that we get positive bounds on the Eschermann instances Esc16d and Esc16i, where most of the other bounds are less than 0.

The bounds $\mathrm{QAP}_{\mathrm{R}_2}$ and $\mathrm{QAP}_{\mathrm{R}_3}$ from Table 3 and 4 are obtained after 300 iterations of the bundle algorithm. To give an impression of how the bound improves in the course of the bundle iterations, we present in Table 5 $\mathrm{QAP}_{\mathrm{R}_3}$ bounds for the Nugent type instances obtained after 10, 20, 50, 100, 200 and 300 bundle iterations. The results show that after fast initial progress (first 100 iterations), there is a strong tailing–off effect. Figure 1 gives a graphical representation of the results from Table 5. We have plotted the relative gap in % to the optimal value. Note the similar behavior for all instances: after 50 iterations the gap is below 20 %, after 150 iterations it is below 10 %, and it approaches 5 % after 300 iterations.

Table 6 gives the computation time required for one single iteration of the bundle algorithm for QAP_{R_3}. Note that the time per iteration of the bundle algorithm for solving QAP_{R_3} does not significantly differ from the time required for solving QAP_{R_2} (see Table 2) with the same method. This is due to the dynamic version of the bundle method that is implemented in our computations, see [10, 26].

Table 3: Comparing bounds for QAPLIB instances I

	OPT	GLB	KCCEB	PB	QPB1	QAP_{R_2}	QAP_{R_3}	gap (%)
Esc16a	68	38	41	47	55	49	59	13.24
Esc16b	292	220	274	250	250	275	288	1.37
Esc16c	160	83	91	95	95	111	142	11.25
Esc16d	16	3	4	- 19	-19	-13	8	50.00
Esc16e	28	12	12	6	6	11	23	17.86
Esc16g	26	12	12	9	9	10	20	23.08
Esc16h	996	625	704	708	708	905	970	2.61
Esc16i	14	0	0	-25	-25	-22	9	35.71
Esc16j	8	1	2	-6	-6	-5	7	12.50
Had12	1652	1536	1619	1573	1592	1639	1643	0.54
Had14	2724	2492	2661	2609	2630	2707	2715	0.33
Had16	3720	3358	3553	3560	3595	3675	3699	0.56
Had18	5358	4776	5078	5104	5143	5282	5317	0.77
Had20	6922	6166	6567	6625	6677	6843	6885	0.53
Kra30a	88900	68360	75566	63717	68572	68526	77647	12.66
Kra30b	91420	69065	76235	63818	69021	71429	81156	10.79
Kra32	88700	67390	n.a.	59735	n.a.	75848	79659	10.19
Nug12	578	493	521	472	482	528	557	3.63
Nug14	1014	852	n.a.	871	891	958	992	2.17
Nug15	1150	963	1033	973	996	1069	1122	2.43
Nug16a	1610	1314	1419	1403	1448	1526	1570	2.48
Nug16b	1240	1022	1082	1046	1071	1136	1188	4.19
Nug17	1732	1388	1498	1487	1529	1619	1669	3.64
Nug18	1930	1554	1656	1663	1705	1798	1852	4.04
Nug20	2570	2057	2173	2196	2254	2380	2451	4.63
Nug21	2438	1833	2008	1979	2055	2244	2323	4.72
Nug22	3596	2483	2834	2966	3080	3372	3440	4.34
Nug24	3488	2676	2857	2960	3028	3217	3310	5.10
Nug25	3744	2869	3064	3190	3272	3438	3535	5.58
Nug27	5234	3701	n.a.	4493	n.a.	4887	4965	5.14
Nug28	5166	3786	n.a.	4433	n.a.	4780	4901	5.13
Nug30	6124	4539	4785	5266	5365	5651	5803	5.24

Table 4: Comparing bounds for QAPLIB instances II

	OPT	GLB	KCCEB	PB	QPB1	QAP_{R_2}	QAP_{R_3}	gap(%)
Rou12	235528	202272	223543	200024	206102	219018	223680	5.03
Rou15	354210	298548	323589	296705	303777	220567	333287	5.91
Rou20	725522	599948	641425	597045	607822	641577	663833	8.50
Scr12	31410	27858	29538	4727	8585	23844	29321	6.65
Scr15	51140	44737	48547	10355	12479	41881	48836	4.51
Scr20	110030	86766	94489	16113	23960	82106	94998	13.90
Tai12a	224416	195918	220804	193124	199597	215241	222784	0.73
Tai15a	388214	327501	351938	325019	330310	349179	364761	6.04
Tai17a	491812	412722	441501	408910	416033	440333	451317	8.23
Tai20a	703482	580674	616644	575831	585139	617630	637300	9.41
Tai25a	1167256	962417	1005978	956657	983456	1008248	1041337	10.79
Tai30a	1818146	1504688	1565313	1500407	1518059	1573580	1652186	9.13
Tho30	149936	90578	99855	119254	124684	134368	136059	9.26

Table 5: QAP_{R_3} bounds in dependence of number of iterations of the bundle algorithm

	exact	10 it.	20 it.	50 it.	100 it.	200 it.	300 it.
Nug20	2570	1519	2070	2276	2412	2451	2451
Nug21	2438	1163	1935	2122	2253	2320	2323
Nug22	3596	1590	2757	3107	3370	3434	3440
Nug24	3488	1214	2553	2953	3193	3302	3310
Nug25	3744	1994	2880	3194	3394	3527	3535
Nug27	5234	464	3441	4399	4767	4946	4965
Nug28	5166	197	3664	4115	4580	4869	4901
Nug30	6124	416	3277	4957	5249	5715	5803

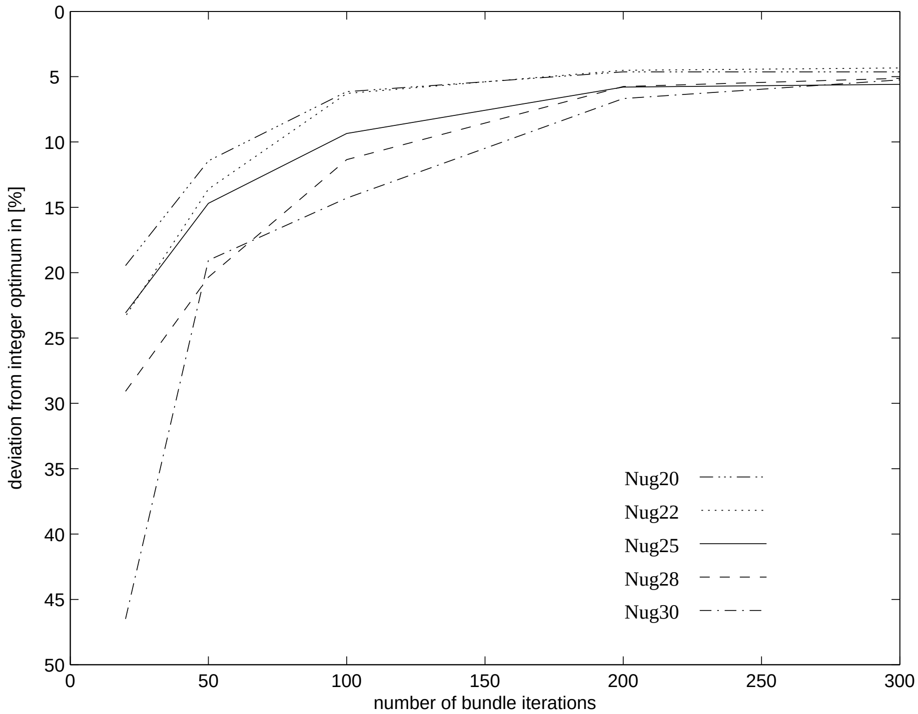

Figure 1: QAP_{R_3} bounds in dependence of number of iterations of the bundle algorithm

Table 6: Computation time per one iteration of the bundle algorithm for the QAP_{R_3} relaxation

n	20	21	22	30
time	22.86”	28.89”	43.38”	223.56”

Table 7: Results for the first level in the branching tree for Had12

	exact	PB	QPB	QAP_3
Had12	1652	1573	1592	1643
Had12.1	1674	1593	1629	1673
Had12.2	1690	1590	1639	1680
Had12.3	1652	1573	1607	1652
Had12.4	1662	1585	1616	1656
Had12.5	1696	1608	1647	1694
Had12.6	1706	1616	1649	1696
Had12.7	1714	1601	1656	1705
Had12.8	1654	1566	1610	1653
Had12.9	1660	1573	1617	1655
Had12.10	1672	1605	1628	1670
Had12.11	1694	1601	1641	1690
Had12.12	1700	1618	1656	1699

4.2 The Bounds After Branching

For the purpose of applying the bound QAP_{R_3} within a branch and bound framework we investigate the effect on the bound after fixing an assignment $x_{ij} = 1$. A considerable growth of the bound by stepping down one level in the branching tree is a desirable feature for a bounding procedure in a Branch and Bound setting. In order to evaluate the growth rate of QAP_{R_3}, we first compare our results for Had12 with results presented in [3]. Table 7 gives lower bounds for Had12 in the first level of the branching tree. The first column lists the root problem and all 12 child problems. With Had12.j we denote jth "child" problem obtained by setting $x_{1j} = 1,\ j = 1, \ldots, 12$. The meaning of the rest of the columns is as follows; the second column presents exact solutions of the "child" problems; PB and QPB are the projected eigenvalue bound and quadratic programming bound respectively (see [3]), and QAP_{R_3} is the bound presented in Section 2. Table 7 shows that the performance of QPB is far superior to that of PB, and that the performance of QAP_{R_3} is far superior to that of QPB. Note that the value of QPB is sufficient to fathom Had12.7 and Had12.12, but the value of QAP_{R_3} is sufficient to fathom all "child" problems, proving optimality at the first level of the branching tree.

Further branching experiments are done on the Nugent set of problems.

Since the Nugxx instances possess inherent symmetries due to their distance matrices, only four subproblems are to be considered in the first level of Nug12

Table 8: Results for the first level in the branching tree for Nug12

	exact	QAP_2	QAP_3
Nug12	578	529	557
Nug12.1	586	551	578
Nug12.2	586	551	577
Nug12.5	578	552	575
Nug12.6	600	556	584

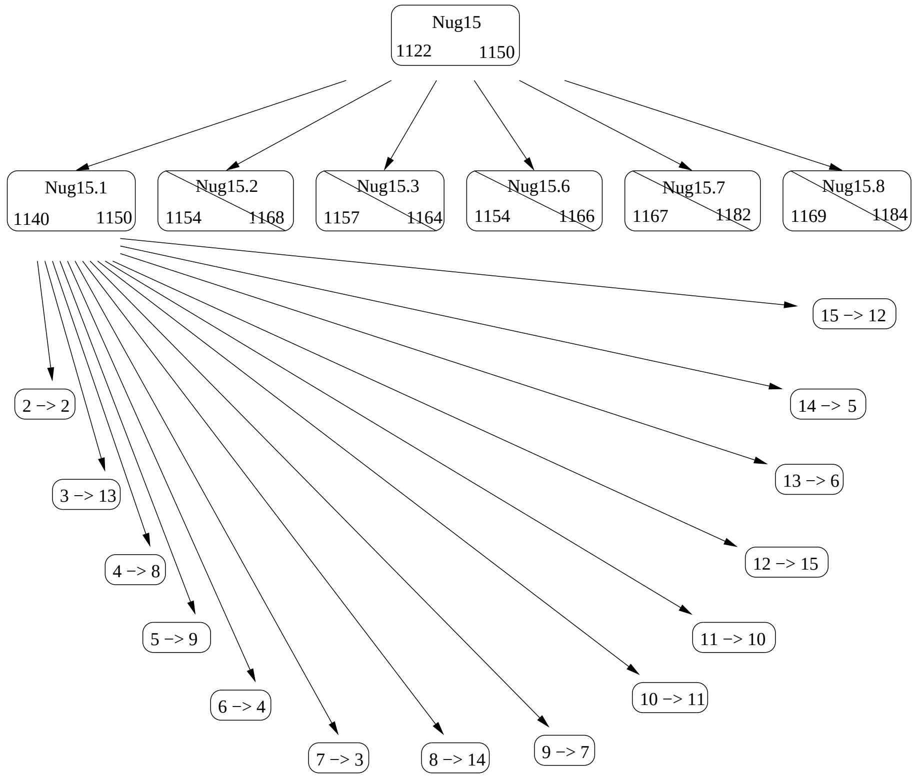

Figure 2: First and second level in the branching tree for Nug15

Table 9: Results for the first level in the branching tree for Nug20

	exact	QAP_2	QAP_3
Nug20	2570	2380	2451
Nug20.1	2612	2449	2518
Nug20.2	2570	2420	2488
Nug20.3	2586	2421	2487
Nug20.6	2592	2427	2501
Nug20.7	2584	2420	2491
Nug20.8	2604	2419	2502

Table 10: Results for the first level in the branching tree for Nug30

	QAP_2	QAP_3
Nug30	5568	5803
Nug30.1	5809	5939
Nug30.2	5771	5895
Nug30.3	5756	5881
Nug30.7	5767	5900
Nug30.8	5750	5885
Nug30.9	5756	5891
Nug30.13	5756	5896
Nug30.14	5750	5883
Nug30.15	5768	5889

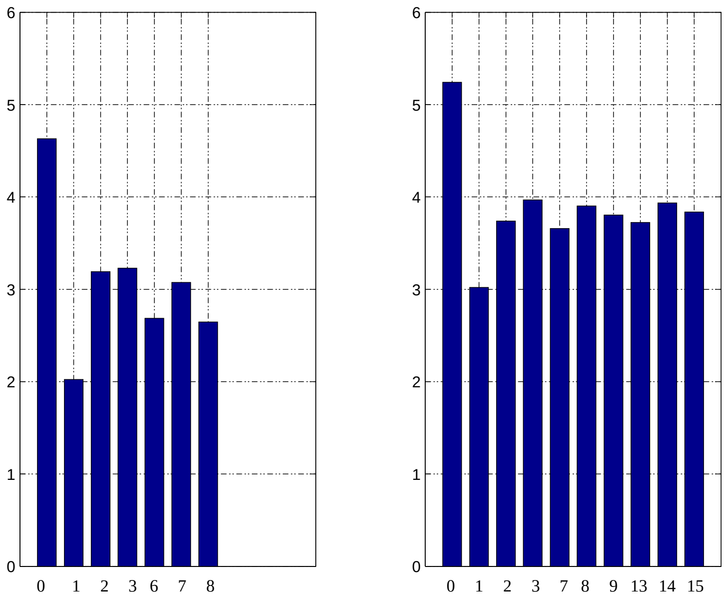

Figure 3: Gap reduction at first level of branching tree for Nug20 and Nug30. The bar labeled 0 corresponds to the root problem, the other bars give the relative gap at the first level of branching.

problem and six subproblems in the first level of Nug15 problem. With Nugxx.j we denote jth "child" problem obtained by setting $x_{j1} = 1$. Table 8 gives results for the first level in the branching tree of Nug12. The first column contains again the problem instances. The remaining columns give exact solution, QAP_2, and QAP_3 bounds, respectively.

Figure 2 shows that in the first level of the branching tree for Nug15, all "child" problems except Nug15.1 are fathomed. Our computations of all "child" problems of Nug15.1 (196 since there is no symmetry) resulted with only 14 not fathomed problems (see Figure 2). Hence, we have proved the optimal solution of Nug15 problem in the second level of the branching tree.

Tables 9 and 10 present the bounds in the first level of the branching tree for Nug20 and Nug30.

It is instructive to look at the relative gap of these bounds at the root and the first level of branching. In Figure 3 we plot the results for Nug20 and Nug30 and show the deviation in % from the integer optimum. For Nug20, the first level of branching reduces the initial gap of 4.6% to 3% or lower. Turning to Nug30, we see that the initial gap of 5.2 % goes down to below 4% after branching. We consider this a very promising feature of the relaxation for use in a Branch and

Bound framework.

5 Concluding remarks

We have shown that a basic semidefinite relaxation of QAP, combined with the bundle method, yields very good approximations to the relaxations QAP_{R2} and QAP_{R3} which are currently the among strongest bounds available for QAP.

Further improvement is possible to speed up the bundle iterations. We have not exploited the fact that in the course of the iterations, there are only very small changes in the dual variables, hence the primal cost function changes only slightly. Using sensitivity theory, it should be possible to warm-start the function evaluation, rather than solving the basic SDP from scratch in each iteration, as we do now.

Finally, the bundle method provides estimates of the dual variables corresponding to the sign constraints. This information may be useful to guide the branching process.

References

[1] W. P. Adams and T. A. Johnson. Improved Linear Programming–Based Lower Bounds for the Quadratic Assignment Problem. in Proceedings of the DIMACS Workshop on Quadratic Assignment Problems, *DIMACS Series in Discrete Mathematics and Theoretical Computes Sciences*, American Mathematical Society, 16:43-75, 1994.

[2] K. Anstreicher. Recent advances in the solution of quadratic assignment problems. *Mathematical Programming B*, 97:27–42, 2003.

[3] K. Anstreicher and N. Brixius. A New Bound for the Quadratic Assignment Problem Based on Convex Quadratic Programming. *Mathematical Programming*, 89:341–357, 2001.

[4] K. Anstreicher, N. Brixius, J.–P. Goux and J. Linderoth. Solving Large Quadratic Assignment Problems on Computational Grids. *Mathematical Programming B*, 91:563–588, 2002.

[5] S. Burer and D. Vandenbussche. Solving Lift-and-Project Relaxations of Binary Integer Programs. Technical report, June 2004.

[6] R. E. Burkard, S. Karisch and F. Rendl. QAPLIB – A Quadratic Assignment Problem Library. *European Journal of Operational Research*, 55:115–119, 1991.

[7] F. Çela. *The Quadratic Assignment Problem: Theory and Algorithms.* Kluwer, Massachessets, USA, 1998.

[8] J. Clausen and M. Perregaard. Solving Large Quadratic Assignment Problems in Parallel. *Computational Optimization and Applications*, 8:111–127, 1997.

[9] G. Finke, R. E. Burkard and F. Rendl. Quadratic Assignment Problems. *Annals of Discrete Mathematics*, 31:61–82, 1987.

[10] I. Fischer, G. Gruber, F. Rendl and R. Sotirov. The Bundle Method in Combinatorial Optimization. University of Klagenfurt, Austria, working paper, 2003.

[11] I. Foster and C. Kesselman. Computational Grids. In I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Francisco, CA, 1999.

[12] A. Graham. Kronecker Products and Matrix Calculus with Applications, Mathematics and its Applications. *Ellis Horwood Limited, Chichester*, 1981.

[13] S. W. Hadley F. Rendl and H. Wolkowicz. A New Lower Bound via Projection for the Quadratic Assignment Problem. *Mathematics of Operations Research*, 17:727–739, 1992.

[14] P. M. Hahn, T. Grant and N. Hall. A Branch–and–Bound Algorithm for the Quadratic Assignment Problem Based on the Hungarian Method. *European Journal of Operational Research*, 108:629–640, 1998.

[15] P. M. Hahn, W. L. Hightower, T. A. Johnson, M. Guignard–Spielberg and C. Roucairol. Tree Elaboration Strategies in Branch and Bound Algorithms for Solving the Quadratic Assignment Problem. *Yugoslav Journal of Operations Research*, 11:41–60, 2001.

[16] C. Helmberg. A Cutting Plane Algorithm for Large Scale Semidefinite Relaxations. *Padberg Festschrift: The Sharpest Cut*, edited by M. Grötschel, MPS-SIAM, 233–256, 2004.

[17] C. Helmberg and F. Rendl. A Spectral Bundle Method for Semidefinite Programming. *SIAM Journal on Optimization*, 10(3): 673–69, 2000.

[18] J. B. Hiriart–Urruty and C. Lemaréchal. Convex Analysis and Minimization Algorithms II. *Springer Verlag*, 1991.

[19] S. E. Karisch. Nonlinear Approaches for Quadratic Assignment and Graph Partition Problems. Dissertation, Technical University of Graz, Austria 1995.

[20] S. E. Karisch, E. Çela, J. Clausen, and T. Espersen. A Dual Framework for Lower Bounds of the Quadratic Assignment Problem Based on Linearization. *Computing* 63:351–403, 1999.

[21] P. Pardalos and H. Wolkowicz, editors. *Quadratic assignment and related problems*. American Mathematical Society, Providence, RI, 1994. Papers from the workshop held at Rutgers University, New Brunswick, New Jersey, May 20–21, 1993.

[22] F. Rendl and H. Wolkowicz. Applications of Parametric Programming and Eigenvalue Maximization to the Quadratic Assignment Problem. *Mathematical Programming*, 53:63–78, 1992.

[23] M. G. C. Resende, K. G. Ramakrishnan and Z. Drezner. Computing Lower Bounds for the Quadratic Assignment Problem with an Interior Point Algorithm for Linear Programming. *Operations Research*, 43(5):63–78, 1992.

[24] S. Sahni and T. Gonzales. P–Complete Approximation Problems. *Journal of ACM*, 23:555–565, 1976.

[25] H. Schramm and J. Zowe. A Version of the Bundle Idea for Minimizing a Nonsmooth Function: Conceptional Idea, Convergence Analysis, Numerical Results. *SIAM Journal on Optimization*, 2:121–152, 1992.

[26] R. Sotirov. Bundle Methods in Combinatorial Optimization. Dissertation, University of Klagenfurt, Austria, 2003.

[27] Q. Zhao, S. E. Karisch, F. Rendl and H. Wolkowicz. Semidefinite Programming Relaxations for the Quadratic Assignment Problem. *Journal of Combinatorial Optimization*, 2:71-109, 1998.

[28] J. Zowe. Nondifferentiable Optimization – a Motivation and a Short Introduction Into the Subgradient – and the Bundle Concept. NATO ASI Series, vol.f15 *Computational Mathematical Programming* edited by K. Schittkowski, Springer–Verlag, Berlin Heidelberg 1985.