The Quadratic Assignment Problem *

Rainer E. Burkard † Eranda Çela † Panos M. Pardalos‡
Leonidas S. Pitsoulis‡

Abstract

This paper aims at describing the state of the art on quadratic assignment problems (QAPs). It discusses the most important developments in all aspects of the QAP such as linearizations, QAP polyhedra, algorithms to solve the problem to optimality, heuristics, polynomially solvable special cases, and asymptotic behavior. Moreover, it also considers problems related to the QAP, e.g. the biquadratic assignment problem, and discusses the relationship between the QAP and other well known combinatorial optimization problems, e.g. the traveling salesman problem, the graph partitioning problem, etc.
The paper will appear in the Handbook of Combinatorial Optimization to be published by Kluwer Academic Publishers, P. Pardalos and D.-Z. Du, eds.

Keywords: quadratic assignment problem, algorithms, asymptotic behavior, polynomially solvable special cases.

AMS-classification: 90C27, 90B80, 68Q25

Contents

*This research has been supported by the Spezialforschungsbereich F 003 "Optimierung und Kontrolle", Projektbereich Diskrete Optimierung.

†Technische Universität Graz, Institut für Mathematik B, Steyrergasse 30, A-8010 Graz, Austria.

‡Center for Applied Optimization, Industrial and Systems Engineering Department, University of Florida, Gainesville, FL 32611

1 Introduction

The quadratic assignment problem (QAP) was introduced by Koopmans and Beckmann in 1957 as a mathematical model for the location of a set of indivisible economical activities [113]. Consider the problem of allocating a set of facilities to a set of locations, with the cost being a function of the distance and flow between the facilities, plus costs associated with a facility being placed at a certain location. The objective is to assign each facility to a location such that the total cost is minimized. Specifically, we are given three $n \times n$ input matrices with real elements $F = (f_{ij})$, $D = (d_{kl})$ and $B = (b_{ik})$, where f_{ij} is the flow between the facility i and facility j, d_{kl} is the distance between the location k and location l, and b_{ik} is the cost of placing facility i at location k. The Koopmans-Beckmann version of the QAP can be formulated as follows: Let n be the number of facilities and locations and denote by N the set $N = \{1, 2, \ldots, n\}$.

$$\min_{\phi \in \mathcal{S}_n} \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{\phi(i)\phi(j)} + \sum_{i=1}^{n} b_{i\phi(i)} \tag{1}$$

where $\mathcal{S}_n$ is the set of all permutations $\phi : N \to N$. Each individual product $f_{ij} d_{\phi(i)\phi(j)}$ is the cost of assigning facility i to location $\phi(i)$ and facility j to location $\phi(j)$. In the context of facility location the matrices F and D are symmetric with zeros in the diagonal, and all the matrices are nonnegative. An instance of a QAP with input matrices F, D and B will be denoted by $QAP(F, D, B)$, while we will denote an instance by $QAP(F, D)$, if there is no linear term (i.e., $B = 0$).
A more general version of the QAP was introduced by Lawler [118]. In this version we are given a four-dimensional array $C = (c_{ijkl})$ of coefficients instead of the two matrices F and D and the problem can be stated as

$$\min_{\phi \in \mathcal{S}_n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij\phi(i)\phi(j)} + \sum_{i=1}^{n} b_{i\phi(i)} \tag{2}$$

Clearly, a Koopmans-Beckmann problem $QAP(F, D, B)$ can be formulated as a Lawler QAP by setting $c_{ijkl} := f_{ij} d_{kl}$ for all i, j, k, l with $i \neq j$ or $k \neq l$ and $c_{iikk} := f_{ii} d_{kk} + b_{ik}$, otherwise.

Although extensive research has been done for more than three decades, the QAP, in contrast with its linear counterpart the linear assignment problem (LAP), remains one of the hardest optimization problems and no exact algorithm can solve problems of size $n > 20$ in reasonable computational time. In fact, Sahni and Gonzalez [164] have shown that the QAP is NP-hard and that even finding an approximate solution within some constant factor from the optimal solution cannot be done in polynomial time unless P=NP. These results hold even for the Koopmans-Beckmann QAP with coefficient matrices fulfilling the triangle inequality (see Queyranne [152]). So far only for a very special case of the Koopmans-Beckmann QAP, the *dense linear arrangement problem* a polynomial time approximation scheme has been found , due to Arora, Frieze, and Kaplan [7]. Complexity aspects of the QAP will be discussed in more detail in Section 3.

Let us conclude this section with a brief review of some of the many applications of the QAP. In addition to facility layout problems, the QAP appears in applications such as backboard wiring, computer manufacturing, scheduling, process communications, turbine balancing, and many others.

One of the earlier applications goes back to Steinberg [168] and concerns *backboard wiring*. Different devices such as controls and displays have to be placed on a panel, where they have to be connected to each other by wires. The problem is to find a positioning of the devices so as to minimize the total wire length. Let n be the number of devices to be placed and let d_{kl} denote the wire length from position k to position l. The flow matrix $F = (f_{ij})$ is given by

$$f_{ij} = \begin{cases} 1 & \text{if device } i \text{ is connected to device } j, \\ 0 & \text{otherwise.} \end{cases}$$

Then the solution to the corresponding QAP will minimize the total wire length. Another application in the context of location theory is a *campus planning* problem due to Dickey and Hopkins [58]. The problem consists of planning the sites of n buildings in a campus, where d_{kl} is the distance from site k to site l, and f_{ij} is the traffic intensity between building i and building j The objective is to minimize the total walking distance between the buildings.

In the field of ergonomics Burkard and Offermann [36] showed that QAPs can be applied to *typewriter keyboard design*. The problem is to arrange the keys in a keyboard such as to minimize the time needed to write some text. Let the set of integers $N = \{1, 2, \ldots, n\}$ denote the set of symbols to be arranged. Then f_{ij} denotes the frequency of the appearance of the pair of symbols i and j. The entries of the distance matrix $D = d_{kl}$ are the times needed to press the key in position l after pressing the key in position k for all the keys to be assigned. Then a permutation $\phi \in \mathcal{S}_n$ describes an assignment of symbols to keys An optimal solution ϕ^* for the QAP minimizes the average time for writing a text. A similar application related to ergonomic design, is the development of control boards in order to minimize eye fatigue by McCormick [126]. There are also numerous other applications of the QAP in different fields e.g. hospital lay-out (Elshafei [63]), ranking of archeological data (Krarup and Pruzan [114]), ranking of a team in a relay race (Heffley [93]), scheduling parallel production lines (Geoffrion and Graves [76]), and analyzing chemical reactions for organic compounds (Ugi, Bauer, Friedrich, Gasteiger, Jochum, and Schubert [173]).

2 Formulations

For many combinatorial optimization problems there exist different, but equivalent mathematical formulations, which stress different structural characteristics of the problem, which may lead to different solution approaches. Let us start with the observation that every permutation ϕ of the set $N = \{1, 2, \ldots, n\}$ can be represented by an $n \times n$ matrix $X = (x_{ij})$, such that

$$x_{ij} = \begin{cases} 1 & \text{if } \phi(i) = j, \\ 0 & \text{otherwise.} \end{cases}$$

Matrix X is called a *permutation matrix* and is characterized by following *assignment constraints*

$$\begin{aligned} \sum_{i=1}^{n} x_{ij} &= 1, \qquad j = 1, 2, \ldots, n, \\ \sum_{j=1}^{n} x_{ij} &= 1, \qquad i = 1, 2, \ldots, n, \\ x_{ij} &\in \{0, 1\}, \qquad i, j = 1, 2, \ldots, n. \end{aligned}$$

We denote the set of all permutation matrices by $\mathbf{X}_n$. Due to a famous theorem of Birkhoff the permutation matrices correspond in a unique way to the vertices of the assignment polytope (the Birkhoff polytope, the perfect matching polytope of $K_{n,n}$ etc.). This leads to the following description of a QAP as quadratic integer program.

2.1 Quadratic Integer Program Formulation

Using permutation matrices instead of permutations, the QAP ((2) can be formulated as the following integer program with quadratic objective function (hence the name Quadratic Assignment Problem by Koopmans and Beckmann [113]).

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} c_{ijkl}x_{ik}x_{jl} + \sum_{i,j=1}^{n} b_{ij}x_{ij} \tag{3}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n, \tag{4}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n, \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad i, j = 1, 2, \ldots, n. \tag{6}$$

¿From now on, whenever we write $(x_{ij}) \in \mathbf{X}_n$, it will be implied that the x_{ij} satisfy the assignment constraints (4), (5) and (6).

Many authors have proposed methods for linearizing the quadratic form of the objective function (3) by introducing additional variables; some of these of linearizations will be discussed in Section 4.

A QAP in Koopmans-Beckmann form can be formulated in a more compact way if we define an *inner product* between matrices. Let the inner product of two real $n \times n$ matrices A, B be defined by

$$\langle A, B\rangle := \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}b_{ij}.$$

Given some $n \times n$ matrix A, a permutation $\phi \in \mathcal{S}_n$ and the associated permutation matrix $X \in \mathbf{X}_n$, then AX^T and XA permute the columns and rows of A, respectively, according to the permutation ϕ and therefore

$$XAX^T = (a_{\phi(i)\phi(j)}).$$

Thus we can formulate a Koopmans-Beckmann QAP alternatively as

$$\begin{aligned} \min \quad & \langle F, XDX^T\rangle + \langle B, X\rangle \\ \text{s.t.} \quad & X \in \mathbf{X}_n. \end{aligned} \tag{7}$$

2.2 Concave Quadratic Formulation

In the objective function of (3), let the coefficients c_{ijkl} be the entries of an $n^2 \times n^2$ matrix S, such that c_{ijkl} is on row $(i-1)n + k$ and column $(j-1)n + l$. Now let $Q := S - \alpha I$, where I is the $(n^2 \times n^2)$ unit matrix and α is greater than the row norm $\|S\|_\infty$ of matrix S. The subtraction of a constant from the entries on the main diagonal of S does not change the optimal solutions of the corresponding QAP, it simply adds a constant to the

objective function. Hence we can consider a QAP with coefficient array Q instead of S. Let $x = (x_{11}, x_{12}, \ldots, x_{1n}, x_{21}, \ldots, x_{nn})^t = (x_1, \ldots, x_{nn})^t$. Then we can rewrite the objective function of the QAP with array of coefficients Q as a quadratic form $x^T Q x$, where:

$$\begin{aligned} x^T Q x &= \sum_{i=1}^{n^2} q_{ii} x_i^2 + 2 \sum_{i=1}^{n^2-1} \sum_{j=i+1}^{n^2} q_{ij} x_i x_j \\ &= \sum_{i=1}^{n^2} (q_{ii} + \sum_{\substack{j=1 \\ j \neq i}}^{n^2} q_{ij}) x_i^2 - \sum_{i=1}^{n^2-1} \sum_{j=i+1}^{n^2} q_{ij} (x_i - x_j)^2 \\ &= \sum_{i=1}^{n^2} (-\alpha + \sum_{j=1}^{n^2} s_{ij}) x_i^2 - \sum_{i=1}^{n^2-1} \sum_{j=i+1}^{n^2} s_{ij} (x_i - x_j)^2 \\ &\leq \sum_{i=1}^{n^2} (-\alpha + \sum_{j=1}^{n^2} s_{ij}) x_i^2. \end{aligned}$$

Since $x^T[1/2(Q + Q^T)]x = 1/2 x^T Q x$, we can assume that Q is symmetric and negative definite. Therefore we have a quadratic concave minimization problem and can formulate the QAP as

$$\begin{aligned} \min \quad & x^T Q x \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n, \\ & \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n, \\ & x_{ij} \geq 0, \quad i, j = 1, 2, \ldots, n. \end{aligned} \tag{8}$$

Bazaraa and Sherali [16] introduced the above formulation, and used it to derive cutting plane procedures. Although their exact methods were computationally not efficient, heuristics derived from these procedures produced suboptimal solutions of good quality.

By adding the term αI to the matrix Q instead of subtracting it, we could always assume that the objective function of the QAP is convex. This leads to the formulation of the QAP as a quadratic convex minimization problem.

2.3 Trace Formulation

The trace of an $n \times n$ matrix B is defined to be the sum of its diagonal elements, i.e.:

$$trB := \sum_{i=1}^{n} b_{ii}.$$

Consider a Koopmans-Beckmann QAP instance with input matrices F, D and B. Letting $\bar{D} = X D^T X^T$, then

$$tr(F\bar{D}) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} \bar{d}_{ji} = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{\phi(i)\phi(j)},$$

since $\bar{d}_{ji} = d_{\phi(i)\phi(j)},\ i,j = 1,\ldots,n$, where $\phi \in \mathcal{S}_n$ is the permutation associated with X (see 2.1). Since $tr(BX^T) = \sum_{i=1}^{n} b_{i\phi(i)}$, the QAP in (7) can be formulated as

$$\begin{array}{ll} \min & tr(FXD^T + B)X^T \\ \text{s.t.} & X \in \mathbf{X}_n. \end{array} \tag{9}$$

The trace formulation of the QAP first appeared in Edwards [61, 62], and was used by Finke, Burkard, and Rendl [67] to introduce the eigenvalue lower bounding techniques for symmetric QAPs (see Section 7.1). Given any two real $n \times n$ matrices A, B, recall the well known properties $tr(AB) = tr(BA)$, $(AB)^T = B^T A^T$ and $trA = trA^T$. For $F = F^T$ we can then write the quadratic term in (9) as

$$trFXD^TX^T = trFXDX^T,$$

where D is not necessarily symmetric. Therefore, given a QAP instance where only one of the matrices is symmetric (say F), we can transform it into a QAP instance where both matrices are symmetric. This is done by introducing a new symmetric matrix $E = \frac{1}{2}(D + D^T)$:

$$trFXE^TX^T = \frac{1}{2}tr(FXD^TX^T + FXDX^T) = trFXD^TX^T.$$

2.4 Kronecker Product

Let A be a real $m \times n$ matrix and let B be a real $p \times q$ matrix. Then the *Kronecker product* of matrices A and B is defined as

$$A \otimes B := \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix}.$$

That is, $A \otimes B$ is the $mp \times nq$ matrix formed from all possible pairwise element products of A and B. If we let $vec(X) \in \mathbb{R}^{n^2}$ be the vector formed by the columns of a permutation matrix X, the QAP can be formulated as

$$\begin{array}{ll} \min & vec(X)^T(F \otimes D)vec(X) + vec(B)^T vec(X), \\ \text{s.t.} & X \in \mathbf{X}_n. \end{array} \tag{10}$$

Operations using the Kronecker product and its properties have been studied in detail by Graham [84]. However, the above formulation is rarely used in investigations of the QAP. Based on that formulation Lawler [118] gave an alternative formulation of the QAP as a linear assignment problem (LAP) of size n with the additional constraint that only $(n^2 \times n^2)$ permutation matrices which are Kronecker products of $n \times n$ permutation matrices are feasible. If as before the $(n^2 \times n^2)$ cost matrix C contains the n^4 costs c_{ijkl}, such that the $(ijkl)$-th element corresponds to the element in the $((i-1)n+k)$-th row and $((j-1)n+l)$-th column of C, the QAP can be written as

$$\begin{array}{ll} \min & \langle C, Y \rangle \\ \text{s.t.} & Y = X \otimes X, \\ & X \in \mathbf{X}_n. \end{array} \tag{11}$$

Because of the additional constraint to be fulfilled by the feasible solutions the resulting LAP cannot be solved efficiently.

3 Computational complexity

The results described in this section bring evidence to the fact that the QAP is a "very hard" problem from the theoretical point of view. Not only that the QAP cannot be solved efficiently but it even cannot be approximated efficiently within some constant approximation ratio. Furthermore, finding local optima is not a trivial task even for simply structured neighborhoods like the 2-opt neighborhood.

Two early results obtained by Sahni and Gonzalez [164] in 1976 settled the complexity of solving and approximating the QAP. It was shown that the QAP is NP-hard and that even finding an ϵ-approximate solution for the QAP is a hard problem, in the sense that the existence of a polynomial ϵ-approximation algorithm implies $P = NP$. In the following, let $Z(F, D, \phi)$ denote the objective function value of a solution ϕ for a QAP with flow matrix F and distance matrix D.

Definition 3.1 *Given a real number $\epsilon > 0$, an algorithm Υ for the QAP is said to be an* ϵ-approximation algorithm *if*

$$\left| \frac{Z(F, D, \pi_\Upsilon) - Z(F, D, \pi_{opt})}{Z(F, D, \pi_{opt})} \right| \leq \epsilon \,, \tag{12}$$

holds for every instance $QAP(F, D)$, where π_Υ is the solution of $QAP(F, D)$ computed by algorithm Υ and π_{opt} is an optimal solution of $QAP(F, D)$. The solution of $QAP(F, D)$ produced by an ϵ-approximation algorithm is called an ϵ-approximate solution.

Theorem 3.2 (Sahni and Gonzalez [164], 1976)
The quadratic assignment problem is strongly NP-hard.
For an arbitrary $\epsilon > 0$, the existence of a polynomial time ϵ-approximation algorithm for the QAP implies $\mathcal{P} = \mathcal{NP}$.

The proof is done by a reduction from the *Hamiltonian cycle problem*: Given a graph G, does G contain a cycle which visits each vertex exactly once (see [73])?

Queyranne [152] derives an even stronger result which further confirms the widely spread belief on the inherent difficulty of the QAP in comparison with other difficult combinatorial optimization problems. It it well known and very easy to see that the *traveling salesman problem (TSP)* is a special case of the QAP. The TSP on n cities can be formulated as a $QAP(F, D)$ where F is the distance matrix of the TSP instance and D is the adjacence matrix of a Hamiltonian cycle on n vertices. In the case that the distance matrix is symmetric and satisfies the triangle inequality, the TSP is approximable in polynomial time within 3/2 as shown by Christofides [46]. Queyranne [152] showed that, unless $P = NP$, $QAP(A, B)$ is not approximable in polynomial time within some finite approximation ratio, even if A is the distance matrix of some set of points on a line and B is a symmetric block diagonal matrix.

A more recent result of Arora, Frieze and Kaplan [7] answers partially one of the open questions stated by Queyranne in [152]. What happens if matrix A is the distance matrix of n points which are regularly spaced on a line, i.e., points with abscissae given by $x_p = p$, $p = 1, \ldots, n$? This special case of the QAP is termed *linear arrangement problem* and is a well studied NP-hard problem. In the linear arrangement problem the matrix B is not restricted to have the block diagonal structure mentioned above, but is simply a symmetric 0-1 matrix. Arora et al. give a *polynomial time approximation scheme (PTAS)* for the

linear arrangement problem in the case that the 0-1 matrix B is *dense*, i.e., the number of 1 entries in B is in $\Omega(n^2)$, where n is the size of the problem. They show that for each $\epsilon > 0$ there exists an ϵ-approximation algorithm for the dense linear arrangement problem with time complexity depending polynomially on n and exponentially on $1/\epsilon$, hence polynomial for each fixed $\epsilon > 0$.

Recently it has been shown that even finding a locally optimal solution of the QAP can be prohibitively hard, i.e., even local search is hard in the case of the QAP. Below we formalize this idea to some extent.

Assume that an optimization problem P is given by specifying a ground set $\mathcal{E}$, a set $\mathcal{F} \subseteq 2^{\mathcal{E}}$ of feasible solutions and a cost function $c\colon \mathcal{E} \to \mathbb{R}$. This cost function c implies an objective function $f\colon \mathcal{F} \to \mathbb{R}$ defined by $f(S) = \sum_{x \in S} c(x)$, for all $S \in \mathcal{F}$. The goal is to find a feasible solution which minimizes the objective function. For every feasible solution $S \in \mathcal{F}$ let a *neighborhood* $\mathcal{N}(S) \subset \mathcal{F}$ of S be given. This neighborhood consists of feasible solutions which are somehow "close" to S. Now, instead of looking for a globally optimal solution $S^* \in \mathcal{F}$ of the problem P, that is

$$f(S^*) = \min_{S \in \mathcal{F}} f(S),$$

we look for a *locally optimal solution* or a *local minimum* of P, that is an $\bar{S} \in \mathcal{F}$ such that

$$f(\bar{S}) = \min_{S \in \mathcal{N}(\bar{S})} f(S).$$

An algorithm which produces a locally optimal solution, is frequently called a *local search* algorithm. Some local search algorithms for the QAP are described in Section 8.

Let us consider the intriguing question "Is it easy to find a locally optimal solution for the QAP?". Clearly the answer depends on the involved neighborhood structure. If the neighborhoods $\mathcal{N}(S)$ are replaced by new neighborhoods $\mathcal{N}'(S)$, one would generally expect changes in the local optimality status of a solution. The theoretical basis for facing this kind of problems was introduced by Johnson, Papadimitriou and Yannakakis in [97]. They define the so-called *polynomial-time local search problems*, shortly *PLS problems*. A pair $(P,\mathcal{N})$, where P is a (combinatorial) optimization problem P and $\mathcal{N}$ is an associated neighborhood structure, defines a *local search problem* which consists of finding a locally optimal solution of P with respect to the neighborhood structure $\mathcal{N}$. Without going into technical details a PLS problem is a local search problem for which local optimality can be checked in polynomial time. In analogy with decision problems, there exist complete problems in the class of PLS problems. The PLS-complete problems, are – in the usual complexity sense – the most difficult among the PLS problems.

Murthy, Pardalos and Li [138] introduce a neighborhood structure for the QAP which is similar to the neighborhood structure proposed by Kernighan and Lin [109] for the graph partitioning problem. For this reason we will call it a *K-L type neighborhood structure for the QAP.* Murthy et al. show that the corresponding local search problem is PLS-complete.

A K-L type neighborhood structure for the QAP. Consider a permutation $\phi_0 \in \mathcal{S}_n$. A *swap* of ϕ_0 is a permutation $\phi \in \mathcal{S}_n$ obtained from ϕ_0 by applying a transposition (i,j) to it, $\phi = \phi_0 \circ (i,j)$. A transposition (i,j) is defined as a permutation which maps i to j, j to i, and k to k for all $k \notin \{i,j\}$. In the facility location context a swap is obtained by interchanging the facilities assigned to two locations i and j. A *greedy swap* of permutation

ϕ_0 is a swap ϕ_1 which minimizes the difference $Z(F, D, \phi) - Z(F, D, \phi_0)$ over all swaps ϕ of ϕ_0. Let $\phi_0, \phi_1, \ldots, \phi_l$ be a set of permutations in $\mathcal{S}_n$, each of them being a greedy swap of the preceding one. Such a sequence is called *monotone* if for each pair of permutations ϕ_k, ϕ_t in the sequence, $\{i_k, j_k\} \cap \{i_t, j_t\} = \emptyset$, where ϕ_k (π_t) is obtained by applying transposition (i_k, j_k) ((i_t, j_t)) to the preceding permutation in the sequence. The *neighborhood* of ϕ_0 consists of all permutations which occur in the (unique) maximal monotone sequence of greedy swaps starting with permutation ϕ_0. Let us denote this neighborhood structure for the QAP by $\mathcal{N}_{\text{K-L}}$. It is not difficult to see that, given a $QAP(F, D)$ of size n and a permutation $\phi \in \mathcal{S}_n$, the cardinality of $\mathcal{N}_{\text{K-L}}(\pi)$ does not exceed $\lfloor n/2 \rfloor + 1$.

It is easily seen that the local search problem $(\text{QAP}, \mathcal{N}_{\text{K-L}})$ is a PLS problem. Pardalos, Rendl, and Wolkowicz [147] have shown that a PLS-complete problem, namely the *graph partitioning problem* with the neighborhood structure defined by Kernighan and Lin [109] is PLS-reducible to $(\text{QAP}, \mathcal{N}_{\text{K}-\text{L}})$. This implies the following result.

Theorem 3.3 (Pardalos, Rendl and Wolkowicz [147], 1994)
The local search problem $(\text{QAP}, \mathcal{N}_{K\text{-}L})$*, where* $\mathcal{N}_{K\text{-}L}$ *is the Kernighan-Lin type neighborhood structure for the QAP, is PLS-complete.*

The PLS-completeness of $(\text{QAP}, \mathcal{N}_{\text{K-L}})$ implies that, in the worst case, a general local search algorithm as described above involving the Kernighan-Lin type neighborhood finds a local minimum only after a time which is exponential on the problem size. Numerical results, however, show that such local search algorithms perform quite well when applied to QAP test instances, as reported in [138].

Another simple and frequently used neighborhood structure in $\mathcal{S}_n$ is the so-called *pair-exchange* (or *2-opt*) neighborhood $\mathcal{N}_2$. The pair-exchange neighborhood of a permutation $\phi_0 \in \mathcal{S}_n$ consists of all permutations $\phi \in \mathcal{S}_n$ obtained from ϕ_0 by applying some transposition (i, j) to it. Thus, $\mathcal{N}_2(\phi) = \{\phi \circ (i, j) \colon 1 \leq i, j \leq n,\ i \neq j, \}$.
It can also be shown that $(QAP, \mathcal{N}_2)$ is PLS-complete. Schäffer and Yannakakis [165] have proven that the graph partitioning problem with a neighborhood structure analogous to $\mathcal{N}_2$ is PLS-complete. A similar PLS-reduction as in [147] implies that the local search problem $(\text{QAP}, \mathcal{N}_2)$, where $\mathcal{N}_2$ is the pair-exchange neighborhood, is PLS-complete. This implies that the time complexity of a general local search algorithm for the QAP involving the pair-exchange neighborhood is also exponential in the worst case.

Finally, let us mention that no local criteria are known for deciding how good a locally optimal solution is as compared to a global one. ¿From the complexity point of view, deciding whether a given local optimum is a globally optimal solution to a given instance of the QAP, is a hard problem, see Papadimitriou and Wolfe [145].

4 Linearizations

The first attempts to solve the QAP eliminated the quadratic term in the objective function of (2), in order to transform the problem into a (mixed) 0-1 linear program. The linearization of the objective function is usually achieved by introducing new variables and new linear (and binary) constraints. Then existing methods for (mixed) linear integer programming (MILP) can be applied. The very large number of new variables and constraints, however, usually poses an obstacle for efficiently solving the resulting linear integer programs.
MILP formulations provide moreover LP relaxations of the problem which can be used to compute lower bounds. In this context the "tightness" of the continuous relaxation of the resulting linear integer program is a desirable property.

In this section we present four linearizations of the QAP: Lawler's linearization [118], which was the first, Kaufmann and Broeckx's linearization [108], which has the smallest number of variables and constraints, Frieze and Yadegar's linearization [70] and the linearization of Adams and Johnson [3]. The last linearization which is a slight but relevant modification of the linearization proposed by Frieze and Yadegar [70], unifies most of the previous linearizations and is important for getting lower bounds.

4.1 Lawler's Linearization

Lawler [118] replaces the quadratic terms $x_{ij}x_{kl}$ in the objective function of (2) by n^4 variables

$$y_{ijkl} := x_{ij}x_{kl}, \quad i,j,k,l = 1,2,\ldots,n,$$

and obtains in this way a 0-1 linear program with n^4+n^2 binary variables and n^4+2n^2+1 constraints. Thus the QAP can be written as the following 0-1 linear program (see [118, 23])

$$\begin{aligned}
\min \quad & \sum_{i,j=1}^{n}\sum_{k,l=1}^{n} c_{ijkl}y_{ijkl} \\
\text{s.t.} \quad & (x_{ij}) \in \mathbf{X}_n, \\
& \sum_{i,j=1}^{n}\sum_{k,l=1}^{n} y_{ijkl} = n^2, \\
& x_{ij} + x_{kl} - 2y_{ijkl} \geq 0, \quad i,j,k,l = 1,2,\ldots,n, \\
& y_{ijkl} \in \{0,1\}, \quad i,j,k,l = 1,2,\ldots,n.
\end{aligned} \tag{13}$$

4.2 Kaufmann and Broeckx Linearization

By adding a large enough constant to the cost coefficients, which does not change the optimal solution, we may assume that all cost coefficients c_{ijkl} are nonnegative. By rearranging terms in the objective function (2) we obtain

$$\sum_{i,j=1}^{n} x_{ij} \sum_{k,l=1}^{n} c_{ijkl}x_{kl}. \tag{14}$$

Kaufmann and Broeckx [108] define n^2 new real variables

$$w_{ij} := x_{ij} \sum_{k,l=1}^{n} c_{ijkl}x_{kl}, \quad i,j = 1,\ldots,n, \tag{15}$$

and plug them in the objective function of (14) to obtain a linear objective function of the form

$$\sum_{i,j=1}^{n} w_{ij}.$$

Then they introduce n^2 constants $a_{ij} := \sum_{k,l=1}^{n} c_{ijkl}$ for $i,j = 1,\ldots,n$, and show that the QAP (2) is equivalent to the following mixed 0-1 linear program

$$\begin{aligned} \min \quad & \sum_{i,j=1}^{n} w_{ij} \\ \text{s.t.} \quad & (x_{ij}) \in \mathbf{X}_n, \\ & a_{ij}x_{ij} + \sum_{k,l=1}^{n} c_{ijkl}x_{kl} - w_{ij} \le a_{ij}, \quad i,j = 1,\ldots,n, \\ & w_{ij} \ge 0, \quad i,j = 1,2,\ldots,n. \end{aligned} \tag{16}$$

This formulation employs n^2 real variables, n^2 binary variables and $n^2 + 2n$ constraints. The proof of equivalence of the QAP to the mixed integer linear program (16) can be found in [23, 108]. The above linearization, as well as others that appeared in the literature (see e.g. [24, 29]), are obtained by applying the general linearization strategy proposed by Glover [78].

4.3 Frieze and Yadegar Linearization

Frieze and Yadegar [70] replace the products $x_{ij}x_{kl}$ of the binary variables by continuous variables y_{ijkl} ($y_{ijkl} := x_{ij}x_{kl}$) and get the following mixed integer linear programming formulation for the QAP (2)

$$\min \quad \sum_{i,j=1}^{n} \sum_{k,l=1}^{n} c_{ijkl}y_{ijkl} \tag{17}$$

$$\text{s.t.} \quad (x_{ij}) \in \mathbf{X}_n, \tag{18}$$

$$\sum_{i=1}^{n} y_{ijkl} = x_{kl}, \quad j,k,l = 1,\ldots,n, \tag{19}$$

$$\sum_{j=1}^{n} y_{ijkl} = x_{kl}, \quad i,k,l = 1,2,\ldots,n, \tag{20}$$

$$\sum_{k=1}^{n} y_{ijkl} = x_{ij}, \quad i,j,l = 1,\ldots,n, \tag{21}$$

$$\sum_{l=1}^{n} y_{ijkl} = x_{ij}, \quad i,j,k = 1,2,\ldots,n, \tag{22}$$

$$y_{ijij} = x_{ij}, \quad i,j = 1,2,\ldots,n, \tag{23}$$

$$0 \le y_{ijkl} \le 1, \quad i,j,k,l = 1,2,\ldots,n. \tag{24}$$

This mixed integer program has n^4 real variables, n^2 binary variables and $n^4+4n^3+n^2+2n$ constraints. For obtaining a lower bound Frieze and Yadegar considered a Lagrangean relaxation of this mixed integer program by relaxing the constraints (21) and (24) and solved it approximately by applying subgradient optimization techniques. They showed that the solution of the Lagrangean relaxation is larger than all lower bounds derived from reduction techniques applied to the Gilmore-Lawler bound for the QAP (see Section 7.1). ¿From a result of Geoffrion [75] follows that the solution of the Lagrangean relaxation equals the solution of the continuous relaxation of the mixed integer program (17)-(24).

It is interesting to notice here that the gap between the optimal value of this continuous relaxation and the optimal value of the QAP can be enormous. Dyer, Frieze, and McDiarmid [60] showed for QAPs whose coefficients c_{ijkl} are independent random variables uniformly distributed on $[0, 1]$ that the expected optimal value of the above mentioned linearization has a size of $O(n)$. On the other hand the expected optimal value of such QAPs increases with high probability as $\Omega(n^2)$, as shown by Burkard and Fincke [32]. Consequences of this asymptotic behavior will be discussed in some detail in Section 12. No similar asymptotic result is known for the continuous relaxation of the linearization due to Adams and Johnson [3] which is presented in the following section.

4.4 Adams and Johnson Linearization

Adams and Johnson presented in [3] a new 0-1 linear integer programming formulation for the QAP, which resembles to a certain extent the linearization of Frieze and Yadegar. It is based on the linearization technique for general 0-1 polynomial programs introduced by Adams and Sherali in [4, 5]. The QAP with array of coefficients $C = (c_{ijkl})$ is proved to be equivalent to the following mixed 0-1 linear program

$$\begin{aligned} \min \quad & \sum_{i,j=1}^{n} \sum_{k,l=1}^{n} c_{ijkl} y_{ijkl} \\ \text{s.t.} \quad & (x_{ij}) \in \mathbf{X}_n, \end{aligned} \tag{25}$$

$$\begin{aligned} & \sum_{i=1}^{n} y_{ijkl} = x_{kl}, \quad j,k,l = 1,\ldots,n, \\ & \sum_{j=1}^{n} y_{ijkl} = x_{kl}, \quad i,k,l = 1,2,\ldots,n, \\ & y_{ijkl} = y_{klij}, \quad i,j,k,l = 1,\ldots,n, \\ & y_{ijkl} \geq 0, \quad i,j,k,l = 1,2,\ldots,n, \end{aligned} \tag{26}$$

where each y_{ijkl} represents the product $x_{ij}x_{kl}$. The above formulation contains n^2 binary variables x_{ij}, n^4 continuous variables y_{ijkl}, and $n^4 + 2n^3 + 2n$ constraints excluding the nonnegativity constraints on the continuous variables. Although as noted by Adams and Johnson [3] a significant smaller formulation in terms of both the variables and constraints could be obtained, the structure of the continuous relaxation of the above formulation is favorable for solving it approximately by means of the Lagrangean dual. (See Section 6.2 for more information.)

The theoretical strength of the linearization (25) lies in the fact that the constraints of the continuous relaxations of previous linearizations can be expressed as linear combinations of the constraints of the continuous relaxation of (25), see [3, 98]. Moreover, many of the previously published lower-bounding techniques can be explained based on the Lagrangean dual of this relaxation. For more details on this topic we refer to Section 6.2.

As noted by the Adams et al. [3], the constraint set of (25) describes a solution matrix Y which is the Kronecker product of two permutation matrices (i.e., $Y = X \otimes X$ where $X \in \mathcal{S}_n$), and hence this formulation of the QAP is equivalent to (11).

5 QAP Polytopes

A polyhedral description of the QAP and of some of his relatives have been recently investigated by Barvinok [12], Jünger and Kaibel [100, 101], Kaibel [102], and Padberg and Rijal [142, 161]. Although in an early stage yet, the existing polyhedral theory around the QAP counts already a number of results concerning basic features like dimensions, affine hulls, and valid and facet defining inequalities for the general QAP polytope and the symmetric QAP polytope.

The linearization of Frieze and Yadegar introduced in the previous section can be used as a starting point for the definition of the QAP polytope. The QAP polytope is defined as a convex hull of all 0-1 vectors (x_{ij}, y_{ijkl}), $1 \leq i, j, k, l \leq n$, which are feasible solutions of the MILP formulation of Frieze and Yadegar [70].

Another possibility to introduce the QAP polytope is the formulation of the QAP as a graph problem as proposed by Jünger and Kaibel [100]. This formulation provides some additional insight in the problem and allows an easier use of some technical tools e.g. projections and affine transformations. The latter lead to a better understanding of the relationship between the general QAP polytope and related polytopes, e.g. the symmetric QAP polytope, or well studied polytopes of other combinatorial optimization problems like the traveling salesman polytope or the cut polytope (see [102]).

For each $n \in \mathbb{N}$ consider a graph $G_n = (V_n, E_n)$ with vertex set $V_n = \{(i, j) \colon 1 \leq i, j \leq n\}$ and edge set $E_n = \{((i, j), (k, l)) \colon i \neq k, j \neq l\}$. Clearly, the maximal cliques in G_n have cardinality n and correspond to the permutation matrices. Given an instance of the Lawler QAP with coefficients c_{ijkl} and linear term coefficients b_{ij}, we introduce b_{ij} as vertex weights and c_{ijkl} as weight of the edge $((i, j), (k, l))$. Solving the above QAP instance is equivalent to finding a maximal clique with minimum total vertex- and edge-weight. For each clique C in G_n with n vertices we denote its incidence vector by (x^C, y^C), where $x^C \in \mathbb{R}^{n^2}$, $y^C \in \mathbb{R}^{\frac{n^2(n-1)^2}{2}}$

$$x_{ij} = \begin{cases} 1 & \text{if } (i,j) \in C, \\ 0 & \text{otherwise} \end{cases} \qquad y_{ijkl} = \begin{cases} 1 & \text{if } (i,j),(k,l) \in C, \\ 0 & \text{otherwise} \end{cases}$$

The QAP polytope denoted by QAP_n is then given by

$$QAP_n := conv\{(x^C, y^C) \colon C \text{ is a clique with } n \text{ vertices in } G_n\}.$$

It turns out that *the traveling salesman polytope* and *the linear ordering polytope* are projections of QAP_n, and that QAP_n is a face of the *Boolean quadric polytope*, see [102].

Barvinok [12], Padberg and Rijal [142], and Jünger and Kaibel [100] have independently computed the dimension of QAP_n, and have shown that the inequalities $y_{ijkl} \geq 0$, $i \neq k$, $j \neq l$, are facet defining. (These are usually called trivial facets of QAP_n.) Moreover, Padberg and Rijal [142], and Jünger and Kaibel [100] have independently shown that the

affine hull of QAP_n is described by the following equations which are linearly independent:

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad 1 \le j \le n-1 \tag{27}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad 1 \le i \le n, \tag{28}$$

$$-x_{kl} + \sum_{i=1}^{k-1} y_{ijkl} + \sum_{i=k+1}^{n} y_{klij} = 0 \qquad \begin{array}{l} 1 \le j \ne l \le n, 1 \le k \le n-1, \\ \text{or} \quad 1 \le l < j \le n, k = n \end{array} \tag{29}$$

$$-x_{ij} + \sum_{l=1}^{j-1} y_{ijkl} + \sum_{l=j+1}^{n} y_{ijkl} = 0 \qquad \begin{array}{l} 1 \le j \le n, 1 \le i \le n-3, \\ i < k \le n-1 \quad \text{or} \\ 1 \le j \le n-1, i = n-2, \\ k = n-1 \end{array} \tag{30}$$

$$-x_{kj} + \sum_{l1}^{j1} y_{ilkj} + \sum_{l=j+1}^{n} y_{ilkj} = 0 \qquad \begin{array}{l} 1 \le j \le n-1, 1 \le i \le n-3, \\ i < k \le n-1 \end{array} \tag{31}$$

Summarizing we get the following theorem:

Theorem 5.1 (Barvinok [12], 1992, Jünger and Kaibel [100], 1996, Padberg and Rijal [142], 1996)

(i) The affine hull of the QAP polytope QAP_n is given by the linear equations (27)-(31). These equations are linearly independent and the rank of the system is $2n(n-1)^2 - (n-1)(n-2)$, for $n \ge 3$.

(ii) The dimension of QAP_n is equal to $1 + (n-1)^2 + n(n-1)(n-2)(n-3)/2$, for $n \ge 3$.

(iii) The inequalities $y_{ijkl} \ge 0$, $i < k$, $j \ne l$, define facets of QAP_n.

Padberg and Rijal [142] identified additionally two classes of valid inequalities for QAP_n, the *clique* inequalities and the *cut* inequalities, where the terminology is related to the graph G_n. The authors identify some conditions under which the cut inequalities are not facet defining. It is an open problem, however, to identify facet defining inequalities within these classes. A larger class of valid inequalities, the so-called *box inequalities* have been described by Kaibel [102]. Those inequalities are obtained by exploiting the relationship between the Boolean quadric polytope and the QAP polytope. A nice feature of the box inequalities is that it can be decided efficiently whether they are facet defining or not, and in the latter case some facet defining inequality which dominates the corresponding box inequality can be derived.

Similar results have been obtained for the symmetric QAP polytope, $SQAP_n$, arising in the case that at least one of the coefficient matrices of the given QAP (matrices F, D in (1)) is symmetric. The definition of $SQAP_n$ is given by means of a hypergraph $H_n = (V_n, F_n)$, where V_n is the same set of vertices as in graph G_n and F_n is the set of hyperedges $\{(i,j),(k,l),(i,l),(k,j)\}$ for all $i \ne k$, $j \ne l$. A set $C \subset V_n$ is called a clique in

H_n if it is a clique in G_n. Again, the incidence vector (x^C, y^C) of a clique C is introduced by

$$x_{ij} = \begin{cases} 1 & \text{if } (i,j) \in C \\ 0 & \text{otherwise} \end{cases} \qquad y_{ijkl} = \begin{cases} 1 & \text{if } i < k,\ l \neq j,\ (i,j),(k,l) \in C \\ 0 & \text{otherwise} \end{cases}$$

Here, $x^C \in \mathbb{R}^{n^2}$ and $y^C \in \mathbb{R}^{\frac{n^2(n-1)^2}{4}}$. The polytope $SQAP_n$ is then defined as

$$SQAP_n := conv\{(x^C, y^C) \colon C \text{ is a clique with } n \text{ vertices in } G_n\}$$

Padberg and Rijal [142] and Jünger and Kaibel [101] showed that the following system of equations (32)-(35) offers a minimal linear description of the affine hull of $SQAP_n$.

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad 1 \leq i \leq n \tag{32}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad 1 \leq j \leq n-1 \tag{33}$$

$$-x_{ij} - x_{kj} + \sum_{l=1}^{j-1} y_{ilkj} + \sum_{l=j+1}^{n} y_{ijkl} = 0 \qquad \begin{array}{l} 1 \leq i < k \leq n \\ 1 \leq j \leq n, \end{array} \tag{34}$$

$$-x_{kj} - x_{kl} + \sum_{i=1}^{k-1} y_{ijkl} + \sum_{i=k+1}^{n} y_{kjil} = 0 \qquad \begin{array}{l} 1 \leq k \leq n \\ 1 \leq j \leq n-3, \\ 1 \leq j < l \leq n-1 \end{array} \tag{35}$$

Jünger and Kaibel [101] proved a conjecture of Padberg and Rijal concerning the dimension of $SQAP_n$. They also introduced a class of facet defining inequalities, so-called *curtain inequalities*. The separation problem for these inequalities has been shown to be NP-hard. By summarizing these results we get the following theorem

Theorem 5.2 (Jünger and Kaibel [101], 1996, Padberg and Rijal [142], 1996)

(i) The affine hull of the symmetric QAP polytope $SQAP_n$ is described by the linear equations (32)-(35). These equations are linearly independent and the rank of the system is $n^2(n-2) + 2n - 1$.

(ii) The dimension of $SQAP_n$ is equal to $(n-1)^2 + n^2(n-3)^2/4$.

(iii) The inequalities $y_{ijkl} \geq 0$ for $i < k$, $j < l$, and $x_{ij} \geq 0$ for $1 \leq i, j \leq n$, define facets of QAP_n.

(iv) For each $i < k$ and for all $J \subseteq \{1, 2, \ldots, n\}$ the row curtain inequalities

$$-\sum_{j \in J} x_{ij} + \sum_{\substack{j,l \in J \\ j<l}} y_{ijkl} \leq 0$$

are valid for $SQAP_n$. For each $j < l$ and for all $I \subseteq \{1, 2, \ldots, n\}$ the column curtain inequalities

$$-\sum_{i \in I} x_{ij} + \sum_{\substack{i,k \in I \\ i<k}} y_{ijkl} \leq 0$$

are valid for $SQAP_n$.
All curtain inequalities with $3 \leq |I|, |J| \leq n-3$ define facets of $SQAP_n$. The other curtain inequalities define faces which are contained in trivial facets of $SQAP_n$.

Finally, there are some additional results concerning the affine description and the facial structure of polytopes of special versions of sparse QAPs, e.g. sparse Koopmans-Beckmann QAPs, see Kaibel [102]. The idea is to take advantage of the sparsity for a better analysis and description of the related polytopes. These investigations, however, are still in their infancy.

6 Lower Bounds

Lower bounding techniques are used within implicit enumeration algorithms, such as branch and bound, to perform a *limited* search of the feasible region of a minimization problem, until an optimal solution is found. A more limited use of lower bounding techniques concerns the evaluation of the performance of heuristic algorithms by providing a relative measure of proximity of the suboptimal solution to the optimum. In comparing lower bounding techniques, the following criteria should be taken into consideration:

- Complexity of computing the lower bound.
- *Tightness* of the lower bound (i.e., "small" gap between the bound and the optimum solution).
- Efficiency in computing lower bounds for subsets of the original feasible set.

Since there is no clear ranking of the performance of the lower bounds that will be discussed below, all of the above criteria should be kept in mind while reading the following paragraphs. Considering the asymptotic behavior of the QAP (see Section 12) it should be fair to assume that the tightness of the lower bound probably dominates all of the above criteria. In other words, if there is a large number of feasible solutions close to the optimum, then a lower bound which is not tight enough, will fail to eliminate a large number of subproblems in the branching process.

6.1 Gilmore-Lawler Type Lower Bounds

Based on the formulation of the general QAP as an LAP of dimension n^2 stated in formulation (11), Gilmore [77] and Lawler [118] derived lower bounds for the QAP, by constructing a solution matrix Y in the process of solving a series of LAPs. If the resulting matrix Y is a permutation matrix, then the objective function value yielded by Y is optimal, otherwise it is bounded from below by $\langle C, Y \rangle$. In this section we briefly describe a number of bounding procedures which exploit this basic idea.

The Gilmore-Lawler bound

Consider an instance of the Lawler QAP (2) with coefficients $C = (c_{ijkl})$, and partition the array C into n^2 matrices of dimension $n \times n$, $C^{(i,j)} = (c_{ijkl})$, for each fixed pair (i, j), $i, j = 1, 2, \ldots, n$. Each matrix $C^{(i,j)}$ essentially contains the costs associated with the assignment $x_{ij} = 1$. Partition the solution array $Y = (y_{ijkl})$ also into n^2 matrices, $Y^{(i,j)} = (y_{ijkl})$, for fixed $i, j = 1, 2, \ldots, n$.

For each pair (i,j), $1 \le i,j \le n$, solve the LAP with cost matrix $C^{(i,j)}$ and denote its optimal value by l_{ij}:

$$\begin{aligned} l_{ij} = \min \quad & \sum_{k=1}^{n}\sum_{l=1}^{n} c_{ijkl} y_{ijkl} && (36)\\ \text{s.t.} \quad & \sum_{k=1}^{n} y_{ijkl} = 1, \quad l = 1,2,\ldots,n, \\ & \sum_{l=1}^{n} y_{ijkl} = 1, \quad k = 1,2,\ldots,n, \\ & y_{ijij} = 1 && (37)\\ & y_{ijkl} \in \{0,1\}, \quad i,j = 1,2,\ldots,n. && (38) \end{aligned}$$

Observe that constraint (37) essentially reduces the problem into an LAP of dimension $(n-1)$ with cost matrix obtained from $C^{(i,j)}$ by deleting its i-th row and j-th column. For each i,j, denote by $Y^{(i,j)}$ the optimal solution matrix of the above LAP.

The Gilmore-Lawler lower bound GLB(C) for the Lawler QAP with coefficient array C is given by the optimal value of the LAP of size n with cost matrix (l_{ij})

$$\begin{aligned} GLB(C) = \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} l_{ij} x_{ij} && (39)\\ \text{s.t.} \quad & (x_{ij}) \in \mathbf{X}_n\,. \end{aligned}$$

Denote by $X^* = (x^*_{ij})$ the optimal solution matrix of this last LAP. If $\frac{1}{n}\sum_{ij} x^*_{ij} Y^{(ij)} \in \mathbf{X}_n$, then the array $Y^* = (y^*_{ijkl})$ with matrices $Y^{(i,j)^*} = x^*_{ij} Y^{(ij)}$ for all i,j, $1 \le i,j \le n$, is a Kronecker product of two permutation matrices of dimension n, and hence an optimal solution of the considered QAP. Since each LAP can be solved in $O(n^3)$ time, the above lower bound for the Lawler QAP (2) of dimension n can be computed in $O(n^5)$ time.

For the more special Koopmans-Beckmann QAP (1), where the quadratic costs c_{ijkl} are given as entry-wise products of two matrices $F = (f_{ij})$ and $D = (d_{ij})$, $c_{ijkl} = f_{ij} d_{kl}$ for all i,j,k,l, the computational effort can be reduced to $O(n^3)$. This is due to the following well known result of Hardy, Littlewood, and Pólya [92]:

Proposition 6.1 (Hardy, Littlewood and Pólya [92], 1952)
Given two n-dimensional real vectors $a = (a_i)$, $b = (b_i)$ such that $0 \le a_1 \le a_2 \le \ldots \le a_n$ and $b_1 \ge b_2 \ge \ldots \ge b_n \ge 0$, the following inequalities hold for any permutation ϕ of $1,2,\ldots,n$:

$$\sum_{i=1}^{n} a_i b_i \le \sum_{i=1}^{n} a_i b_{\phi(i)} \le \sum_{i}^{n} a_i b_{n-i+1}$$

Given two arbitrary nonnegative vectors $a, b \in \mathbb{R}^n$, let ϕ be a permutation which sorts a non-decreasingly and ψ a permutation which sorts a non-increasingly. Moreover, let π be a permutation which sorts b non-increasingly. We denote

$$\langle a,b\rangle^- := \sum_{i=1}^{n} a_{\phi(i)} b_{\pi(i)} \qquad \langle a,b\rangle^+ := \sum_{i=1}^{n} a_{\psi(i)} b_{\pi(i)} \tag{40}$$

Consider now an instance (1) of the Koopmans-Beckmann QAP. This can be written as a Lawler QAP of the form (2) by setting

$$c_{ijkl} := \begin{cases} f_{ik}d_{jl}, & \text{for} \quad i \neq k, j \neq l \\ f_{ii}d_{jj} + b_{ij}, & \text{for} \quad i = k, j = l. \end{cases}$$

Each matrix $C^{(i,j)}$ of the array C is then given by $C^{(i,j)} = (f_{ik}d_{jl})$. Therefore, instead of solving n^2 LAPs we can easily compute the values l_{ij} by applying Proposition 6.1, as

$$l_{ij} = f_{ii}d_{jj} + b_{ij} + \langle \hat{f}_{(i,.)}, \hat{d}_{(j,.)} \rangle^{-}, \tag{41}$$

where $\hat{f}_{(i,.)}, \hat{d}_{(j,.)} \in \mathbb{R}^{n-1}$ are $(n-1)$-dimensional vectors obtained from the i-th and the j-th row of F and D by deleting the i-th and the j-th element, respectively. Finally, by solving the LAP with cost matrix (l_{ij}) as in (39), we obtain the Gilmore-Lawler lower bound for the Koopmans-Beckman QAP. The appropriate sorting of the rows and columns of F and D can be done in $O(n^2 \log n)$ time. Then the computation of all l_{ij} takes $O(n^3)$ time and the same amount of time is needed to solve the last LAP.

Similar bounds have been proposed by Christofides and Gerrard [48]. The basic idea relies again on decomposing the given QAP into a number of subproblems which can be solved efficiently. First solve each subproblem, then build a matrix with the optimal values of the subproblems, and solve an LAP with that matrix as cost matrix to obtain a lower bound for the given QAP. Christofides et al. decompose the Koopmans-Beckmann $QAP(F, D)$ based on isomorphic-subgraphs of graphs whose weighted adjacency matrices are F and D. The GLB is obtained as a special case, if these subgraphs are stars, and it generally outperforms the bounds obtained by employing other subgraphs, like single edges, or double stars (see also [74]).

The Gilmore-Lawler bound is simple to compute, but it deteriorates fast as n increases. The quality of this lower bound can be improved if the given problem is transformed such that the contribution of the quadratic term in the objective function is decreased by moving costs to the linear term. This is the aim of the so-called *reduction methods*.

Reduction methods

Consider a Lawler QAP as in (2), and assume that $b_{ij} = 0$ for all i, j. By the above discussion the GLB will be given as solution of the following LAP

$$\begin{aligned} \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n}(l_{ij} + c_{ijij})x_{ij} \\ \text{s.t.} \quad & (x_{ij}) \in \mathbf{X}_n. \end{aligned} \tag{42}$$

We want to decompose the cost coefficients in the quadratic term of (2) and transfer some of their value into the linear term such that $c_{ijij} \gg l_{ij}$. This would yield a tighter lower bound because the LAP can be solved exactly. This procedure is known as *reduction* and was introduced by Conrad [54]. Reductions have been investigated by many researchers (see [21, 162, 62, 70]). The general idea is to decompose each quadratic cost coefficient into several terms so as to guarantee that some of them end up in being linear cost coefficients and can be moved in the linear term of the objective function. Consider the following general decomposition scheme:

$$\textbf{D-1:} \quad c_{ijkl} = \bar{c}_{ijkl} + e_{ijk} + g_{ijl} + h_{ikl} + t_{jkl}, \quad i \neq k,\ j \neq l,$$

where $e, g, h, t \in \mathbb{R}^{n^3}$. Substituting the above in the objective function of (2) we obtain a new QAP which is equivalent with the given one and whose objective function has a quadratic and a linear part. (Formulas for the coefficients of this new QAP can be found in the literature, e.g. [70].) For the quadratic term we can compute the Gilmore-Lawler bound. Then we add it to the optimal value of the linear part in order to obtain a lower bound for the QAP.
In the case of the Koopmans-Beckman QAP the general decomposition scheme is

$$\textbf{D-2:} \qquad \begin{aligned} f_{ij} &= \bar{f}_{ij} + \lambda_i + \mu_j, \quad i \neq j, \\ d_{kl} &= \bar{d}_{kl} + \nu_k + \phi_l, \quad k \neq l, \end{aligned}$$

where $\lambda, \mu, \nu, \phi \in \mathbb{R}^n$.

Frieze and Yadegar [70] have shown that the inclusion of vectors h and t in **D-1**, or similarly the inclusion of vectors μ and ϕ in **D-2**, does not affect the value of the lower bound. Therefore these vectors are redundant.

As mentioned also in Section 4.3, Frieze and Yadegar derived lower bounds for the QAP based on a Lagrangean relaxation of the mixed integer linear programming formulation (17)-(24). By including the constraints (19) and (20) in the objective function (17) and using vectors e and g as Lagrangean multipliers, we get the following Lagrangean problem

$$\begin{aligned} &\mathcal{L}(e,g) = \\ &\min \left\{ \textstyle\sum_{ijkl} c_{ijkl} y_{ijkl} + \sum_{jkl} e_{jkl} \left(x_{kl} - \sum_i y_{ijkl}\right) + \sum_{ikl} g_{ikl} \left(x_{kl} - \sum_j y_{ijkl}\right) \right\} = \\ &\textstyle\sum_{ijkl} (c_{ijkl} - e_{jkl} - g_{ikl}) y_{ijkl} + \sum_{ij} \left(\sum_k e_{kij} + \sum_l g_{lij}\right) x_{ij} \\ &\text{s.t.} \quad \text{constraints (18), (21),...,(24).} \end{aligned}$$

As proved in [70], for any choice of e and g, the solution to the above Lagrangean problem equals the value of the GLB obtained after the decomposition of the coefficient c_{ijkl} by using only vectors e and g in **D-1**. Therefore, $\max_{e,g} \mathcal{L}(e,g)$ constitutes a lower bound for the QAP which is larger (i.e., better) than all GLBs obtained after applying reduction methods according to **D-1** (**D-2**). Frieze and Yadegar propose two subgradient algorithms to approximately solve $\max_{e,g} \mathcal{L}(e,g)$, and obtain two lower bounds, denoted by $FY1$ and $FY2$. These bounds seem to be sharper than the previously reported Gilmore-Lawler bounds obtained after applying reductions.

Bounding techniques based on reformulations

Consider the Lawler QAP with a linear term in the objective function:

$$\begin{aligned} \min \quad & \sum_{i,k=1}^{n} \sum_{j,l=1}^{n} c_{ijkl} x_{ik} x_{jl} + \sum_{i,k=1}^{n} b_{ik} x_{ik} \\ \text{s.t.} \quad & \\ & \sum_{i=1}^{n} x_{ik} = 1, \quad 1 \leq k \leq n, \\ & \sum_{k=1}^{n} x_{ik} = 1, \quad 1 \leq i \leq n, \\ & x_{ik} \in \{0,1\}, \quad 1 \leq i,k \leq n. \end{aligned}$$

As already mentioned in Section 1, we assume without loss of generality that the coefficients c_{ijkl}, $1 \le i,j,k,l \le n$ are nonnegative.

A *reformulation* of this QAP is another QAP of the same form with new coefficients c'_{ijkl}, $1 \le i,j,k,l \le n$, and b'_{ik}, $1 \le i,k \le n$, such that for all permutation matrices (x_{ij})

$$\sum_{i,k=1}^{n}\sum_{j,l=1}^{n} c_{ijkl}x_{ik}x_{jl} + \sum_{i,k=1}^{n} b_{ik}x_{ik} = \sum_{i,k=1}^{n}\sum_{j,l=1}^{n} c'_{ijkl}x_{ik}x_{jl} + \sum_{i,k=1}^{n} b'_{ik}x_{ik},$$

holds. The basic idea is to derive a sequence of reformulations of the given problem by applying some "appropriate" reformulation rule. When we compute the GLB for each reformulation in the sequence, the best among these bounds is a valid bound for the original QAP. The reformulation rule is "appropriate" if the sequence of GLBs computed for the reformulations is monotonically nondecreasing. Usually, the construction of a new reformulation exploits the previous reformulations and the bounds obtained for them. Carraresi and Malucelli in [40] have proposed the following scheme to derive the coefficients of the reformulation

$$c'_{ijkl} = c_{ijkl} + \tau_{ijkl} - \alpha_{ijl} - \beta_{jkl} + \theta_{ik}, \quad 1 \le i,j,k,l \le n,$$

$$b'_{ik} = b_{ik} + \sum_{j=1}^{n}\alpha_{ijk} + \sum_{l=1}^{n}\beta_{ikl} - (n-1)\theta_{ik}, \quad 1 \le i,k \le n.$$

This type of bounding strategies has been proposed by Carraresi and Malucelli [39] and Assad and Xu [8]. The parameters α, β, τ and θ are updated in each reformulation step. Their values are determined by making use of the lower bound obtained for the last reformulation and the optimal values and the dual variables of the linear assignment problems solved during the last GLB computation. Clearly, not all choices of the parameters τ, α, β and θ in the above formulas produce a reformulation but there are settings of those parameters which do so, as shown in [8, 39].
To illustrate the idea consider the reformulation formulas proposed by Carraresi and Malucelli in [40]:

$$\tau_{ijkl}^{(t+1)} = c_{ijkl}^{(t)} - c_{jilk}^{(t)}, \tag{43}$$

$$\alpha_{ijl}^{(t+1)} = u_{ijl}^{(t)}, \tag{44}$$

$$\beta_{jkl}^{(t+1)} = v_{jkl}^{(t)}, \tag{45}$$

$$\theta_{ik}^{(t+1)} = \frac{1}{n-1}\left(c_{ik}^{(t)} + u_{i}^{(t)} + v_{k}^{(t)}\right), \tag{46}$$

for all $1 \le i,j,k,l \le n$. Here t is an index which counts the reformulations, $u_{ijl}^{(t)}$, $1 \le i \le n$, and $v_{jkl}^{(t)}$, $1 \le k \le n$, are the optimal values of the dual variables of the LAP with cost matrix $(c_{ijkl}^{(t)} + b_{jl}^{(t)})$, for $1 \le j,l \le n$. Let $l_{ik}^{(t)}$ be the optimal values of these LAPs, $1 \le i,k \le n$. Then $u_i^{(t)}$, $1 \le i \le n$, and $v_k^{(t)}$, $1 \le k \le n$, are optimal values of the dual variables for the LAP with costs matrix $(l_{ik}^{(t)} + b_{ik}^{(t)})$ (i.e., the last LAP solved to compute the GLB of the t-th reformulation). The bound produced with these settings is often denoted by CMB in the literature. Clearly, the computation of CMB (as well as the computation of the bounds obtained by applying the reformulation schemes proposed in [8, 39]) involves $O(n^5)$ elementary operations per iteration.

The reformulation schemes generally produce bounds of good quality. However, these bounding techniques are quite time-consuming, as $n^2 + 1$ linear assignment problems per iteration have to be solved. Finally it has been shown in [39] that in the case that $c_{ijkl} = c_{jilk}$, for all $1 \leq i, j, k, l \leq n$, the general reformulation scheme cannot produce lower bounds which are better than the optimal value of the continuous relaxation of the mixed integer programming formulation of Frieze and Yadegar.

Lower bounds for the QAP based on a dual formulation

More recently another bounding procedure which shares the basic idea of the GLB has been proposed by Hahn and Grant [90, 91]. This procedure combines GLB ideas with reduction steps in a general framework which works also for the Lawler QAP (2). The resulting bound is denoted by HGB. Recall that we assume w.l.o.g. that all c_{ijkl} in (2) are nonnegative. As described in 2.4 the four dimensional array $C = (c_{ijkl})$ is thought as being an $n^2 \times n^2$ matrix composed of n^2 submatrices $C^{(i,j)}$, $1 \leq i, j \leq n$, where each $C^{(i,j)}$ is an $n \times n$ matrix given by $C^{(i,j)} = (c_{ijkl})$. This structure of C complies with the structure of the Kronecker product $X \otimes X$, where X is an $n \times n$ permutation matrix. The entries c_{ijij} are called *leaders*. Clearly, there is only one leader in each matrix $C^{(i,j)}$. The objective function value corresponding to permutation ϕ consists of the sum of those entries c_{ijkl} which correspond to 1-entries in the Kronecker product $X_\phi \otimes X_\phi$, where X_ϕ is the permutation matrix corresponding to permutation ϕ. Hence, entries of the form c_{ijil}, $j \neq l$, or c_{ijkj}, $i \neq k$, do not contribute to the value of the objective function. Such entries are called *disallowed entries*. Entries which are not disallowed are said to be *allowed*.

The bounding procedure uses the following classes of operations acting on the matrix (c_{ijkl}):

(R1) Add a constant to all allowed entries of some row (column) of some submatrix $C^{(ij)}$ and either subtract the same constant from the allowed entries of another row (column) of the same submatrix, or subtract it from the leader in that submatrix.

(R2) Add a constant to all allowed entries of some row (column) of the $n^2 \times n^2$ matrix (c_{ijkl}).

Clearly, operations of class R1 do not change the objective function; They just redistribute the entries of the submatrices $C^{(ik)}$. Operations of class R2 add a constant to the objective function, and hence they maintain the order of permutations with respect to the corresponding values of the objective function. The main idea is then to transform C by applying operations of the classes R1 and R2 so as to decrease the objective function by some amount, say R, and to preserve the nonnegativity of entries of the transformed array C'. Then, clearly, R is a lower bound for the optimal solution of the given QAP. If, moreover, the 0-entries in the transformed matrix C' comply with the pattern of zeros in the Kronecker product $X_\phi \otimes X_\phi$ for some permutation matrix X_ϕ, then R is the optimal value of the original QAP and permutation ϕ is an optimal solution.

The procedure developed to find such a lower bound R, or possibly, to optimally solve the problem, is essentially similar to the Hungarian method for the linear assignment problem. It uses operations of classes R1 and R2 to redistribute the entries of C so as to obtain a pattern of zeros which complies with the pattern of zeros of the Kronecker product $X \otimes X$ for some permutation matrix X. The whole process is a repeated computation of Gilmore-Lawler bounds on iteratively transformed problem data, where the transformations

generalize the ideas of reduction methods. The time complexity of each iteration is basically that of the GLB computation for a Lawler QAP (i.e. $O(n^5)$).

A deeper investigation of this bounding procedure reveals that it is an iterative approach in which the dual of some LP relaxation of the original problem is solved and reformulated iteratively (see Karisch, Çela, Clausen and Espersen [104]). The reformulation step makes use of the information furnished by the preceding solution step. Some more details of this interpretation are given in Section 6.2.

As reported in [90] this bounding procedure has been tested on small and middle sized QAP instances from QAPLIB [34]. The computational results show an improved trade-off between quality of bounds and computation time, when compared to other bounding techniques. Other computational results of Hahn et al. [91] show that it is promising to involve the HGB in branch and bound approaches.

6.2 Bounds Based on Linear Programming Relaxations

As we saw in Section 4 several mixed integer linear programming (MILP) formulations have been proposed for the QAP. Clearly, the optimal solution of the continuous relaxation of an MILP formulation is a lower bound for the optimal value of the corresponding QAP. Moreover, each feasible solution of the dual of this relaxation is also a lower bound. The identification of appropriate continuous relaxations of MILP formulations, and the development of solution methods to solve these relaxations or their duals, have been important aspects of research on the QAP.

In the context of lower bound computation two MILP formulations of the QAP play a special role: The formulation of Frieze and Yadegar [70] described in Section 4.3 and that of Adams and Johnson [3] described in Section 4.4.

As we have already mentioned Frieze and Yadegar consider a Lagrangean relaxation of their MILP formulation and develop two subgradient optimization based algorithms to approximately solve the latter. The resulting bounds denoted by FY1 and FY2, respectively, perform better than the Gilmore-Lawer bound.

Adams and Johnson build upon the MILP formulation of Frieze and Yadegar and propose a slightly different MILP formulation. As shown in [3] the continuous relaxation of this formulation is tighter than the continuous relaxation of the formulation of Frieze et al. in the sense that the optimal value of the former may be strictly larger than that of the latter. Moreover, the constraints of the continuous relaxation of the formulations of Frieze et al. can be obtained as a linear combination of the constraints of the continuous relaxation of the formulation of Adams and Johnson.

Adams et al. consider a Lagrangean relaxation of (25) obtained by adding the so-called *complementary constraints* (26) to the objective function with Lagrangean multipliers α_{ijkl}. This Lagrangean relaxation denoted by $AJ(\alpha)$ is given below

$$\begin{aligned}\min \quad & \sum_{i=1}^{n}\sum_{\substack{j=1\\j>i}}^{n}\sum_{k=1}^{n}\sum_{\substack{l=1\\l\neq k}}^{n}(c_{ijkl}-\alpha_{ikjl})y_{ikjl}- \\ & \sum_{i=1}^{n}\sum_{\substack{j=1\\j<i}}^{n}\sum_{k=1}^{n}\sum_{\substack{l=1\\l\neq k}}^{n}(c_{ijkl}-\alpha_{jlik})y_{ikjl}+\sum_{i=1}^{n}\sum_{k=1}^{n}a_{ik}b_{ik}x_{ik}\end{aligned}$$

s.t.

$$
\begin{aligned}
&\sum_{i=1}^{n} x_{ik} = 1, && 1 \le k \le n,\\
&\sum_{k=1}^{n} x_{ik} = 1, && 1 \le i \le n,\\
&\sum_{j=1}^{n} y_{ijkl} = x_{ik}, && 1 \le i,k,l \le n,\\
&\sum_{l=1}^{n} y_{ijkl} = x_{ik}, && 1 \le i,j,k \le n,\\
&x_{ik} \in \{0,1\}, && 1 \le i,k \le n,\\
&0 \le y_{ijkl} \le 1, && 1 \le i,j,k,l \le n.
\end{aligned}
\qquad (\mathrm{AJ}(\alpha))
$$

Let $\theta(\alpha)$ denote the the optimal value of $AJ(\alpha)$. Then $\max_\alpha \theta(\alpha)$ equals the optimal value of the continuous relaxation of (25). Adams and Johnson [3] show that for each fixed set of the multipliers α the problem AJ(α) can be solved efficiently by solving n^2+1 LAPs, where n is the size of the considered QAP. Moreover they develop an iterative dual ascent procedure to approximately solve the above maximization problem. In each iteration problem AJ(α) is solved to optimality and the optimal value $\theta(\alpha)$ is computed. Clearly, $\theta(\alpha)$ is a lower bound for the considered QAP. Then the multipliers α_{ijkl} are updated by using the information contained in the dual variables of the LAPs solved during the previous iteration. The algorithm stops after having performed a prespecified number of iterations, and then clearly, the solution it outputs gives a lower for the original QAP. These bounds are denoted by AJB. Adams and Johnson propose two updating rules for the multipliers, one of them leading to a non-decreasing sequence of lower bounds $\theta(\alpha)$. In both cases the time complexity of this bounding procedure is dominated by the solution of n^2+1 LAPs in each iteration and amounts to $O(n^5)$ per iteration.

The strength of AJB relies on the fact that it generalizes and unifies all Gilmore-Lawler-like bounds (see Section 6.1) but the HGB. Adams et al. have shown that $\theta(0)$ equals the Gilmore-Lawler bound whereas GLBs obtained after applying reductions as well as the bounds of Carraresi and Malucelli [39] and Assad and Xu [8] equal $\theta(\alpha)$ for special settings of the Lagrangean multipliers α_{ijkl}. ¿From a practical point of view numerical experiments with instances from QAPLIB show that AJB generally outperforms the above mentioned bounds. However, according to the numerical results reported in [3, 90], HGB outperforms AJB in terms of quality, while having higher computation time requirements.

The theoretical relationship between AJB and HGB has been investigated recently by Karisch, Çela, Clausen and Espersen [104]. It turns out that unlike other Gilmore-Lawler-like bounds, HGB cannot be obtained by applying the algorithm of Adams and Johnson to solve the Lagrangean relaxation. However, both AJB and HGB can be obtained as feasible solutions of the dual of the continuous relaxation of the MILP formulation (25) proposed by Adams and Johnson. Karisch et al. propose an iterative algorithm to approximately solve this dual, and show that AJB, HGB, and all other Gilmore-Lawler-like bounds can be obtained by applying this algorithm with specific settings for the control parameters. Moreover, the same authors identify a setting of parameters which seems to produce a bound which is competitive with HGB in terms of quality and provides a better time/quality trade-off. This bound denoted by KCCEB seems to be especially suitable for use within branch and bound algorithms (see [104] for more details).

Concerning the solution to optimality of the continuous relaxation of (25), Adams and Johnson point out that the resulting linear program (LP) is highly degenerated, and degeneracy poses a problem for primal approaches. An effort to solve this LP relaxation has been done by Resende, Ramakrishnan and Drezner [158]. These authors use an interior point approach to solve the LP relaxation for QAP instances of size smaller than or equal

to 30 taken from QAPLIB [34]. For larger instances the memory requirements become prohibitive. The bounds of Resende et al., frequently denoted by IPLP, turn out to be the best existing bounds for a large number of test instances from QAPLIB. However, the computation of the IPLP bounds requires very high computation times (see [158]) and therefore, the IPLP bounds cannot be used within branch and bound algorithms, despite their good quality.

The HGB bound of Hahn et al. [90] and the KCCEB bound of Karisch et al. [104] seem to be the only linearization bounds comparable with IPLP, in terms of tightness. Moreover, generally, HGB can be computed much faster than IPLP, whereas KCCEB seems to be computable at least one order of magnitude faster than IPLP (see [104]).

6.3 Variance Reduction Lower Bounds

The variance reduction lower bounds were introduced by Li, Pardalos, Ramakrishnan and Resende in [123]. Consider an instance of the Koopmans-Beckmann QAP of size n, with flow and distance matrices $F = (f_{ij})$ and $D = (d_{ij})$. Partition both matrices as $F = F_1 + F_2$ and $D = D_1 + D_2$, where $F_1 = (f_{ij}^{(1)})$, $F_2 = (f_{ij}^{(2)})$ and $D_1 = (d_{ij}^{(1)})$, $D_2 = (d_{ij}^{(2)})$, and define a new $n \times n$ matrix $L = (l_{ij})$, by solving the following n^2 LAPs

$$l_{ij} := \min_{\substack{\phi \in \mathcal{S}_n \\ \phi(i)=j}} \sum_{k=1}^{n} \left(f_{ik}^{(1)} d_{j\phi(k)}^{(1)} + f_{ki}^{(2)} d_{\phi(k)j} + f_{ki} d_{\phi(k)j}^{(2)} - f_{ki}^{(2)} d_{\phi(k)j}^{(2)} \right). \tag{47}$$

It has been shown in [123] that the solution of the LAP with cost matrix L constitutes a lower bound for the considered QAP. The problem of concern now is to choose F_1, F_2 and D_1, D_2 such that the resulting lower bound is maximized. Notice that by setting $F_1 = F$ and $D_1 = D$ we obtain the GLB.

Given an $m \times n$ matrix M, denote its rows and columns $m_{(i.)}$, and $m_{(.j)}$, $i, j = 1, \ldots, n$, respectively. Think of M as a data set of mn elements m_{ij}, and define an average $\gamma(M)$ and a variance $V(M)$ as

$$\gamma(M) := \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} m_{ij}, \qquad V(M) := \sum_{i=1}^{m} \sum_{j=1}^{n} (\gamma(M) - m_{ij})^2.$$

Also define the total variance

$$T(M, \lambda) := \lambda \sum_{i=1}^{m} V(m_{(i.)}) + (1 - \lambda) V(M), \qquad \lambda \in [0, 1].$$

The term $V(m_{(i.)})$ stands for the variance of $m_{(i.)}$, treated as an $1 \times n$ matrix. Li et al. observed that as the variances of the matrices F and D decrease, the GLB increases. Moreover, GLB becomes maximum if the variances of the rows of the matrices equal zero. The partition scheme considered is of the form $F_1 = F + \Delta_F, F_2 = -\Delta_F$, and $D_1 = D + \Delta_D, D_2 = -\Delta_D$. We will only describe how Δ_F is obtained; Δ_D is then obtained in an analogous way. Thus, the problem is to find a matrix Δ_F, such that the variances of F_1 and F_2 and the sum of the variances of the rows for each F_1 and F_2 are minimized. This problem can be stated mathematically as

$$\min \qquad \theta T(F + \Delta_F, \lambda) + (1 - \theta) T(-\Delta_F^T, \lambda), \tag{48}$$

where $\Delta_F = (\delta_{ij})$ is an $n \times n$ matrix and $\theta \in [0,1]$ is a parameter. Two approximate solutions

$$\begin{aligned} \textbf{R-1:} \quad & \delta_{ij} = \theta(f_{nn} - f_{ij}) + \delta_{nn}, \;\; i,j = 1,\ldots,n, \\ \textbf{R-2:} \quad & \delta_{ij} = \theta(\gamma(f_{(.n)}) - \gamma(f_{(.j)}) \;\; i,j = 1,\ldots,n\,, \end{aligned}$$

where δ_{nn} is arbitrary, were proposed in [123], The matrix Δ_D is constructed in the same way. After the partitioning of the matrices F and D according to **R-1** or **R-2**, the solution to the LAP with cost matrix $L = (l_{ij})$ (where l_{ij} are defined in (47)) yields the bounds $LB1(\theta)$ or $LB2(\theta)$, respectively.
Notice that **R-2** is obtained under the assumption that the columns of the matrix Δ_F (Δ_D) are constant. This fact can be used to speed the computation of $LB2(\theta)$ by applying Proposition 6.1.
In the case of computing $LB1(\theta)$, the direct approach would be to solve n^2 LAPs defined in (47), and this would require $O(n^5)$ elementary operations. A different approach is to calculate lower bounds $\hat{l}_{ij}$ for the values $l_{ij},\ i,j = 1,\ldots,n$, and to solve than the LAP with cost matrix $(\hat{l}_{ij})$

$$\hat{l}_{ij} \;\; := \;\; \langle \hat{f}^{(1)}_{(i,.)}, \hat{d}^{(1)}_{(j,.)} \rangle^- + \langle \hat{f}^{(2)}_{(.,i)}, \hat{d}_{(.,j)} \rangle^- + \langle \hat{f}_{(.,i)}, \hat{d}^{(2)}_{(.,j)} \rangle^- + \langle \hat{f}^{(2)}_{(.,i)}, \hat{d}^{(2)}_{(.,j)} \rangle^+ .$$

It takes $O(n^3)$ time to compute all $\hat{l}_{ij}$ and the same time to solve the final LAP. Thus, the variance reduction lower bound can be computed in $O(n^3)$ time. These lower bounds perform well on QAPs with input matrices that have high variances, but their performance reduces to that of the GLB when the variance of the matrices is small.

It is worth noting that there is also a closed form solution to problem (48) given by Jansen [96]. However, as reported in [123], using that closed form to compute the lower bounds, poses implementation obstacles.

6.4 Eigenvalue Based Lower Bounds

These bounds were introduced by Finke, Burkard, and Rendl [67], and can be applied to the Koopmans-Beckmann QAP in (1). They are based on the relationship between the objective function value of the QAP in the trace formulation (9) and the eigenvalues of its coefficient matrices. When designed and implemented carefully, these techniques produce bounds of good quality in comparison with Gilmore-Lawler-like bounds or, more generally, with bounds based on linear relaxations. However, these bounds are quite expensive in terms of computation time requirements and therefore are not appropriate for use within branch and bound algorithms. Moreover, these bounds deteriorate quickly when lower levels of the branch and bound tree are searched, as shown by Karisch, Clausen, Perregaard, and Rendl [49].

Upon the introduction of the method in [67], many improvements and generalizations have appeared [86, 87, 88, 89, 154, 155]. There is a resemblance with the Gilmore-Lawler based lower bounds in the sense that, based upon a general eigenvalue bound, reduction techniques are applied to the quadratic terms of the objective function in order to improve its quality. In this case the reduction techniques yield a significant improvement, which is not really the case with the GLB.

Bound EV

Consider the trace formulation of the QAP in (9), with F and D being real symmetric matrices (see Section 2.3), and hence having only real eigenvalues. The following theorem

describes the relations between the eigenvalues of matrices F and D and the objective function of $QAP(F, D)$:

Theorem 6.1 (Finke, Burkard, and Rendl [67], 1987)
Let D, F be symmetric $n \times n$ matrices with real entries. Denote by $\lambda = (\lambda_1, \ldots, \lambda_n)^T$ and $x_1, \ldots, x_n$ the eigenvalues and eigenvectors of F, and by $\mu = (\mu_1, \ldots, \mu_n)^T$ and $y_1, \ldots, y_n$ the eigenvalues and eigenvectors of D, respectively. Then the following two relations are true for all $X \in \mathbf{X}_n$,

$$(i) \qquad tr(FXDX^T) = \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \mu_j \langle x_i, Xy_j \rangle^2 = \lambda^T S(X) \mu$$

$$\text{where } S(X) = (\langle x_i, Xy_j \rangle^2) \text{ is a doubly stochastic matrix,}$$

$$(ii) \qquad \langle \lambda, \mu \rangle^- \leq tr(FXDX^T) \leq \langle \lambda, \mu \rangle^+.$$

By using part (ii) of Theorem 6.1 we obtain a lower bound (EVB) for the considered QAP

$$\text{EVB} \quad := \quad \langle \lambda, \mu \rangle^- + \min_{X \in \mathbf{X}_n} tr(BX^T).$$

The second term is the optimal value of an LAP and can be computed efficiently.
EVB is not a strong bound. It often takes a negative value for QAP instances with non-negative coefficients. According to Theorem 6.1 the smaller the interval $[\langle \lambda, \mu \rangle^-, \langle \lambda, \mu \rangle^+]$ is, the closer is $\langle \lambda, \mu \rangle^-$ to $tr(FXDX^T)$. Thus, trying to equivalently transform the given QAP so as to decrease the length of that interval is one possibility to improve EVB.

Reduction methods and bound EV1

One possibility to make the interval $[\langle \lambda, \mu \rangle^-, \langle \lambda, \mu \rangle^+]$ smaller, and hence to improve EVB, is to decompose the matrices F and D such that some amount will be transferred to the linear term, and the eigenvalues of the matrices resulting in the quadratic term are as uniform in value as possible. Define the *spread* of the matrix F as

$$spread(F) := \max \{ \mid \lambda_i - \lambda_j \mid \; : \; i, j = 1, \ldots, n \}.$$

Our goal is to minimize the spreads of the matrices that compose the quadratic term. There is no simple closed form for expressing $spread(F)$ in terms of f_{ij}, however there is a closed formula for an upper bound $m(F)$ due to Mirsky [136]

$$spread(F) \leq m(F) = \left[2 \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij}^2 - \frac{2}{n} (trF)^2 \right]^{1/2}. \tag{49}$$

Finke, Burkard, and Rendl [67] have proposed the following decomposition scheme

$$f_{ij} = \bar{f}_{ij} + e_i + e_j + r_{ij}, \tag{50}$$
$$d_{kl} = \bar{d}_{kl} + g_k + g_l + s_{kl}, \tag{51}$$

where $r_{ij} = s_{ij} = 0$, for $i \neq j$. Denote $\bar{F} = (\bar{f}_{ij})$ and $\bar{D} = (\bar{d}_{ij})$. The values of e_i and r_{ii} (g_j and s_{jj}) which minimize the function $f(e, r) = m(\bar{F})$ ($h(g, r) = m(\bar{D})$) obtained by substituting the values of $\bar{f}_{ij}$ ($\bar{d}_{ij}$) in (49) are given by closed formula, see [67].

By replacing F and D in (9) we obtain

$$tr(FXD + B)X^T = tr(\bar{F}X\bar{D} + \bar{B})X^T,$$

where $\bar{b}_{ij} = b_{ij} + f_{ii}d_{jj} + 2e_i \sum_{\substack{k=1 \\ k\neq j}}^{n} d_{jk}$. Let $\bar{\lambda} = (\bar{\lambda}_1, \ldots, \bar{\lambda}_n)$ and $\bar{\mu} = (\bar{\mu}_1, \ldots, \bar{\mu}_n)$ be the eigenvalues of matrices $\bar{F}$ and $\bar{D}$, respectively. By applying EVB to the QAP with transformed coefficient matrices we obtain a new eigenvalue bound EVB1

$$\text{EVB1} \quad := \quad \langle \bar{\lambda}, \bar{\mu} \rangle^- + \min_{X \in \mathbf{X}_n} tr\bar{B}X^T.$$

Bound EV2

If we restrict ourselves only to purely quadratic, symmetric QAPs ($f_{ii} = d_{ii} = 0$, for all i, $B = 0$), the matrix $\bar{B}$ in the above decomposition becomes $\bar{B} = cw^T$, where $c = 2(e_1, \ldots, e_n)^T$ and $w = (\sum_j d_{1j}, \ldots, \sum_j d_{nj})^T$. Therefore $\min_{X \in \mathbf{X}_n} tr(\bar{B}X^T) = \langle c, w \rangle^-$, and

$$\text{EVB1} \quad = \quad \langle \bar{\lambda}, \bar{\mu} \rangle^- + \langle c, w \rangle^- \leq \min_{X \in \mathbf{X}_n} tr(\bar{F}X\bar{D} + \bar{B})X^T.$$

One can, however, obtain a further improvement as suggested by Rendl [154] as follows. Let $S_k := \{X_1, \ldots, X_k\} \subseteq \mathbf{X}_n$, and

$$L(X_i) := \min \{ \langle c, X_i w \rangle \ : \ X_i \in \mathbf{X}_n \setminus S_{i-1} \}.$$

Thus, for any integer $k \geq 1$ we have $L(X_1) \leq L(X_2) \leq \cdots \leq L(X_k)$. In other words the set S_k contains the k best solutions (permutation matrices) of the problem $\min_{X \in \mathbf{X}_n} \langle c, X_i w \rangle$.

$Z(\bar{F}, \bar{D}, X_i)$ is the value of the objective function of $QAP(\bar{F}, \bar{D})$ yielded by solution X_i, i.e.,

$$Z(\bar{F}, \bar{D}, X_i) = tr(\bar{F}X_i\bar{D} + \bar{B})X_i^T.$$

Further define $Z(k) := \min \{ Z(\bar{F}, \bar{D}, X_i) \colon i = 1, \ldots, k \}$. Then the following inequalities hold (see [154])

$$Z(1) \geq \cdots \geq Z(k) \geq \langle \bar{\lambda}, \bar{\mu} \rangle^- + L(X_k) \geq \cdots \geq \langle \bar{\lambda}, \bar{\mu} \rangle^- + L(X_1),$$

where the equality $Z(i) = \langle \bar{\lambda}, \bar{\mu} \rangle^- + L(X_i)$ for some i implies that X_i is an optimal solution of $QAP(\bar{F}, \bar{D})$. Thus, essentially, we try to reduce the gap between the optimal value of the QAP and the lower bound EVB1, by increasing the value of the linear term $\langle c, w \rangle^-$ in the bound in k steps, where k is specified as a parameter. The generation of the set S_k is a special case of the problem of finding the k best solutions of an assignment problem. Murty [139] has given an $O(kn^3)$ algorithm to solve this problem. Rendl [154] presents an $O(n\log n + (n + \log k)k)$ algorithm for the special case where the cost matrix of the assignment problem is given as a product matrix $(c_i w_j)$.

Rendl [154] addresses two issues regarding the effectiveness of the above ranking procedure in improving the lower bound. First, if the vectors c and w have $m \leq n$ equal elements, then there are at least $m!$ permutation matrices $\{X_i\}$ such that the values $\langle c, X_i w \rangle$ are equal. This implies in turn that there will be none or small improvement in the lower bound while generating S_k for quite some number of iterations. It can be shown that c and w will have equal elements if the row sums of F and D are equal (see [67]). Hence, the ranking procedure could give good results in the case that most of the row sums of F and

D are not equal. Secondly, Rendl defines a ratio Λ called the *degree of linearity* based on the ranges of the quadratic and linear terms that compose the lower bound

$$\Lambda := \frac{\langle \bar{\lambda}, \bar{\mu} \rangle^+ - \langle \bar{\lambda}, \bar{\mu} \rangle^-}{\langle c, w \rangle^+ - \langle c, w \rangle^-}.$$

The influence of the linear term on the lower bound is inversely proportional to the value of Λ. A *small* value of Λ suggests that the ranking procedure would be beneficial for the improvement of EVB1 for symmetric, pure quadratic QAPs. For large values of Λ, we can expect that the quadratic term dominates the linear term in the objective function. In this case Finke et al. [67] suggest the following improvement of EVB1. Consider part (i) of Theorem 6.1 applied to the reduced matrices $\bar{F}$ and $\bar{D}$, and denote the elements of the matrix $S(X)$ by s_{ij}, $s_{ij} = \langle x_i, Xy_j \rangle^2$. It is easy to see that $l_{ij} \leq s_{ij} \leq u_{ij}$, where

$$u_{ij} = \max\{(\langle x_i, y_j \rangle^-)^2, (\langle x_i, y_j \rangle^+)^2\},$$

$$l_{ij} = \begin{cases} 0, & \text{if } \langle x_i, y_j \rangle^- \cdot \langle x_i, y_j \rangle^+ < 0, \\ \min\{(\langle x_i, y_j \rangle^-)^2, (\langle x_i, y_j \rangle^+)^2\}, & \text{otherwise.} \end{cases}$$

Recalling the fact that the s_{ij} are the elements of a doubly stochastic matrix, we can then form the capacitated transportation problem

$$\begin{aligned} CTP^* = \min \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} \bar{\lambda}_i \bar{\mu}_j s_{ij} \\ \text{s.t.} \quad & \sum_{i=1}^{n} s_{ij} = 1, \quad j = 1, \ldots, n, \\ & \sum_{j=1}^{n} s_{ij} = 1, \quad i = 1, \ldots, n, \\ & l_{ij} \leq s_{ij} \leq u_{ij}. \end{aligned}$$

Then, a new lower bound would be

$$EVB2 \quad = \quad CTP^* + \langle c, w \rangle^-.$$

Other eigenvalue related bounds

Rendl and Wolkowicz [155] derive a new lower bound similar to EVB2. Notice that the decomposition scheme in (50) and (51) is uniquely determined by the $4n$-dimensional vector $d := (e^T, g^T, r^T, s^T) \in \mathbb{R}^{4n}$, where $r = (r_{11}, \ldots, r_{nn})^T$ and $s = (s_{11}, \ldots, s_{nn})^T$. EVB1 is then a function of d. Maximizing this function with respect to d will result in a lower bound with the best possible decomposition with respect to both the linear and the quadratic term. Maximizing EVB1 as a function of d leads to a nonlinear, nonsmooth, nonconcave maximization problem which is hard to solve to optimality. Rendl et al. propose a steepest ascent algorithm to approximately solve this problem (see [155]). The new bound, denoted EVB3, produces the best lower bounds for a number of QAP instances from QAPLIB, with the expense, however, of high computational time requirements.

A more general approach to eigenvalue based lower bounding techniques, was employed by Hadley, Rendl and Wolkowicz [87]. Consider the following sets of $n \times n$ matrices, where I is the $n \times n$ identity matrix and $u := (1, \ldots, 1)^T$ is the n-dimensional vector of all ones:

$$\begin{aligned} \mathcal{O} &:= \{X : X^T X = I\}, && \text{set of } \textit{orthogonal matrices}, \\ \mathcal{E} &:= \{X : Xu = X^T u = u\}, && \text{set of } \textit{matrices with row} \\ & && \textit{and column sums equal to one}, \\ \mathcal{N} &:= \{X : X \geq 0\}, && \text{set of } \textit{nonnegative matrices}. \end{aligned} \tag{52}$$

It is a well known result that $\mathbf{X}_n = \mathcal{O} \cap \mathcal{E} \cap \mathcal{N}$, while the set Ω of *doubly stochastic matrices* is given as $\Omega = \mathcal{E} \cap \mathcal{N}$. Moreover, by Birkhoff's theorem [17] we know that Ω is a convex polyhedron with vertex set $\mathbf{X}_n$, i.e., $\Omega = conv\{X : X \in \mathbf{X}_n\}$. The above characterization of $\mathbf{X}_n$ implies that we get a relaxation of the QAP, if we delete one or two of the matrix sets $\mathcal{O}, \mathcal{E}$ and $\mathcal{N}$ in the intersection $\mathbf{X}_n = \mathcal{O} \cap \mathcal{E} \cap \mathcal{N}$. Obviously, the relaxation, and therefore the lower bound, will be tighter if only one of the matrix sets is excluded. In relation to Theorem 6.1, Rendl and Wolkowicz [155] have shown that

$$\begin{aligned} \min_{X \in \mathcal{O}} tr(FXDX^T) = &\quad tr(F\Lambda_F \Lambda_D^T D \Lambda_D \Lambda_F^T) = \quad \langle \lambda, \mu \rangle^-, \\ \max_{X \in \mathcal{O}} tr(FXDX^T) = &\quad tr(F\Lambda_F \Lambda_D^T D \Lambda_D \Lambda_F^T) = \quad \langle \lambda, \mu \rangle^+, \end{aligned}$$

where Λ_F, Λ_D are matrices whose columns consist of the eigenvectors of F and D, respectively, in the order specified by their minimal (maximal) inner product. In other words, the lower bound on the quadratic part of the QAP as obtained in EVB, is derived by relaxing the feasible set to the set of orthogonal matrices.

All eigenvalue bounds discussed above relax the set of permutation matrices to $\mathcal{O}$. A tighter relaxation was proposed in [86, 88], where the set of permutation matrices was relaxed to $\mathcal{O} \cap \mathcal{E}$. The authors incorporate $\mathcal{E}$ in the objective function by exploiting the fact that the vector of ones u is both a left and right eigenvector with eigenvalue 1, for any $X \in \mathbf{X}_n$. More specifically, define

$$P := [u/\|u\| \vdots V], \quad \textit{where} \quad V^T u = 0, \quad V^T V = I_{n-1}.$$

Then, V is an orthonormal basis for $\{u\}^\perp$, while $Q := VV^T$ is the orthogonal projection on $\{u\}^\perp$. The following characterization of the permutation matrices is given in [88].

Lemma 6.1 (Hadley [86], 1989, Hadley, Rendl, Wolkowicz [88], 1992)
Let X be a real $n \times n$ matrix and Y be a real $(n-1) \times (n-1)$ matrix. If

$$X = P \begin{bmatrix} 1 & 0 \\ 0 & Y \end{bmatrix} P^T, \tag{53}$$

then

$$X \in \mathcal{E}\,, \quad X \in \mathcal{N} \Leftrightarrow VYV^T \geq -uu^T/\|u\|^2\,, \quad \textit{and} \quad X \in \mathcal{O} \Leftrightarrow Y \in \mathcal{O}_{n-1}\,.$$

Conversely, if $X \in \mathcal{E}$, there exists a Y such that (53) holds.

Note that the above characterization of permutation matrices preserves the orthogonality and the trace structure of the problem. By substituting $X = -uu^T/\|u\|^2 + VYV^T$ in the

trace formulation of the QAP (9) as suggested by (53), we obtain an equivalent *projected problem* (PQAP) of dimension $n-1$ with variable matrix Y. The new lower bound, often called *elimination bound* and denoted by ELI, is obtained by dropping the requirement $VYV^T \geq -uu^t/\|u\|^2$ and simply requiring $Y \in \mathcal{O}_{n-1}$. In this way we derive a lower bound for the quadratic part of the PQAP. The linear part can be solved exactly as an LAP.

Concluding this section notice that there is a possibility to apply eigenvalue bounds to non-symmetric QAPs, i.e., QAPs with both coefficient matrices being non-symmetric. Hadley [86] and Rendl and Wolkowicz [89] show that analogous eigenvalue bounds to those for QAPs with at least one symmetric coefficient matrix can be derived for QAPs with Hermitian coefficient matrices. Moreover, these authors show that each QAP can be equivalently transformed into a QAP with Hermitian coefficient matrices.

6.5 Bounds Based on Semidefinite Relaxations

Semidefinite programming (SDP) is a generalization of linear programming where the variables are taken from the Euclidean space of matrices with the trace operator acting as an inner product. The non-negativity constraints are replaced by semidefiniteness constraints and the linear constraints are formulated in terms of linear operators on the above mentioned Euclidean space of matrices. Successful applications of semidefinite programming in discrete optimization are presented in Goemans and Williamson [82], and Lovász and Schrijver [125].
Recently, semidefinite programming relaxations for the QAP were considered by Karisch [103], Zhao [176], and Zhao, Karisch, Rendl and Wolkowicz [177]. The SDP relaxations considered in these papers are solved by interior point methods or cutting plane methods, and the obtained solutions are valid lower bounds for the QAP.
In terms of quality the bounds obtained in this way are competitive with the best existing lower bounds for the QAP. For many test instances from QAPLIB, such as some instances of Hadley, Roucairol, Nugent et al. and Taillard, they are the best existing bounds. However, due to prohibitively high computation time requirements, the use of such approaches as basic bounding procedures within branch and bound algorithms is up to now not feasible. We refer to [103, 177] for a detailed description of SDP approaches to the QAP and illustrate the idea by describing just one semidefinite programming relaxation for the QAP.

The set of $n \times n$ permutation matrices $\mathbf{X}_n$ is the intersection of the set of $n \times n$ 0-1 matrices, denoted by $\mathcal{Z}_n$, and the set $\mathcal{E}_n$ of $n \times n$ matrices with row and column sums equal to 1. Moreover, $\mathbf{X}_n$ is also the intersection of $\mathcal{Z}_n$ with the set of $n \times n$ orthogonal matrices, denoted by $\mathcal{O}_n$. Hence

$$\mathbf{X}_n = \mathcal{Z}_n \cap \mathcal{E}_n = \mathcal{Z}_n \cap \mathcal{O}_n.$$

Recall that

$$\mathcal{O}_n = \left\{X \in \mathbb{R}^{n\times n} \colon XX^T = X^TX = I\right\} \quad \text{and}$$

$$\mathcal{E}_n = \left\{X \in \mathbb{R}^{n\times n} \colon Xu = X^Tu = u\right\},$$

where I is the $n \times n$ identity matrix and u is the n-dimensional vector of all ones. Then, the trace formulation of the QAP (2.3) with the additional linear term

$$-2\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}x_{ij},$$

can be represented equivalently as follows:

$$(QAP_{\mathcal{E}}) \qquad \begin{array}{ll} \min & tr(FXDX^T - 2BX^T) \\ \text{s.t.} & \\ & XX^T = X^TX = I, \\ & Xu = X^Tu = u, \\ & x_{ij}^2 - x_{ij} = 0. \end{array}$$

In order to obtain a semidefinite relaxation for the QAP from the formulation $QAP_{\mathcal{E}}$ above, we introduce first an n^2-dimensional vector $vec(X)$. $vec(X)$ is obtained as a column-wise ordering of the entries of matrix X. Then the vector $vec(X)$ is *lifted* into the space of $(n^2+1) \times (n^2+1)$ matrices by introducing a matrix Y_X,

$$Y_X = \begin{pmatrix} x_0 & vec(X)^T \\ vec(X) & vec(X)vec(X)^T \end{pmatrix}.$$

Thus, Y_X has some entry x_0 in the left-upper corner followed by the vector $vec(X)$ in its first row (column). The remaining terms are those of the matrix

$$vec(X)vec(X)^T$$

sitting on the right lower $n^2 \times n^2$ block of Y_X.
Secondly, the coefficients of the problem are collected in an $(n^2+1) \times (n^2+1)$ matrix K given as

$$K = \begin{pmatrix} 0 & -vec(B)^T \\ vec(B) & D \otimes F \end{pmatrix},$$

where the operator vec is defined as above and $D \otimes F$ is the Kronecker product of D and F.
It is easy to see that with these notations the objective function of $QAP_{\mathcal{E}}$ equals $tr(KY_X)$. By setting $y_{00} := x_0 = 1$ as done in Zhao et al. [177], one obtains two additional constraints to be fulfilled by the matrix Y_X: Y_X is positive semidefinite and matrix Y_X is a rank-one matrix. Whereas the semidefiniteness and the equality $y_{00} = 1$ can be immediately included in an SDP relaxation, the rank-one condition is hard to handle and is discarded in an SDP relaxation. In order to assure that the rank-one positive semidefinite matrix Y_X is obtained by an $n \times n$ permutation matrix as described above, other constraints should be imposed to Y_X. Such conditions can be formulated as valid constraints of an SDP formulation for the QAP by means of some new operators, acting on matrices or vectors as introduced below. $diag(A)$ produces a vector containing the diagonal entries of matrix A in their natural order, i.e., from top-left to bottom-right. The adjoint operator $Diag$ acts on a vector V and produces a square matrix $Diag(V)$ with off-diagonal entries equal to 0 and the components of V on the main diagonal. Clearly, for an n dimensional vector V, $Diag(V)$ is an $n \times n$ matrix.
$arrow$ acts on an $(n^2+1) \times (n^2+1)$ matrix Y and produces an n^2+1 dimensional vector $arrow(Y) = diag(Y) - (0, Y_{0,1:n^2})$, where $(0, Y_{(0,1:n^2)})$ is an n^2+1 dimensional vector with first entry equal to 0 and other entries coinciding with the entries of Y lying on the 0-th row and in columns between 1 and n^2, in their natural order[1]. The adjoint operator $Arrow$ acts on an n^2+1 dimensional vector W and produces an $(n^2+1) \times (n^2+1)$ matrix $Arrow(W)$

$$Arrow(W) = \begin{pmatrix} w_0 & 1/2W_{1:n^2}^T \\ 1/2W_{(1:n^2)} & Diag(W_{1:n^2}) \end{pmatrix},$$

[1]Note here that the rows and columns of an $(n^2+1) \times (n^2+1)$ matrix are indexed by $0, 1, \ldots, n^2$.

where $W_{(1:n^2)}$ is the n^2 dimensional vector obtained from W by removing its first entry w_0. Further, we are going to consider an $(n^2+1)\times(n^2+1)$ matrix Y as composed of its first row $Y_{(0,.)}$, of its first column $Y_{(.,0)}$, and of n^2 submatrices of size $n\times n$ each, which are arranged in an $n\times n$ array of $n\times n$ matrices and produce its remaining $n^2\times n^2$ block. (This is similar to the structure of a Kronecker product of two $n\times n$ matrices, see Section 2.4 and 6.1.) The entry $y_{\alpha\beta}$, $1\le\alpha,\beta\le n^2$, will be also denoted by $y_{(ij)(kl)}$, with $1\le i,j,k,l\le n$, where $\alpha=(i-1)n+j$ and $\beta=(k-1)n+l$. Hence, $y_{(ij)(kl)}$ is the element with coordinates (j,l) within the $n\times n$ block with coordinates (i,k).
With these formal conventions let us define the so-called *block*-0-*diagonal* and *off*-0-*diagonal* operators, acting on an $(n^2+1)\times(n^2+1)$ matrix Y, and denoted by b^0diag and o^0diag, respectively. $b^0diag(Y)$ and $o^0diag(Y)$ are $n\times n$ matrices given as follows:

$$b^0diag(Y)=\sum_{k=1}^{n}Y_{(k,.)(k,.)},\quad o^0diag(Y)=\sum_{k=1}^{n}Y_{(.,k),(.,k)}\,,$$

where, for $1\le k\le n$, $Y_{(k,.)(k,.)}$ is the k-th $n\times n$ matrix on the diagonal of the $n\times n$ array of matrices, defined as described above. Analogously, $Y_{(.,k),(.,k)}$ is an $n\times n$ matrix consisting of the diagonal elements sitting on the position (k,k) of the $n\times n$ matrices (n^2 matrices altogether) which form the $n^2\times n^2$ lower right block of matrix Y. The corresponding adjoint operators B^0Diag and O^0Diag act on an $n\times n$ matrix S and produce $(n^2+1)\times(n^2+1)$ matrices as follows:

$$B^0Diag=\begin{pmatrix}0&0\\0&I\otimes S\end{pmatrix},\quad O^0Diag=\begin{pmatrix}0&0\\0&S\otimes I\end{pmatrix}.$$

Finally, let us denote by e_0 the n^2+1 dimensional unit vector with first component equal to 1 and all other components equal to 0, and let R be the $(n^2+1)\times(n^2+1)$ matrix given by

$$R=\begin{pmatrix}n&-u^T\otimes u^T\\--u\otimes u&I\otimes E\end{pmatrix}+\begin{pmatrix}n&-u^T\otimes u^T\\--u\otimes u&E\otimes I\end{pmatrix},$$

where E is the $n\times n$ matrix of all ones.

With these notations, a semidefinite relaxation for $QAP_{\mathcal{E}}$ is given as follows

$$(QAP_{R0})\qquad\begin{array}{ll}\min & tr(KY)\\ \text{s.t.}&\\ & b^0diag(Y)=I,\\ & o^0diag(Y)=I,\\ & arrow(Y)=e_0,\\ & tr(RY)=0,\\ & Y\succeq 0.\end{array}$$

where $\preceq$ is the so-called *Löwner partial order*, i.e., $A\preceq B$ if and only if $B-A\succeq 0$, that is $B-A$ is positive semidefinite.

Zhao et al. [177] have shown that an equivalent formulation for the considered QAP is obtained from QAP_{R0} by imposing one additional condition on the matrix Y, namely, the rank-one condition.

6.6 Improving Bounds by Means of Decompositions

The idea of applying so-called *decompositions* to improve lower bounds for specially structured QAPs was initially proposed by Chakrapani and Skorin-Kapov [44], and then further

elaborated by Karisch and Rendl [105]. The applicability of this approach seems to be restricted to QAPs with a very special structure, the so-called *grid QAPs* (or *rectilinear QAPs*) to be introduced below. This procedure yields the best existing bounds for many grid QAP instances from QAPLIB and a good trade off between computation time and bound quality.

A grid QAP is a Koopmans-Beckmann QAP with flow matrix F and distance matrix $D = (d_{ij})$ being the distance matrix of a uniform rectangular grid. If $d_{ij} = d_{ik} + d_{kj}$, we say that k is on the shortest path connecting i and j. The triple $u = (i, j, k)$ is then called a *shortest path triple*. The shortest path triple $v = (i, j, k)$ for which $d_{ik} = d_{kj} = 1$ is called a *shortest triangle*.

We associate a matrix $R_u = (r_{ij}^{(u)})$ to each shortest path triple $u = (k, m, l)$, and a matrix $T_v = (t_{ij}^{(v)})$ to each shortest triangle $v = (k', m', l')$, where R_u and T_v are defined by

$$r_{kl}^{(u)} = r_{lk}^{(u)} = r_{ml}^{(u)} = r_{lm}^{(u)} = 1,\ r_{km}^{(u)} = r_{mk}^{(u)} = -1,$$

$$t_{k'm'}^{(v)} = t_{l'm'}^{(v)} = t_{m'l'}^{(v)} = t_{l'k'}^{(v)} = t_{k'm'}^{(v)} = t_{m'k'}^{(v)} = 1,$$

$$r_{ij}^{(u)} = 0 \text{ and } t_{ij}^{(v)} = 0 \text{ if } \{i, j\} \not\subseteq \{k, l, m\}.$$

The set of all shortest path triples is denoted by $\mathcal{R}$ and the set of all shortest triangles is denoted by $\mathcal{T}$.
The key observation is that, for each $R_u \in \mathcal{R}$ and for each $T_v \in \mathcal{T}$, the identity permutation is an optimal solution of $QAP(R_u, D)$ and $QAP(T_v, D)$. The optimal values for these QAPs are 0 and 8, respectively, and these simple QAPs can be used to improve the quality of lower bounds for an arbitrary grid QAP. Let us decompose the distance matrix F as

$$F = \sum_{u \in \mathcal{R}} \alpha_u R_u + \sum_{v \in \mathcal{T}} \beta_v T_v + F_r, \tag{54}$$

where F_r is the *residual* matrix given as

$$F_r := F - \sum_{u \in \mathcal{R}} \alpha_u R_u + \sum_{v \in \mathcal{T}} \beta_v T_v.$$

For every choice of the parameters $\alpha_u \geq 0$, $u \in \mathcal{R}$, and $\beta_v \geq 0$, $v \in \mathcal{T}$, and for any permutation ϕ we have

$$Z(F, D, \phi) = \sum_{u \in \mathcal{R}} \alpha_u Z(R_u, D, \phi) + \sum_{v \in \mathcal{T}} \beta_v Z(T_v, D, \phi) + Z(F_r, D, \phi)\,. \tag{55}$$

Equality (55) implies

$$\min_{\phi} Z(F, D, \phi) \geq 8 \sum_{v \in \mathcal{T}} \beta_v + \min_{\phi} Z(F_r, D, \phi) \geq$$

$$8 \sum_{v \in \mathcal{T}} \beta_v + LB(F_r, D)\,,$$

where $LB(F_r, D)$ is any lower bound for the QAP with flow matrix F_r and distance matrix D. Clearly, the expression on the right hand side of (55) is a lower bound for the original

QAP. This lower bound, which depends on the vectors $\alpha = (\alpha_u)$, $\beta = (\beta_v)$, is denoted by $h(\alpha, \beta)$. Then, $h(0, 0)$ equals $LB(F, D)$), and therefore,

$$\max_{\alpha \geq 0, \beta \geq 0} h(\alpha, \beta) \geq LB(F, D)\,,$$

where a vector is said to be nonnegative if all its components are nonnegative. Hence, $\max_{\alpha \geq 0, \beta \geq 0} h(\alpha, \beta)$ is an improvement upon the bound $LB(QAP(F, D))$.

Chakrapani et al. [44] improve the Gilmore-Lawler bound (GLB), and the elimination bound (ELI), by using only the matrices R_u, $u \in \mathcal{R}$, for the decomposition. Karisch et al. [105] use the decomposition scheme (54) to improve the elimination bound (ELI) (introduced in [88]).

7 Exact Solution Methods

An exact algorithm for a combinatorial optimization problem provides the global optimal solution to the problem. In this section we will briefly discuss several exact algorithms that have been used for solving the QAP, like branch and bound, cutting plane and branch and cut algorithms.

7.1 Branch and Bound

Branch and bound algorithms have been applied successfully to many hard combinatorial optimization problems, and they appear to be the most efficient exact algorithms for solving the QAP.

The basic ingredients of branch and bound algorithms are *bounding*, *branching*, and the *selection rule*. Although many bounding techniques have been developed for the QAP the most efficient branch and bound algorithms for this problem employ the Gilmore-Lawler bound (GLB). The reason is that other bounds which outperform GLB in terms of bound quality are simply to expensive in terms of computation time. However, more recently some efforts have been made to employ other Gilmore-Lawler-like bounds in branch and bound algorithms. The bound of Hahn and Grant (HGB) [90], has been used in a branch and bound algorithm by Hahn, Grant, and Hall [91], and the results are promising. Pardalos, Ramakrishnan, Resende and Li [150] solve some previously unsolved instances from QAPLIB by applying a branch and bound algorithm which employs the variance reduction lower bound.

Three types of branching strategies are mostly used for the QAP: *single assignment branching*, see Gilmore [77], Lawler [118], *pair assignment branching* see Gavett and Plyter [74], Land [116], Nugent et al. [141], and branching based on *relative positioning* see Mirchandani and Obata [135]. The single assignment branching which is the most efficient assigns a facility to a location in each branching step, i.e., each problem is divided into subproblems by fixing the location of one of the facilities which are not assigned yet. Several rules for the choice of the facility-location pair to determine the subproblems of a new level of the search tree have been proposed by different authors. The appropriate rule usually depends on the bounding technique. If the GLB is employed the above mentioned rule is frequently formulated in terms of the reduced costs of the last assignment problem solved to bound the subproblem which is currently being branched [14, 23, 131].

The pair assignment algorithms assign a pair of facilities to a pair of locations at a branching step, whereas in relative positioning algorithms the levels of the search tree do not

correspond to the number of facilities already assigned to locations. Here the fixed assignments within each subproblem are determined in terms of distances between facilities, i.e., their relative positions. Numerical results show that pair assignment or relative positioning algorithms are outperformed by single-assignment algorithms.
Roucairol [163] developed another branching rule which does not belong to any of the above groups, the so-called *polytomic* or *k-partite branching rule*. The search tree produced by this algorithm is not binary as in most of the other approaches. In this case the GLB is employed and the branching rule is based on the solution ϕ of the last linear assignment problem solved to compute the lower bound at the current node of the search tree. Let $\mathbf{X}_n^{(i)}$ be the subset of $\mathbf{X}_n$ (the set of permutations of $\{1, 2, \ldots, n\}$) consisting of those permutations π such that $\pi(i) = \phi(i)$. Analogously, $\bar{\mathbf{X}}_n^{(i)}$ is the set of permutations $\pi \in \mathbf{X}_n$, such that $\pi(i) \neq \phi(i)$. The current node is branched into $n+1$ new nodes with sets of feasible solutions given by $\mathbf{X}_n^{(1)}, \mathbf{X}_n^{(1)} \cap \bar{\mathbf{X}}_n^{(2)}, \ldots, \mathbf{X}_n^{(1)} \cap \mathbf{X}_n^{(2)} \cap \ldots \cap \mathbf{X}_n^{(n-1)} \cap \bar{\mathbf{X}}_n^{(n)}, \mathbf{X}_n^{(1)} \cap \mathbf{X}_n^{(2)} \cap \ldots \cap \mathbf{X}_n^{(n)}$.

Another issue in the implementation of branch and bound algorithms concerns the so-called *selection rule* which determines the choice of the subproblem to be branched, i.e., the vertex of the search tree to be branched. Several strategies, ranging from problem-independent depth or breadth first search to instance dependent criteria related to the maximization of lower bounds or reduced costs, have been tested by different authors. There seems to be no clear winner among the tested strategies.
Better results on solving large size problems have been achieved lately by parallel implementations, see Pardalos and Crouse [146], Bruengger, Clausen, Marzetta, and Perregaard [19], and Clausen and Perregaard [50]. The Nugent et al. test instances [141] are widely considered as "stubborn" QAP instances and has become an obvious challenge for every new algorithm designed for solving the QAP to optimality. The largest Nugent et al. test instance which has ever been solved to optimality has size equal to 25 and has been solved by a parallel branch and bound algorithm which employs a special implementation of the GLB, see Marzetta [130].

7.2 Traditional Cutting Plane Methods

Traditional cutting plane algorithms for the QAP have been developed by a different authors, Bazaraa and Sherali [15, 16], Balas and Mazzola [9, 10, 11], and Kaufmann and Broeckx [108]. These algorithms make use of mixed integer linear programming (MILP) formulations for the QAP which are suitable for Benders' decomposition. In the vein of Benders, the MILP formulation is decomposed into a master problem and a subproblem, called also *slave problem*, where the master problem contains the original assignment variables and constraints. For a fixed assignment the slave problem is usually a linear program and hence, solvable in polynomial time. The master problem is a linear program formulated in terms of the original assignment variables and of the dual variables of the slave problem, and is solvable in polynomial time for fixed values of those dual variables. The algorithms work typically as follows. First, a heuristic is applied to generate a starting assignment. Then the slave problem is solved for fixed values of the assignment variables implied by that assignment, and optimal values of the primal and dual variables are computed. If the dual solution of the slave problem satisfies all constraints of the master problem, we have an optimal solution for the original MILP formulation of the QAP. Otherwise, at least one of the constraints of the master problem is violated. In this case, the master problem is solved with fixed values for the dual variables of the slave problem and the obtained solution is given as input to the slave problem. The procedure is then repeated until the solution of

the slave problem fulfills all constraints of the master problem.

Clearly any solution of the master problem obtained by fixing the dual variables of the slave problem to some feasible values, is a lower bound for the considered QAP. On the other side, the objective function value of the QAP corresponding to any feasible setting of the assignment variables is an upper bound. The algorithm terminates when the lower and the upper bounds coincide. Generally, the time needed for the upper and the lower bounds to converge to a common value is too large, and hence these methods may solve to optimality only very small QAPs. However, heuristics derived from cutting plane approaches produce good suboptimal solutions in early stages of the search, e.g. Burkard and Bönniger [24] and Bazaraa and Sherali [16].

7.3 Polyhedral Cutting Planes

Similarly to traditional cutting plane methods also polyhedral cutting planes or *branch and cut algorithms*[2] make use of an LP or MILP relaxation of the combinatorial optimization problem to be solved, in our case the QAP. Additionally, polyhedral cutting plane methods make use of a class of (nontrivial) valid or facet defining inequalities known to be fulfilled by *all* feasible solutions of the original problem. If the solution of the relaxation is feasible for the original problem, we are done. Otherwise, some of the above mentioned *valid* inequalities are probably violated. In this case a "cut" is performed, that is, one or more of the violated inequalities are added to the LP or MILP relaxation of our problem. The latter is resolved and the whole process is repeated. In the case that none of the valid inequalities is violated, but some integrality constraint is violated, the algorithm performs a branching step by fixing (feasible) integer values for the corresponding variable. The branching steps produce the *search tree* like in branch and bound algorithms. Each node of this tree is processed as described above by performing cuts and then by branching it, if necessary. Clearly, related elements of branch and bound algorithms like upper bounds, selection and branching rules play a role in branch and cut algorithms. Hence, such an approach combines elements of cutting plane and branch and bound methods.
The main advantage of polyhedral cutting plane algorithms with respect to traditional cutting planes relies on the use of cuts which are valid for the whole polytope of the feasible solutions, and possibly facet defining. Traditional cutting planes instead rely frequently on cuts which are not valid for the whole polytope of the feasible solutions. In this case the whole computation has to be done from scratch for different variable fixings. This requires additional running time and additional amounts of memory. Another and not less important drawback of traditional cutting plane algorithms is due to the "weakness" of the cuts they involve. In contrast with cuts produced by facet defining inequalities, the weak cuts cannot avoid the slow convergence.

As we saw in Section 5 some properties and few facet defining inequalities of the QAP polytope are already known. But still polyhedral cutting plane methods for the QAP are not yet backed by a strong theory. However, some efforts to design branch and cut algorithms for the QAP have been made by Padberg and Rijal [142] and Kaibel [102]. Padberg and Rijal [142] have tested their algorithm on sparse QAP instances from QAPLIB. The numerical results are encouraging, although the developed software is of preliminary nature, as claimed by the authors. Kaibel [102] has used branch and cut to compute lower bounds for QAP instances from QAPLIB. His results are promising especially in the case where box inequalities are involved.

[2]This term was originally used by Padberg and Rinaldi [143].

8 Heuristics

Although substantial improvements have been done in the development of exact algorithms for the QAP, problems of dimension $n > 20$ are still not practical to solve because of very high computer time requirements. This makes the development of heuristics indispensable as algorithms which provide good quality solutions in a reasonable time. Much research has been devoted to the development of such approaches. We distinguish the following types of heuristic algorithms:

- Construction methods (CM)
- Limited enumeration methods (LEM)
- Improvement methods (IM)
- Tabu search (TS)
- Simulated annealing (SA)
- Genetic algorithms (GA)
- Greedy randomized adaptive search procedures (GRASP)
- Ant systems (AS)

8.1 Construction Methods

Construction methods were introduced by Gilmore [77]. They are iterative approaches which usually start with an empty permutation, and iteratively complete a partial permutation into a solution of the QAP by assigning some facility which has not been assigned yet to some free location. The algorithm is presented in pseudocode in Figure 1. Here $\phi_0, \phi_1, \ldots, \phi_{n-1}$ are partial permutations, and `heur`(i) is some heuristic procedure that assigns facility i to some location j, and returns j. Γ is the set of already assigned pairs of facilities to locations. The procedure `update` constructs a permutation ϕ_i by adding the assignment (i, j) to ϕ_{i-1}. The heuristic `heur`(i) employed by `update` could be any heuristic which chooses a location j for facility i, $(i, j) \notin \Gamma$, in a greedy fashion or by applying local search.

One of the oldest heuristics used in practice, the CRAFT heuristic developed by Buffa, Armour and Vollmann [20], is a construction method. Another construction method which yields good results has been proposed by Müller-Merbach [140].

8.2 Limited Enumeration Methods

Limited enumeration methods rely on the observation that often enumeration methods (e.g. branch and bound algorithms) find good solutions in early stages of the search, and employ then a lot of time to marginally improve that solution or prove its optimality. This behavior of enumeration methods suggests a way to save time in the case that we are interested in a good but not necessarily optimal solution: impose some limit to the enumeration process. This limit could be a time limit, or a limit on the number of iterations the algorithm may perform.

Another strategy which serves the same goal is to manipulate the lower bound. This can be done by increasing the lower bound if no improvement in the solution is achieved during

```
procedure construction(φ_0, Γ)
1    φ = {};
2    do i = 1, ..., n − 1 →
3         if (i, j) ∉ Γ →
4              j =heur(i);
5              update(φ_i, (i, j));
6              Γ = Γ ∪ (i, j);
7         fi;
8         φ = φ_i;
9    od;
10   return(φ)
end construction;
```

Figure 1: Pseudo-code for a construction method

a large number of iterations, and would yield deeper cuts in the search tree to speed up the process. Clearly, such an approach may cut off the optimal solution and hence should be used carefully, possibly in conjunction with certain heuristics that perform elaborate searches in the feasible space.

8.3 Improvement methods

These methods belong to the larger class of *local search* algorithms. A local search procedure starts with an initial feasible solution and iteratively tries to improve the current solution. This is done by substituting the latter with a (better) feasible solution from its *neighborhood*. This iterative step is repeated until no further improvement can be found. Improvement methods are local search algorithm which allow only improvements of the current solution in each iteration. For a comprehensive discussion of theoretical and practical aspects of local search in combinatorial optimization the reader is referred to the book edited by Aarts and Lenstra [2].

Basic ingredients of improvement methods (and of local search in general) are the *neighborhood* and *the order* in which the neighborhood is scanned. Frequently used neighborhoods for QAPs are the *pair-exchange* neighborhood and the *cyclic triple-exchange* neighborhood. In the case of pair-exchanges the neighborhood of a given solution (permutation) consists of all permutations which can be obtained from the given one by applying a transposition to it. In this case, scanning the whole neighborhood, i.e., computing the objective function values for all neighbors of a given permutation, takes $O(n^3)$ time. (The size of the neighborhood is $\binom{n}{2}$, and it takes $O(n)$ steps to compute the difference of the objective function values of a permutation π and a permutation π' in the neighborhood of π.) If the neighborhood of π is already scanned and π' is a neighbor of π, then the neighborhood of π' can be scanned in $O(n^2)$, see Frieze et al. [71].
In the case of cyclic triple-exchanges, the neighborhood of a solution (permutation) π consists of all permutations obtained from π by a cyclic exchange of some triple of indices. The size of this neighborhood is $O(\binom{n}{3})$. Cyclic triple-exchanges do not really lead to better results when compared with pair-exchanges.

Another important ingredient of improvement methods is the order in which the neighborhood is scanned. This order can be either fixed previously or chosen at random. Given

a neighborhood structure and a scanning order, a rule for the update of the current solution (from the current iteration to the subsequent one) should be chosen. The following update rules are frequently used:

- First improvement
- Best improvement
- Heider's rule [94]

In the case of *first improvement* the current solution is updated as soon as the first improving neighbor solution is found. *Best improvement* scans the whole neighborhood and chooses the *best* improving neighbor solution (if such a solution exists at all). *Heider's rule* starts by scanning the neighborhood of the initial solution in a prespecified *cyclic* order. The current solution is updated as soon as an improving neighbor solution is found. The scanning of the neighborhood of the new solution starts there where the scanning of the previous one was interrupted (in the prespecified cyclic order).
In order to get better results, improvement methods and local search algorithms in general are performed several times starting with different initial solutions.

8.4 Tabu Search

Tabu search is a local search method introduced by Glover [79, 80] as a technique to overcome local optimality. One way to overcome local optimality would be to allow also the deterioration of the current solution when moving from one iteration to the subsequent one, in contrast to improvement methods. In the case of tabu search the basic idea is to "remember" which solutions have been visited in the course of the algorithm, in order to derive the promising directions for further search. Thus, the memory and not only the local investigation of the neighborhood of the current solution drives the search. The reader is referred to the book edited by Glover, Laguna, Taillard, and De Werra [81] for a comprehensive introduction to tabu search algorithms.
The main ingredients of tabu search are the *neighborhood structure*, the *moves*, the *tabu list* and the *aspiration criterion*. A *move* is an operation which, when applied to a certain solution π, generates a neighbor π' of it. In the case of QAPs the neighborhood is the pair-exchange neighborhood and the moves are usually *transpositions*. A *tabu list* is a list of forbidden or *tabu* moves, i.e., moves which are not allowed to be applied to the current solution. The tabu status of the moves changes along with the search and the tabu list is updated during the search. An *aspiration criterion* is a condition which, when fulfilled by a tabu move, cancels its tabu status.
A generic tabu search procedure starts with an initial feasible solution S and selects a *best-quality* solution among (a part of) the neighbors of S obtained by non-tabu moves. Note that this neighboring solution does not necessarily improve the value of the objective function. Then the current solution is updated, i.e., it is substituted by the selected solution. Obviously, this procedure can *cycle*, i.e., visit some solution more than once. In an effort to avoid this phenomenon a tabu criterion is introduced in order to identify moves which are expected to lead to cycles. Such moves are then declared tabu and are added to the tabu list. As, however, forbidding certain moves could prohibit visiting "interesting" solutions, an aspiration criterion distinguishes the potentially interesting moves among the forbidden ones. The search stops when a stop criterion (running time limit, limited number of iterations) is fulfilled.

There is a lot of freedom in the implementation of different elements of a tabu search algorithms, e.g. the tabu list (length and maintenance), the aspiration criterion, the tabu criterion. The performance of tabu search algorithms depends very much on the implementation chosen for its basic ingredients, and there is no general agreement about the best implementation of any of those.

Different implementations of tabu search have been proposed for the QAP, e.g. a tabu search with fixed tabu list (Skorin-Kapov [166]), the robust tabu search (Taillard [171]), where the size of the tabu list is randomly chosen between a maximum and a minimum value, and the reactive tabu search (Battiti and Tecchiolli [13]) which involves a mechanism for adopting the size of the tabu list. Reactive tabu search aims at improving the robustness of the algorithm. The algorithm notices when a cycle occurs, i.e., when a certain solution is revisited, and increases the tabu list size according to the length of the detected cycle. The numerical results show that generally the reactive tabu search outperforms other tabu search algorithms for the QAP (see [13]).
More recently, also parallel implementations of tabu search have been proposed, see e.g. Chakrapani and Skorin-Kapov [43]. Tabu search algorithms allow a natural parallel implementation by dividing the burden of the search in the neighborhood among several processors.

8.5 Simulated Annealing

Simulated annealing is a local search approach which exploits the analogy between combinatorial optimization problems and problems from statistical mechanics. Kirkpatrick, Gelatt and Vecchi [110] and Černỳ [42] were among the first authors who recognized this analogy, and showed how the Metropolis algorithm (Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller [133]) used to simulate the behavior of a physical many-particle system can be applied as a heuristic for the traveling salesman problem.

The analogy between a combinatorial optimization problem and a many-particle physical system basically relies on two facts:

- Feasible solutions of the combinatorial optimization problem correspond to states of the physical system.
- The objective function values corresponds to the energy of the the states of the physical system.

In *condensed matter physics annealing* is known as a cooling process which produces low energy *thermal equilibrium* states of a solid in a heat bath. The aim is to reach the so-called *ground state* which is characterized by a minimum of energy.
Burkard and Rendl [37] showed that a simulated cooling process yields a general heuristic which can be applied to any combinatorial optimization problem, as soon as a neighborhood structure has been introduced in the set of its feasible solutions. In particular Burkard et al. applied simulated annealing to the QAP. Other simulated annealing (SA) algorithms for the QAP have been proposed by different authors, e.g. Wilhelm and Ward [175] and Connolly [53]. All these algorithms employ the pair-exchange neighborhood. They differ on the way the *cooling process* or the *thermal equilibrium* is implemented. The numerical experiments show that the performance of SA algorithms strongly depends on the values of the control parameters, and especially on the choice of the cooling schedule.

Simulated annealing (SA) can be modeled mathematically by an inhomogeneous ergodic Markov chain, and this model has been used for the probabilistic analysis of the convergence

of simulated annealing algorithms. Under natural conditions on the involved neighborhood structure and non very restrictive conditions on the slowness of the cooling process it can be shown that SA asymptotically converges to an optimal solution of the considered problem. The investigation of the speed of this convergence remains an (apparently difficult) open problem. For a detailed discussion on the convergence and other theoretical aspects of simulated annealing the reader is referred to the books by Aarts and Korst [1] and Laarhoven and Aarts [115].

8.6 Genetic Algorithms

The so-called *genetic algorithms* (GA) are a nature inspired approach for combinatorial optimization problems. The basic idea is to adapt the evolutionary mechanisms acting in the selection process in nature to combinatorial optimization problems. The first genetic algorithm for optimization problems was proposed by Holland [95] in 1975.

A genetic algorithm starts with a set of initial feasible solutions (generated randomly or by using some heuristic) called the *initial population*. The elements of a population are usually termed "individuals". The algorithm *selects* a number of pairs of individuals or *parents* from the current population and uses so-called *cross-over rules* to produce some feasible solution or *child* out of each pair of individuals. Further, a number of "bad" solutions, i.e., solutions yielding to high values of the objective function, is thrown out of the current population. This process is repeated until a *stop criterion*, e.g. a time limit, a limit on the number of iterations, a measure of convergence, is fulfilled. In the course of the algorithm, *mutations* or *immigrations* are applied periodically to the current population to improve its overall quality by *modifying* some of the individuals or *replacing* them by better ones, respectively. Often local optimization tools are periodically used within GAs resulting in so-called *hybrid algorithms*. The search is diversified by means of so-called *tournaments*. A tournament consists of applying several runs of a GA starting from different initial populations and stopping them before they converge. A "better" population is derived as a union of the final populations of these different runs, and then a new run of the GA is started over this population. For a good coverage of theoretical and practical issues on genetic algorithms the reader is referred to Davis [56] and Goldberg [83].

A number of authors have proposed genetic algorithms for the QAP. Standard algorithms e.g. the one developed by Tate and Smith [172], have difficulties to generate the best known solutions even for QAPs of small or moderate size. Hybrid approaches, e.g. combinations of GA techniques with tabu search as the one developed by Fleurent and Ferland [68] seem to be more promising. More recently another hybrid algorithm, the so-called greedy genetic algorithm proposed by Ahuja, Orlin, and Tivari [6] produced very good results on large scale QAPs from QAPLIB.

8.7 Greedy Randomized Adaptive Search Procedure

The greedy randomized adaptive search procedure (GRASP) was introduced by Feo and Resende [66] and has been applied successfully to different hard combinatorial optimization problems [65, 111, 112, 157] and among them to the QAP [124, 148] and the BiQAP [132]. The reader is referred to [66] for a survey and tutorial on GRASP.
GRASP is a combination of greedy elements with random search elements in a two phase heuristic. It consists of a construction phase and a local improvement phase. In the construction phase good solutions from the available feasible space are constructed, whereas in the local improvement phase the neighborhood of the solution constructed in the first phase

is searched for possible improvements. A pseudocode of GRASP is shown in Figure 2. The input parameters are the size *RCLsize* of the *restricted candidate list* (RCL), a maximum number of iterations, and a random seed. RCL contains the candidates upon which the sampling related to the construction of a solution in the first phase will be performed.

```
procedure GRASP(RCLSize,MaxIter,RandomSeed)
1  InputInstance();
2  do k = 1,..., MaxIter →
3     ConstructGreedyRandomizedSolution(RCLSize,RandomSeed);
4     LocalSearch(BestSolutionFound);
5     UpdateSolution(BestSolutionFound);
6  od;
7  return BestSolutionFound
end GRASP;
```

Figure 2: Pseudo-code for a generic GRASP

For the QAP the construction phase consists of two stages. The RCL contains tuples of partial permutations and values associated to them. Each of these partial permutations fixes the location of facilities 1 and 2. Such partial permutations are called 2-permutations. In the first stage a 2-permutation is chosen randomly from the *restricted candidate list* (RCL).
Given a QAP instance of size n with flow matrix $F = (f_{ij})$ and distance matrix $D = (d_{ij})$, the value $C_{\phi,\psi}$ associated with a pair (ϕ, ψ) of 2-permutations is given as

$$C_{\phi,\psi} = \sum_{i=1}^{2}\sum_{j=1}^{2} d_{\phi(i)\phi(j)} f_{\psi(i)\psi(j)} \, .$$

Clearly, the 2-permutations ϕ, ψ can be seen as elements of the set $K = \{(i,j) \; : \; i,j = 1,2,\ldots,n, i \neq j\}$, and since $|K| = n(n-1)$, there are $n^2(n-1)^2$ pairs (ϕ,ψ) of 2-permutations. If we have a symmetric QAP instance with zeros in the diagonal, the above cost simplifies to

$$C_{\phi,\psi} = 2 d_{\phi(1)\phi(2)} f_{\psi(1)\psi(2)} \, .$$

The RCL contains a number of pairs (ϕ,ψ) - this number equals the RCL size and is denoted by *RCLsize* - having the smallest associated costs. In the case of an asymmetric QAP, we compute the costs $C_{\phi,\psi}$ for all (ϕ,ψ) and keep the *RCLsize* smallest among them. In the symmetric case, we sort the $m = n^2 - n$ off-diagonal entries of matrix D in ascending order, and the off-diagonal entries of F in descending order, i.e.,

$$d_{k_1 l_1} \leq d_{k_2 l_2} \leq \cdots \leq d_{k_m l_m},$$

$$f_{i_1 j_1} \geq f_{i_2 j_2} \geq \cdots \geq f_{i_m j_m} \, .$$

Then, the products $d_{k_s l_s} f_{i_s j_s}$ are the costs associated to pairs of 2-permutations (k_s, l_s), (i_s, j_s), $1 \leq s \leq m$, respectively. These costs are sorted in ascending order and the *RCLsize* smallest among them are put in RCL. Finally, one pair of 2-permutations from RCL is chosen at random, and these determines the locations of two facilities which are kept fixed

in the second stage of the construction phase. Notice that the RCL is constructed only once, and hence, in constant time with regard to the number of iterations.
In the second stage the remaining $n-2$ facilities are assigned to locations. Let the set Γ_r be the set of assignments made prior to the $r-th$ assignment:

$$\Gamma_r := \{(j_1, p_1), (j_2, p_2), \ldots, (j_{r-1}, p_{r-1})\}.$$

Note that at the start of stage 2, $|\Gamma_3| = 2$, since two assignments are made in the first stage, and $r = |\Gamma_r| + 1$ throughout the second stage. In stage 2 we also construct an RCL which contains the single assignments $m \to s$, $(m, s) \notin \Gamma_r$, and their associated costs $\bar{C}_{m,s}$ defined as

$$\bar{C}_{m,s} := \sum_{(i,j)\in T_r} d_{\phi(i)\phi(j)} f_{\psi(i)\psi(j)},$$

where

$$T_r := \{(i,j) \ : \ i,j = 1,2,\ldots,r, \{i,j\} \cap \{r\} \neq \emptyset\},$$

and ϕ, ψ are partial permutations resulting from the $r-1$ assignment which are already fixed and the assignment (m, s). In the case of a symmetric QAP the cost $C_{m,s}$ is given by the simpler formula

$$\bar{C}_{ms} = 2 \sum_{i=1}^{r-1} d_{m\phi(i)} f_{s\psi(i)}. \tag{56}$$

Among the $U = (n-r+1)^2$ possible assignments (m, s), those with the $RLSsize$ smallest associated costs are included in RCL. One assignment is then selected at random from RCL and the set Γ_r is updated

$$\Gamma_r = \Gamma_r \cup \{(m, s)\}.$$

This process is repeated until a permutation of $\{1, 2, \ldots, n\}$, i.e., a feasible solution of the considered QAP, results. Stage 2 of the construction phase of GRASP in pseudocode is shown in Figure 3. Procedure `ConstructionStage2` returns the set Γ with $n-1$ assignments, the last assignment being then trivial. The `inheap()` and `outheap()` procedures are used for sorting and choosing the smallest among the computed $\bar{C}_{m,s}$ costs, respectively. The procedure `random` generates a random number in the interval $[1, RLSsize]$.

Finally, the second phase of the algorithm completes a GRASP iteration by applying an improvement method starting from the solution constructed in the first phase and employing the 2-exchange neighborhood (see also Section 8.3).

8.8 Ant Systems

Ant system (AS) are recently developed heuristic techniques for combinatorial optimization which try to imitate the behavior of an ant colony in search for food. Initially the ants search for food in the vicinity of their nest in a random way. As soon as an ant finds a source of food, it takes some food from the source and carries it back to the nest. During this trip back the ant leaves a trail of a substance called pheromone on the ground. The pheromone trail serves to guide the future search of ants towards the already found source of food. The intensity of the pheromone on the trail is proportional to the amount of food found in the source. Thus, the ways to rich sources of food visited frequently (by a large number of ants) will be indicated by stronger pheromone trails. In an attempt to imitate the behavior of ants to derive algorithms for combinatorial optimization problems, the following analogies

```
procedure ConstructionStage2(α, (j1, p1), (j2, p2))
1   Γ = {(j1, p1), (j2, p2)};
2   do r = 2, ..., n − 1   →
3       U = 0;
4       do m = 1, ..., n →
5           do s = 1, ..., n →
6               if (m, s) ∉ Γ →
7                   C̄ms = Σ(i,j)∈Tr a_{p(i)p(j)} b_{q(i)q(j)};
8                   inheap(Cms);
9                   U = U + 1;
10              fi;
11          od;
12      od;
13      t =random[1, ⌊αU⌋];
14      do i = 1, ..., t →
15          C̄ms =outheap();
16      od;
17      Γ = Γ ∪ {(m, s)};
18  od;
19  return Γ
end ConstructStage2;
```

Figure 3: Stage 2 of Construction Phase of GRASP

can be exploited: a) the area searched by the ants resembles the set of feasible solutions, b) the amount of food at food sources resembles the value of the objective function, and c) the pheromone trail resembles a component of adaptive memory.
AS were originally introduced by Dorigo [59] and Colorni, Dorigo, and Maniezzo [51] and have already produced good results for well known problems like the traveling salesman problem (TSP) and the QAP [52, 72].

In the case of the QAP the pheromone trail which is also the key element of an AS, is implemented by a matrix $T = (\tau_{ij})$. τ_{ij} is a measure for the desirability of locating facility i at location j in the solutions generated by the ants (the algorithm). To illustrate the idea we briefly describe the algorithm of Gambardella, Taillard and Dorigo [72].
The algorithm is iterative and constructs a fixed number, say m, of solutions in each iteration. (This number is a control parameter and is also thought as number of ants.) In the first iteration these solutions are generated randomly, whereas in the subsequent iteration they are updated by exploiting the information contained in the pheromone trail matrix T. Initially the pheromone trail matrix is a constant matrix; the constant is inverse-proportional to the best value of the objective function found so far. This is in compliance with the behavior of ants whose search directions are initially chosen at random. Let us denote the best solution found so far by ϕ^* and its corresponding value of the objective function by $f(\phi^*)$. In the further iterations the entries $\tau_{i\phi^*(i)}$ of T are increased by the same value which is proportional to $f(\phi^*)$. The update of the m solutions in each iteration is done first by means of the pheromone trail matrix, and then by applying some improvement method. In both cases the update consists of swapping the locations for a sequence of

facility pairs. First, the current solution is updated by swapping the locations of pairs of facilities chosen so as to maximize the (normalized) sum of the corresponding pheromone trail entries. Then, the solution obtained after this update is improved by applying some improvement methods, e.g. first or best improvement (see also Section 8.3). As soon as an improvement of the best known solution is detected an intensification component "forces the ants" to further explore the part of the solution space where the improvement was found. If after a large number of iterations there is no improvement of the best known solution, a diversification - which is basically a new random start - is performed.
Numerical results presented in [52, 72] show that ant systems are competitive heuristics especially for real life instances of the QAP with a few very good solutions clustered together. For randomly generated instances which have many good solutions distributed somehow uniformly in the search space, AS are outperformed by other heuristics, e.g. genetic algorithms or tabu search approaches.

9 Available Computer Codes for the QAP

Burkard, Karisch, and Rendl [34] have compiled a library of QAP instances (QAPLIB) which is widely used to test bounds, exact algorithms, and heuristics for the QAP. The instances collected in QAPLIB are due to different authors and range from instances arising in real life applications to instances generated randomly only for test purposes. Many of these instances have not been solved to optimality yet, the most celebrated among them being the instances of Nugent, Vollmann, and Ruml [141] of size larger than 25. QAPLIB can be found at `http://www.opt.math.tu-graz.ac.at/~karisch/qaplib`.

A number of codes to compute lower bounds are available. A FORTRAN code which computes the GLB is due to Burkard and Derigs [29], and is able to compute the bound for instances of size up to 256. The source code can be downloaded from the QAPLIB web page. Another FORTRAN code which can be downloaded from the QAPLIB web page computes the elimination bound (ELI) for symmetric QAP instances of size up to 256.
Recently, Espersen, Karisch, Çela, and Clausen [64] have developed `QAPpack` which is a JAVA package containing a branch and bound algorithm to solve the QAP. In `QAPpack` a number of bounds based on linearization are implemented: the Gilmore-Lawler bound [77, 118], the bound of Carraresi and Malucelli [40], the bound of Adams and Johnson [3], the bound of Hahn and Grant [90], and the bound of Karisch, Çela, Clausen, and Espersen [104]. The implementation is based on the dual framework provided by Karisch et al. [104]. `QAPpack` can be found at `http://www.imm.dtu.dk/~te/QAPpack`.
Besides `QAPpack`, a FORTRAN code of the branch and bound algorithm developed by Burkard and Derigs [29] can be downloaded from the QAPLIB web page.

There are also some codes of heuristics available. The (compressed) FORTRAN source file - `608.Z` - of a heuristic due to West [174], can be downloaded at `ftp://netlib.att.com` in `/netlib/toms`.
The source files (compressed tar-files) of two FORTRAN implementations of GRASP for dense QAPs by Resende, Pardalos and Li [156] and sparse QAPs by Pardalos, Pitsoulis and Resende [149] can be downloaded from Resende's web page at

`http://www.research.att.com/~mgcr/src/index.html`.

The source file of a FORTRAN implementation of the simulated annealing algorithm of Burkard and Rendl [37] can be downloaded from the QAPLIB web page.

The source file of a C++ implementation of the simulated annealing algorithm of Connolly [53], due to Taillard, can be downloaded from Taillard's web page at

`http://www.idsia.ch/~eric/codes.dir/sa_qap.c`.

Also a source file of a PASCAL implementation of Taillard's robust tabu search [171] can be found at Taillard's web page.

Finally, the source file of a FORTRAN implementation of Li and Pardalos' generator for QAP instances with known optimal solution [122] can be obtained by sending an email to `coap@math.ufl.edu` with subject line `send 92006`.

10 Polynomially Solvable Cases

Since the QAP is NP-hard, restricted versions which can be solved in polynomial time are an interesting aspect of the problem. A basic question arising with respect to polynomially solvable versions is the identification of those versions and the investigation of the border line between hard and easy versions of the problem. There are two ways to approach this topic: first, find structural conditions to be imposed on the coefficient matrices of the QAP so as to obtain polynomially solvable versions, and secondly, investigate other combinatorial optimization or graph-theoretical problems which can be formulated as QAPs, and embed the polynomially solvable versions of the former into special cases of the later. These two approaches yield two groups of restricted QAPs which are briefly reviewed in this section. For a detailed information on this topic the reader is referred to [41].

Most of the restricted versions of the QAP with specially structured matrices involve Monge matrices or other matrices having analogous properties. A matrix $A = (a_{ij})$ is a *Monge matrix* iff the following inequalities are fulfilled for each 4-tuples of indices i, j, k, l, $i < k$, $j < l$:

$$a_{ij} + a_{kl} \leq a_{il} + a_{kj}, \qquad \text{(Monge inequalities)}.$$

A matrix $A = (a_{ij})$ is an *Anti-Monge matrix* iff the following inequalities are fulfilled for each 4-tuples of indices i, j, k, l, $i < k$, $j < l$:

$$a_{ij} + a_{kl} \geq a_{il} + a_{kj}, \qquad \text{(Anti-Monge inequalities)}.$$

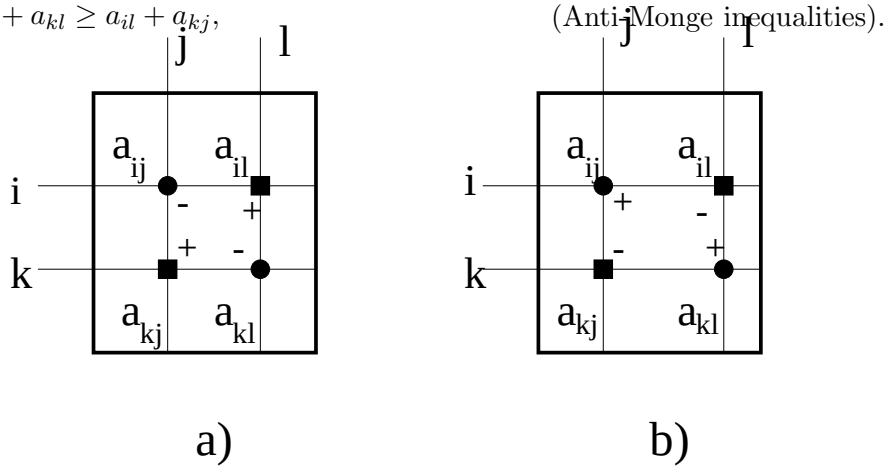

Figure 4: The sum of the depicted entries taken with the corresponding signs must be nonnegative: a) Monge inequality, b) Anti-Monge inequality.

A simple example of Monge and Anti-Monge matrices are the *sum matrices*; the entries of a sum matrix matrix $A = (a_{ij})$ are given as $a_{ij} = \alpha_i + \beta_j$, where (α_i) and (β_j) are the generating row and column vector, respectively. A *product matrix* A is defined in an analogous way: its entries are given as $a_{ij} = \alpha_i\beta_j$, where (α_i), (β_j) are the generating vectors. If the row generating vector (α_i) and the column generating vectors (β_i) are sorted non-decreasingly, then the product matrix $(\alpha_i\beta_j)$ is an Anti-Monge matrix.

In contrast with the traveling salesman problem, it turns out that the QAP with both coefficient matrices being Monge or Anti-Monge is NP-hard, whereas the complexity of a QAP with one coefficient matrix being Monge and the other one being Anti-Monge is still open, see Burkard, Çela, Demidenko, Metelski, and Woeginger [26] and Çela [41]. However, some polynomially solvable special cases can be obtained by imposing additional conditions on the coefficient matrices. These special cases involve very simple matrices like product matrices or so-called *chess-board matrices*. A matrix $A = (a_{ij})$ is a chess-board matrix if its entries are given by $a_{ij} = (-1)^{i+j}$. These QAPs can either be formulated as equivalent LAPs, or they are *constant permutation QAPs* (see [26, 41]), i.e., their optimal solution can be given before hand, without knowing the entries of their coefficient matrices. A few other versions of the QAP involving Monge and Anti-Monge matrices with additional structural properties can be solved by dynamic programming.

Other restricted versions of the QAP involve matrices with a specific diagonal structure e.g. *circulant* and *Toeplitz matrices.* An $n \times n$ matrix $A = (a_{ij})$ is called a *Toeplitz matrix* if there exist numbers $c_{-n+1}, \ldots, c_{-1}, c_0, c_1, \ldots, c_{n-1}$ such that $a_{ij} = c_{j-i}$, for all i, j.
A matrix A is called a *circulant matrix* if it is a Toeplitz matrix and the generating numbers c_i fulfill the conditions $c_i = c_{n-i}$, for $0 \leq i \leq n-1$. In other words, a Toeplitz matrix has constant entries along lines parallel to the diagonal, whereas a circulant is given by its first row and the entries of the $i-$th row resembles the first row shifted by $i-1$ places to the right.

QAPs with one Anti-Monge (Monge) matrix and one Toeplitz (circulant) matrix. In general these versions of the QAP remain NP-hard unless additional conditions, e.g. monotonicity, are imposed on the coefficient matrices. A well studied problem is the so called Anti-Monge–Toeplitz QAP where the rows and columns of the Anti-Monge matrix are non-decreasing, investigated by Burkard, Çela, Rote and Woeginger [28]. It has been shown that this problem is NP-hard and contains as a special case the so called turbine problem introduced by Mosewich [137] and formulated as a QAP by Laporte and Mercure [117]. In the turbine problem we are given a number of blades to be welded in regular spacing around the cylinder of the turbine. Due to inaccuracies in the manufacturing process the weights of the blades differ slightly and consequently the gravity center of the system does not lie on the rotation axis of the cylinder, leading to instabilities. In an effort to make the system as stable as possible, it is desirable to locate the blades so as to minimize the distance between the center of gravity and the rotation axis. The mathematical formulation of this problem leads to an NP-hard Anti-Monge–Toeplitz QAP. (For more details and for a proof of NP-hardness see Burkard et al. [28].) It is probably interesting that the *maximization* version of this problem is polynomially solvable. Further polynomially solvable special cases of the Anti-Monge–Toeplitz QAP arise if additional constraints e.g. *benevolence* or *k-benevolence* are imposed on the Toeplitz matrix. These conditions are expressed in terms of properties of the generating function of these matrices, see Burkard et al. [28].
The polynomially solvable QAPs with one Anti-Monge (Monge) matrix and the other one Toeplitz (circulant) matrix described above, are all constant permutation QAPs. The

techniques used to prove this fact and to identify the optimal permutation is called *reduction to extremal rays*. This technique exploits two facts: first, the involved matrix classes form cones, and secondly, the objective function of the QAP is linear with respect to each of the coefficient matrices. These two facts allow us to restrict the investigations to instances of the QAP with 0-1 coefficient matrices which are extremal rays of the above mentioned cones. Such instances can then be handled by elementary means (exchange arguments, bounding techniques) more easily that the general given QAP. The identification of polynomially solvable special cases of the QAP which are not constant permutation QAPs and can be solved algorithmically remains a challenging open question.

Another class of matrices similar to the Monge matrices are the *Kalmanson matrices*. A matrix $A = (a_{ij})$ is a Kalmanson matrix if it is symmetric and its elements satisfy the following inequalities for all indices i, j, k, l, $i < j < k < l$:

$$a_{ij} + a_{kl} \leq a_{ik} + a_{jl}, \qquad a_{il} + a_{jk} \leq a_{ik} + a_{jl}.$$

For more information on Monge, Anti-Monge and Kalmanson matrices, and their properties the reader is referred to the survey article of Burkard, Klinz and Rudolf [35]. The Koopmans-Beckmann QAP with one coefficient matrix being is a Kalmanson matrix and the other one a Toeplitz matrix, has been investigated by Deĭneko and Woeginger [57]. The computational complexity of this problem is an open question, but analogously as in the case of the Anti-Monge–Toeplitz QAP, polynomially solvable versions of the problem are obtained by imposing additional constraints to the Toeplitz matrix.

Further polynomially solvable cases arise as QAP formulations of other problems, like the linear arrangement problem, minimum feedback arc set problem, packing problems in graphs and subgraph isomorphism, see [26, 41]. Polynomially solvable versions of these problems lead to polynomially solvable cases of the QAP. The coefficient matrices of these QAPs are the (weighted) adjacency matrices of the underlying graphs, and the special structure of these matrices is imposed by properties of these graphs. The methods used to solve these QAPs range from graph theoretical algorithms (in the case of the linear arrangement problem and the feedback arc set problem), to dynamic programming (in the case of subgraph isomorphism).

11 QAP Instances with Known Optimal Solution

[QAPs with known solution] Since the QAP is a very hard problem from a practical point of view, often heuristics are the only reasonable approach to solve it, and so far there exists no performance guarantees for any of the algorithms developed for the QAP. One possibility to evaluate the performance of heuristics and to compare different heuristics is given by QAP instances with known optimal solution. Heuristics are applied to these instances and the heuristic solution is compared to the optimal one known before hand. The instances with known optimal solution should ideally have two properties: first, they should be representative in terms of their hardness, and secondly, they should not be especially easy for any of the heuristics.

Two *generators of QAP instances with known optimal solution* have been proposed so far: Palubeckis' generator [144] and the generator proposed by Li and Pardalos [122].

The first method for generating QAP instances with a known optimal solution was proposed by Palubeckis [144] in 1988. The input of the Palubeckis' algorithm consists of the size n of the instance to be generated, the optimal solution (permutation) π of the

output instance, two control parameters w and z, where $z < w$, and the distance matrix A of an $r \times s$ grid with $rs = n$. A contains *rectilinear distances* also called *Manhattan distances*, i.e., the distance a_{ij} between two given knots i, j lying in rows r_i, r_j and in columns c_i, c_j, respectively, is given by $a_{ij} = |r_i - r_j| + |c_i - c_j|$. The output of the algorithm is a second matrix B such that π is an optimal solution of $QAP(A, B)$. The idea is to start with a matrix B such that $QAP(A, B)$ is a trivial instance with optimal solution π. Then B is transformed such that $QAP(A, B)$ is not any more trivial, but π continues to be its optimal solution.
Palubeckis starts with a constant matrix $B = (b_{ij})$ with $b_{ij} = w$. $QAP(A, B)$ is a trivial problem because all permutations yield the same value of the objective function and thus, are optimal solutions. Hence, also the identity permutation id is an optimal solution of $QAP(A, B)$. Then matrix B is iteratively transformed so that it is not a constant matrix any more and the identity permutation remains an optimal solution of $QAP(A, B)$. In the last iteration the algorithm constructs an instance $QAP(A', B)$ with optimal solution π with the help of $QAP(A, B)$ with optimal solution id, by setting $A' = (a_{\pi(i)\pi(j)})$. The optimal value of $QAP(A', B)$ equals $w \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}$.

Cyganski, Vaz and Virball [55] have observed that the QAP instances generated by Palubeckis' generator are "easy" in the sense that their optimal value can be computed in polynomial time by solving a linear program. (For an accessible proof of this result the reader is referred to [41].) Notice, however, that nothing is known about the computational complexity of QAP instances generated by Palubeckis' generator. We believe that *finding an optimal solution* for these QAPs is NP-hard, although the corresponding decision problem is polynomially solvable.

Another generator of QAP instances with known solution has been proposed by Li and Pardalos [122]. As Palubeckis' generator, Li and Pardalos starts with a trivial instance $QAP(A, B)$ with the identity permutation id as optimal solution and iteratively transforms A and B so that the resulting QAP instance still has the optimal solution id but is not trivial any more. The transformations are such that for all i, j, i', j', $a_{ij} \geq a_{i'j'}$ is equivalent to $b_{ij} \leq b'_{ij}$ at the end of each iteration.

If the coefficient matrices are considered as weighted adjacency matrices of graphs, each iteration transforms entries corresponding to some specific subgraph equipped with signs on the edges and hence called sign-subgraphs. The application of Li and Pardalos' algorithm with different sign-subgraphs yields different QAP generators. A number of generators involving different sign-subgraphs, e.g. subgraphs consisting of a single edge, signed triangles and signed spanning trees have been tested. It is perhaps interesting and surprising that QAP instances generated by involving more complex sign-subgraphs are generally "easier" than those generated by involving subgraphs consisting of single edges. Here a QAP instance is considered to be "easy", if most heuristics applied to it find a solution near to the optimal one in a relatively short time. Nothing is known about the complexity of QAP instances generated by the generator of Li and Pardalos, since the arguments used to analyze Palubeckis' generator do not apply in this case.

12 Asymptotic Behavior

The QAP shows an interesting asymptotic behavior: under certain probabilistic conditions on the coefficient matrices the QAP, the ratio between its "best" and "worst" values of the objective function approaches 1, as the size of the problem approaches infinity. This

asymptotic behavior suggests that the relative error of every heuristic method vanishes as the size of the problem tends to infinity, i.e., every heuristic finds almost always an almost optimal solution when applied to QAP instances which are large enough. In other words the QAP becomes in some sense trivial as the size of the problem tends to infinity. Burkard and Fincke [32] identify a common combinatorial property of a number of problems which, under natural probabilistic conditions on the problem data, behave as described above. This property seems to be also the key for the specific asymptotic behavior of the QAP.

In an early work Burkard and Fincke [31] investigate the relative difference between the worst and the best value of the objective function for Koopmans-Beckmann QAPs. They first consider the case where the coefficient matrix D is the matrix of pairwise distances of points chosen independently and uniformly from the unit square in the plane. Then the general case where entries of the flow and distance matrices F and D are independent random variables taken from a uniform distribution on $[0, 1]$ is considered. In both cases it is shown that the relative difference mentioned above approaches 0 with probability tending to 1 as the size of the problem tends to infinity.
Later Burkard and Fincke [32] consider the ratio between the objective function values corresponding to an optimal (or best) and a worst solution of a generic combinatorial optimization problem described below.
Consider a sequence P_n, $n \in \mathbb{N}$, of combinatorial optimization (minimization) problems with sum objective function as described in Section 3. Let $\mathcal{E}_n$ and $\mathcal{F}_n$ be the ground set and the set of feasible solutions of problem P_n, respectively. Moreover, let $c_n \colon \mathcal{E}_n \to \mathbb{R}^+$ and $f \colon \mathcal{F} \to \mathbb{R}^+$ be the nonnegative cost function and the objective function for problem P_n, respectively, For $n \in \mathbb{N}$, an optimal solution X_{opt} minimizes the objective function, whereas a *worst solution* $X_{\text{wor}} \in \mathcal{F}_n$ maximizes the objective function and is defined as follows:

$$f(X_{\text{wor}}) = \sum_{x \in X_{\text{wor}}} c_n(x) = \max_{X \in \mathcal{F}} f(X) = \max_{X \in \mathcal{F}} \sum_{x \in X} c_n(x).$$

It is shown in [32] that the behavior of the ratio $f(X_{\text{opt}})/f(X_{\text{wor}})$ is strongly related to the ratio $\ln |\mathcal{F}_n|/|X_n|$ between the cardinality of the set of feasible solutions $\mathcal{F}_n$ and the cardinality of an arbitrary feasible solution X_n, under the assumption that all feasible solutions have the same cardinality.

Theorem 12.1 (Burkard and Fincke [32], 1985)
Let P_n be a sequence of combinatorial minimization problems with sum objective function as described above. Assume that the following conditions are fulfilled:

(BF1) *For all $X \in \mathcal{F}_n$, $|X| = |X^{(n)}|$, where $X^{(n)}$ is some feasible solution in $\mathcal{F}_n$.*

(BF2) *The costs $c_n(x)$, $x \in X$, $X \in \mathcal{F}_n$, $n \in \mathbb{N}$, are random variables identically distributed on $[0, 1]$. The expected value $E = E(c_n(x))$ and the variance $\sigma^2 = \sigma^2(c_n(x)) > 0$ of the common distribution are finite. Moreover, for all $X \in \mathcal{F}_n$, $n \in \mathbb{N}$, the variables $c_n(x)$, $x \in X$, are independently distributed.*

(BF3) *$|\mathcal{F}_n|$ and $|X^{(n)}|$ tend to infinity as n tends to infinity and moreover,*

$$\lim_{n \to \infty} \lambda_0 |X^{(n)}| - \ln |\mathcal{F}_n| \to +\infty$$

where λ_0 is defined by $\lambda_0 := (\epsilon_0 \sigma/(\epsilon_0 + 2\sigma^2))^2$ and ϵ_0 fulfills

$$0 < \epsilon_0 < \sigma^2 \quad \text{and} \quad 0 < \frac{E + \epsilon_0}{E - \epsilon_0} \leq 1 + \epsilon\,, \tag{57}$$

for a given $\epsilon > 0$.

Then, as $n \to \infty$

$$P\left\{\frac{\max\limits_{X\in\mathcal{F}_n}\sum\limits_{x\in X} c_n(x)}{\min\limits_{X\in\mathcal{F}_n}\sum\limits_{x\in X} c_n(x)} < 1+\epsilon\right\} \geq 1 - 2|\mathcal{F}_n|\exp(-|X^{(n)}|\lambda_0) \to 1\,.$$

The combinatorial condition represented by the limit in (BF3) says that the cardinality of the feasible solutions is large enough with respect to the cardinality of the set of feasible solutions. Namely, the result of the theorem is true if the following equality holds:

$$\lim_{n\to\infty}\frac{\ln|\mathcal{F}_n|}{|X^{(n)}|} = 0$$

The other conditions of Theorem 12.1 are natural probabilistic requirements on the coefficients of the problem. Theorem 12.1 states that for each $\epsilon > 0$, the ratio between the best and the worst value of the objective function lies on $(1-\epsilon, 1+\epsilon)$, *with probability tending to 1*, as the "size" of the problem approaches infinity. Thus, we have convergence *with probability*. Under one additional natural (combinatorial) assumption (condition (S3) of the theorem below), Szpankowski strengthens this result and improves the range of the convergence to *almost surely*. In the almost sure convergence the probability that the above mentioned ratio tends to 1 is equal to 1. (Detailed explanations on the probabilistic notions used in every text book on probability theory.)

Theorem 12.2 (Szpankowski [170], 1995)
Let P_n *be a sequence of combinatorial minimization problems with sum objective function as above. Assume that the following conditions are fulfilled:*

(S1) *For all* $X \in \mathcal{F}_n$, $|X| = |X^{(n)}|$, *where* $X^{(n)}$ *is some feasible solution in* $\mathcal{F}_n$.

(S2) *The costs* $c_n(x)$, $x \in X$, $X \in \mathcal{F}_n$, $n \in \mathbb{N}$, *are random variables identically and independently distributed on* $[0,1]$. *The expected value* $E = E(c_n(x))$, *the variance, and the third moment of the common distribution are finite.*

(S3) *The worst values of the objective function,* $\max\limits_{X\in\mathcal{F}_n}\sum\limits_{x\in X} c_n(x)$, *form a nondecreasing sequence for increasing* n.

(S4) $|\mathcal{F}_n|$ *and* $|X^{(n)}|$ *tend to infinity as* n *tends to infinity and moreover,* $\ln|\mathcal{F}_n| = o(|X^{(n)}|)$.

Then, the following equalities hold almost surely:

$$\min_{X\in\mathcal{F}_n}\sum_{x\in X} c_n(x) = |X^{(n)}|E - o(|X^{(n)}|)$$

$$\max_{X\in\mathcal{F}_n}\sum_{x\in X} c_n(x) = |X^{(n)}|E + o(|X^{(n)}|)$$

Theorems 12.1 and 12.2 can be applied to the QAP. The reason is that the QAP fulfills the combinatorial condition (S4) in Theorem 12.2 (and therefore, also condition (BF3) in Theorem 12.1). Thus, we immediately get the following corollary:

Corollary 12.3 *Consider a sequence of problems $QAP(A^{(n)}, B^{(n)})$ for $n \in \mathbb{N}$, with $n \times n$ coefficient matrices $A^{(n)} = \left(a_{ij}^{(n)}\right)$ and $B = \left(b_{ij}^{(n)}\right)$. Assume that $a_{ij}^{(n)}$ and $b_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, are independently distributed random variables on $[0, M]$, where M is a positive constant. Moreover, assume that entries $a_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, have the same distribution, and entries $b_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, have also the same distribution (which does not necessarily coincide with that of $a_{ij}^{(n)}$). Furthermore, assume that these variables have finite expected values, variances and third moments.*
Let $\pi_{opt}^{(n)}$ and $\pi_{wor}^{(n)}$ denote an optimal and a worst solution of $QAP(A^{(n)}, B^{(n)})$, respectively, i.e.,

$$Z\left(A^{(n)}, B^{(n)}, \pi_{opt}^{(n)}\right) = \min_{\pi \in \mathcal{S}_n} Z\left(A^{(n)}, B^{(n)}, \pi\right)$$

and

$$Z\left(A^{(n)}, B^{(n)}, \pi_{wor}^{(n)}\right) = \max_{\pi \in \mathcal{S}_n} Z\left(A^{(n)}, B^{(n)}, \pi\right)$$

Then the following equality holds almost surely:

$$\lim_{n \to \infty} \frac{Z\left(A^{(n)}, B^{(n)}, \pi_{opt}^{(n)}\right)}{Z\left(A^{(n)}, B^{(n)}, \pi_{wor}^{(n)}\right)} = 1$$

The above result suggests that the value of the objective function of $QAP(A^{(n)}, B^{(n)})$ (corresponding to an arbitrary feasible solution) gets somehow close to its expected value $n^2 E(A)E(B)$, as the size of the problem increases, where $E(A)$ and $E(B)$ are the expected values of $a_{ij}^{(n)}$ and $b_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, respectively. Frenk, Houweninge, and Rinnooy Kan [69] and Rhee [159, 160] provide different analytical evaluations for this "getting close", by imposing different probabilistic conditions on the data. The following theorem states two important results proved in [69] and [160].

Theorem 12.4 (Frenk et al. [69], 1986, Rhee [160], 1991)
Consider the sequence of $QAP(A^{(n)}, B^{(n)})$, $n \in \mathbb{N}$, as in Corollary 12.3. Assume that the following conditions are fulfilled:

(C1) *$a_{ij}^{(n)}$, $b_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, are random variables independently distributed on $[0, M]$.*

(C2) *$a_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, have the same distribution on $[0, M]$. $b_{ij}^{(n)}$, $n \in \mathbb{N}$, $1 \leq i, j \leq n$, have also the same distribution on $[0, M]$.*

Let $E(A)$, $E(B)$ be the expected values of the variables $a_{ij}^{(n)}$ and $b_{ij}^{(n)}$, respectively. Then, there exists a constant K_1 (which does not depend on n), such that the following inequality holds almost surely, for $\pi \in \mathcal{S}_n$, $n \in \mathbb{N}$

$$\limsup_{n \to \infty} \frac{\sqrt{n}}{\sqrt{\log n}} \left| \frac{Z(A^{(n)}, B^{(n)}, \pi)}{n^2 E(A)E(B)} - 1 \right| \leq K_1$$

Moreover, let Y be a random variable defined by

$$Y = Z\left(A^{(n)}, B^{(n)}, \pi_{opt}^{(n)}\right) - n^2 E(A)E(B)\,,$$

where $\pi_{opt}^{(n)}$ is an optimal solution of $QAP(A^{(n)}, B^{(n)})$. Then there exists another constant K_2, also independent of the size of the problem, such that

$$\frac{1}{K_2} n^{3/2} (\log n)^{1/2} \leq E(Y) \leq K_2 n^{3/2} (\log n)^{1/2}$$

$$P\{|Y - E(Y)| \geq t\} \leq 2 \exp\left(\frac{-t^2}{4n^2 \|A\|_\infty^2 \|B\|_\infty^2}\right)$$

for each $t \geq 0$, where $E(Y)$ denotes the expected value of variable Y and $\|A\|_\infty$ ($\|B\|_\infty$) is the so-called row sum norm *of matrix A (B) defined by $\|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|$.*

These results on the asymptotic behavior of the QAP have been exploited by Dyer, Frieze, and McDiarmid [60] to analyze the performance of branch and bound algorithms for QAPs with coefficients generated randomly as described above. Dyer et al. have shown that for such QAPs the optimal value of the continuous relaxation of Frieze and Yadegar's linearization (17)-(24) is in $O(n)$ with probability tending to 1 as the size n of the QAP tends to infinity. Hence the gap between the optimal value of this continuous relaxation and the optimal value of the QAP grows like $O(n)$ with probability tending to 1 as n tends to infinity. This result leads to the following theorem.

Theorem 12.5 (Dyer, Frieze, and McDiarmid [60], 1986)
Consider any branch and bound algorithm for solving a QAP with randomly generated coefficients as in Corollary 12.3, that uses single assignment branching and employs a bound obtained by solving the continuous relaxation of the linearization (17)-(24). The number of branched nodes explored is at least $n^{(1-o(1))n/4}$ with probability tending to 1 as the size n of the QAP tends to infinity.

13 Related Problems

One possibility to obtain generalizations of the QAP is to consider objective functions of higher degree and obtain in this way *cubic*, *biquadratic* and generally *N-adic* assignment problems (see e.g. [118]). For the cubic assignment problem for example, we have n^6 cost coefficients c_{ijklmp} where $i, j, k, l, m, p = 1, \ldots, n$, and the problem is given as follows:

$$\begin{aligned} \min \quad & \sum_{i,j=1}^{n} \sum_{k,l=1}^{n} \sum_{m,p=1}^{n} c_{ijklmp} x_{ij} x_{kl} x_{mp} \\ \text{s.t} \quad & (x_{ij}) \in \mathbf{X}_n \,. \end{aligned}$$

As it is noted in [118], we can construct an $n^3 \times n^3$ matrix S containing the cost coefficients, such that the cubic assignment problem is equivalent to the LAP

$$\begin{aligned} \min \quad & \langle S, Y \rangle \\ \text{s.t.} \quad & Y = X \otimes X \otimes X, \\ & X \in \mathbf{X}_n. \end{aligned}$$

In an analogous way the LAP can be extended to any N-adic assignment problem, by considering the solution matrix Y to be the Kronecker N^{th} power of a permutation matrix in $\mathbf{X}_n$.

Another modification of the objective function which yields a problem related to the QAP, the *bottleneck quadratic assignment problem* (BQAP), is the substitution of the sum by a *max* operation. The first occurrence of the BQAP is due to Steinberg [168] and arises as an application in backboard wiring while trying to minimize the maximum length of the involved wires (see also Section 1).

In this section several of generalizations and problems related to the QAP are presented, for which real applications have been found that initiated an interest in analyzing them and proposing solution techniques.

13.1 The Bottleneck QAP

In the bottleneck quadratic assignment problem (BQAP) of size n we are given an $n \times n$ flow matrix F and an $n \times n$ distance matrix D, and wish to find a permutation $\phi \in \mathcal{S}_n$ which minimizes the objective function

$$\max\{f_{ij}d_{\phi(i)\phi(j)} : 1 \leq i, j \leq n\}.$$

A more general BQAP analogous to the QAP in (2) is obtained if the coefficients of the problem are of the form c_{ijkl}, $1 \leq i, j, k, l \leq n$:

$$\min_{\phi \in \mathcal{S}_n} \max_{1 \leq i,j \leq n} c_{ij\phi(i)\phi(j)}.$$

Besides the application in backboard wiring mentioned above, the BQAP has many other applications. Basically, all QAP applications give raise to applications of the BQAP because it often makes sense to minimize the largest cost instead of the overall cost incurred by some decision. A well studied problem in graph theory which can be modeled as a BQAP is the *bandwidth problem*. In the bandwidth problem we are given an undirected graph $G = (V, E)$ with vertex set V and edge set E, and seek a labeling of the vertices of G by the numbers $1, 2, \ldots, n$, where $|V| = n$, such that the minimum absolute value of differences of labels of vertices which are connected by an edge is minimized. In other words, we seek a labeling of vertices such that the maximum distance of 1-entries of the resulting adjacency matrix from the diagonal is minimized, i.e., the bandwidth of the adjacency matrix is minimized. It is easy to see that this problem can be modeled as a special BQAP with flow matrix equal to the adjacency matrix of G for some arbitrary labeling of vertices, and distance matrix $D = (|i - j|)$.

The BQAP is NP-hard since it contains the bottleneck TSP as a special case. (This is analogous to the fact that the QAP contains the TSP as a special case, as it is shown in Section 13.4). Some enumeration algorithms to solve BQAP to optimality have been proposed by Burkard [22]. These algorithms employ a Gilmore-Lawler-like bound for the BQAP which involves in turn the solution of bottleneck linear assignment problems. The algorithm for the general BQAP involves also a threshold procedure useful to reduce to 0 as many coefficients as possible.

Burkard and Fincke [30] investigated the asymptotic behavior of the BQAP and proved results analogous to those obtained for the QAP: If the coefficients are independent random variables taken from a uniform distribution on $[0, 1]$, then the relative difference between the worst and the best value of the objective function approaches 0 with probability tending to 0 as the size of the problem approaches infinity.

The BQAP and the QAP are special cases of a more general quadratic assignment problem which can be called *the algebraic QAP* (in analogy to the algebraic linear assignment problem (LAP) introduced by Burkard, Hahn, and Zimmermann [33]). If $(H, *, \prec)$ is a totally

ordered commutative semigroup with composition $*$ and order relation $\prec$, the algebraic QAP with cost coefficients $c_{ijkl} \in H$ is formulated as

$$\min_{\phi \in \mathcal{S}_n} c_{11\phi(1)\phi(1)} * c_{12\phi(1)\phi(2)} * \ldots * c_{1n\phi(1)\phi(n)} * \ldots * c_{nn\phi(n)\phi(n)} \,.$$

The study of the bottleneck QAP and more generally the algebraic QAP was the starting point for the investigation of a number of algebraic combinatorial optimization problem with coefficients taken from linearly ordered semimodules e.g. linear assignment and transportation problems, flow problems etc. The reader is referred to Burkard and Zimmermann [38] for a detailed discussion on this topic.

13.2 The BiQuadratic Assignment Problem

A generalization of the QAP is the BiQuadratic Assignment Problem, denoted BiQAP, which is essentially a quartic assignment problem with cost coefficients formed by the products of two four-dimensional arrays. More specifically, consider two $n^4 \times n^4$ arrays $F = (f_{ijkl})$ and $D = (d_{mpst})$. The BiQAP can then be stated as:

$$\begin{aligned}
\min \quad & \sum_{i,j=1}^{n} \sum_{k,l=1}^{n} \sum_{m,p=1}^{n} \sum_{s,t=1}^{n} f_{ijkl} d_{mpst} x_{im} x_{jp} x_{ks} x_{lt} \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n, \\
& \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n, \\
& x_{ij} \in \{0, 1\}, \quad i, j = 1, 2, \ldots, n.
\end{aligned}$$

The major application of the BiQAP arises in Very Large Scale Integrated (VLSI) circuit design. The majority of VLSI circuits are sequential circuits, and their design process consists of two steps: first, translate the circuit specifications into a state transition table by modeling the system using finite state machines, and secondly, try to find an encoding of the states such that the actual implementation is of minimum size. A detailed description of the mathematical modeling of the VLSI problem to a BiQAP is given by Burkard, Çela and Klinz [27]. Equivalently, the BiQAP can be stated as:

$$\min_{\phi \in \mathcal{S}_n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} f_{ijkl} d_{\phi(i)\phi(j)\phi(k)\phi(l)},$$

where $\mathcal{S}_n$ denotes the set of all permutations of $N = \{1, 2, \ldots, n\}$. All different formulations for the QAP can be extended to the BiQAP, as well as most of the linearizations that have appeared for the QAP.

Burkard et al. [27] compute lower bounds for the BiQAP derived from lower bounds of the QAP. The computational results showed that these bounds are weak and deteriorate as the dimension of the problem increases. This observation suggests that branch and bound methods will only be effective on very small instances. For larger instances, efficient heuristics, that find good-quality approximate solutions, are needed. Several heuristics for the BiQAP have been developed by Burkard and Çela [25], in particular deterministic improvement methods and variants of simulated annealing and tabu search algorithms.

Computational experiments on test problems of size up to $n = 32$, with known optimal solutions (a test problem generator is presented in [27]), suggest that one version of simulated annealing is best among those tested. The GRASP heuristic has also been applied to the BiQAP by Mavridou, Pardalos, Pitsoulis and Resende [132], and produced the optimal solution for all the test problems generated in [27].

13.3 The Quadratic Semi-Assignment Problem

In the quadratic semi-assignment problem (QSAP) we are given again two coefficient matrices, a flow matrix $F = (f_{ij})$ and a distance matrix $D = (d_{ij})$, but in this case we have n "objects" and m "locations", $n > m$. We want to assing all objects to locations and at least one object to each location so as to minimize the overall distance covered by the flow of materials (or people) moving between different objects. Thus the objective function is the same as that of the QAP, and the only different concerns the feasible solutions which are not one-to-one mappings (or bijections) between the set of objects and locations but arbitrary functions mapping the set of objects to the set of locations. Thus SQAP can be formulated as follows:

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}x_{ik}x_{jl} + \sum_{i,j=1}^{n} b_{ij}x_{ij} \tag{58}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n, \tag{59}$$

$$x_{ij} \in \{0, 1\}, \quad i, j = 1, 2, \ldots, n. \tag{60}$$

SQAP unifies some interesting combinatorial optimization problems like *clustering*, *m-coloring*. In a clustering problem we are given n objects and a dissimilarity matrix $F = (f_{ij})$. The goal is to find a partition of these objects into m classes so as to minimize the sum of dissimilarities of objects belonging to the same class. Obviously this problem is a QSAP with coefficient matrices F and D, where D is an $m \times m$ identity matrix. In the m-coloring problem we are given a graph with n vertices and want to check whether its vertices can be colored by m different colors such that each two vertices which are joined by an edge receive different colors. This problem can be modeled as a SQAP with F equal to the adjacency matrix of the given graph and D the $m \times m$ identity matrix. The m-coloring has an answer "yes" if and only if the above SQAP has optimal value equal to 0. Practical applications of the SQAP include distributed computing [169] and scheduling [45].

SQAP was originally introduced by Greenberg [85]. As pointed out by Malucelli [128] this problem is NP-hard. Milis and Magirou [134] propose a Lagrangean relaxation algorithm for this problem, and show that similarly as for for the QAP, it is very hard to provide optimal solutions even for SQAPs of small size. Lower bounds for the SQAP have been provided by Malucelli and Pretolani [129], and polynomially solvable special cases have been discussed by Malucelli [128].

13.4 Other Problems Which Can Be Formulated as QAPs

There are a number of other well known combinatorial optimization problems which can be formulated as QAPs with specific coefficient matrices. Of course, since QAP is not a well tractable problem, it does not make sense to use algorithms developed for the QAP to solve these other problems. All known solution methods for the QAP are far inferior

compared to any of the specialized algorithms developed for solving these problems. However, the relationship between the QAP and these problems might be of benefit for a better understanding of the QAP and its inherent complexity.

Graph Partitioning and Maximum Clique

Two well studied NP-hard combinatorial optimization problems which are special cases of the QAP, are the *graph partitioning problem* (GPP) and the *maximum clique problem* (MCP). In GP we are given an (edge) weighted graph $G = (V, E)$ with n vertices and a number k which divides n. We want to partition the set V into k sets of equal cardinality such that the total weight of the edges cut by the partition is minimized. This problem can be formulated as a QAP with distance matrix D equal to the weighted adjacency matrix of G, and flow matrix F obtained by multiplying with -1 the adjacency matrix of the union of k disjoint complete subgraphs with n/k vertices each. For more informations on graph partitioning problems the reader is referred to Lengauer [120].

In the maximum clique problem we are again given a graph $G = (V, E)$ with n vertices and wish to find the maximum $k \leq n$ such that there exists a subset $V_1 \subseteq V$ which induces a clique in G, i.e. all vertices of V_1 are connected by edges of G. In this case consider a QAP with distance matrix D equal to the adjacency matrix of G and flow matrix F given as adjacency matrix of a graph consisting of a clique of size k and $n - k$ isolated vertices, multiplied by -1. A clique of size k in G exists only if the optimal value of the corresponding QAP is $-k^2$. For a review on the maximum clique problem the reader is referred to [151].

The Traveling Salesman Problem

In the *traveling salesman problem* (TSP) we are given a set of cities and the pairwise distances between them, and our task is to find a tour which visits each city exactly once and has minimal length. Let the set of integers $N = \{1, 2, \ldots, n\}$ represent the n cities and let the symmetric $n \times n$ matrix $D = (d_{ij})$ represent the distances between the cities, where d_{ij} is the distance between city i and city j ($d_{ii} = 0 \ \forall i = 1, 2, \ldots, n$). The TSP can be formulated as

$$\begin{aligned} \min \quad & \sum_{i=1}^{n-1} d_{\phi(i)\phi(i+1)} + d_{\phi(n)\phi(1)} \\ \text{s.t.} \quad & \phi \in \mathcal{S}_n. \end{aligned} \tag{61}$$

The TSP can be formulated as a QAP with the given distance matrix and a flow matrix F being equal to the adjacency matrix of a cycle on n vertices.

The traveling salesman problem (TSP) is a notorious NP-hard combinatorial optimization problem. Among the abounding literature on the TSP we select the book edited by Lawler, Lenstra, Rinnooy Kan and Schmoys [119] as a comprehensive reference.

The linear arrangement problem

In the *linear arrangement problem* we are given a graph $G = (V, E)$ and wish to place its vertices at the points 1, 2, ..., n on the line so as to minimize the sum of pairwise distances between vertices of G which are joined by some edge. If we consider the more general version of weighted graphs than we obtain the backboard wiring problem (see Section 1).

This is an NP-hard problem as mentioned by Garey and Johnson [73]. It can be formulated as a QAP with distance matrix the (weighted) adjacency matrix of the given graph, and flow matrix $F = (f_{ij})$ given by $f_{ij} = |i - j|$, for all i, j.

The minimum weight feedback arc set problem

In the *minimum weight feedback arc set problem* (FASP) a weighted digraph $G = (V, E)$ with vertex set V and arc set E is given. The goal is to remove a set of arcs from E with minimum overall weight, such that all directed cycles, so-called *dicycles*, in G are destroyed and an acyclic directed subgraph remains. Clearly, the minimum weight feedback arc set problem is equivalent to the problem of finding an acyclic subgraph of G with maximum weight. The unweighted version of the FASP, that is a FASP where the edge weights of the underlying digraph equal 0 or 1, is called *the acyclic subdigraph problem* and is treated extensively by Jünger [99].
An interesting application of the FASP is the so-called *triangulation of input-output tables* which arises along with input-output analysis in economics used to forecast the development of industries, see Leontief [121]. For details and a concrete description of the application of triangulation results in economics the reader is referred to Conrad [54] and Reinelt [153].

Since the vertices of an acyclic subdigraph can be labeled topologically, i.e. such that in each arc the label of its head is larger than that of its tail, the FASP can be formulated as a QAP. The distance matrix of the QAP is the weighted adjacency matrix of G and the flow matrix $F = (f_{ij})$ is a *lower triangular matrix*, i.e., $f_{ij} = -1$ if $i \leq j$ and $f_{ij} = 0$, otherwise.

The FASP is well known to be NP-hard (see Karp [107], Garey and Johnson [73]).

Packing problems in graphs

Another well known NP-hard problem which can be formulated as a QAP is the *graph packing problem* cf. Bollobás [18]. In a graph packing problem we are given graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ with n vertices each and edge sets E_1 and E_2. A permutation π of $\{1, 2, \ldots, n\}$ is called a *packing of* G_2 *into* G_1, if $(i, j) \in E_1$ implies $(\pi(i), \pi(j)) \notin E_2$, for $1 \leq i, j \leq n$. In other words, a packing of G_2 into G_1 is an embedding of the vertices of G_2 into the vertices of G_1 such that no pair of edges coincide. The *graph packing problem* consists of finding a packing of G_2 into G_1, if one exists, or proving that no packing exists.

The graph packing problem can be formulated as a QAP with distance matrix equal to the adjacency matrix of G_2 and flow matrix equal to the adjacency matrix of G_1. A packing of G_2 into G_1 exists if and only if the optimal value of this QAP is equal to 0. In the positive case the optimal solution of the QAP determines a packing.

References

[1] E. H. L. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley, Chichester, 1989.

[2] E. Aarts and J. K. Lenstra, eds., *Local Search in Combinatorial Optimization*, Wiley, Chichester, 1997.

[3] W. P. Adams and T. A. Johnson, Improved linear programming-based lower bounds for the quadratic assignment problem, in *Quadratic Assignment and Related Problems*, P. M. Pardalos and H. Wolkowicz, eds., *DIMACS Series on Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 43–75, AMS, Providence, RI.

[4] W. P. Adams and H. D. Sherali, A tight linearization and an algorithm for zero-one quadratic programming problems, *Management Science* **32**, 1986, 1274–1290.

[5] W. P. Adams and H. D. Sherali, Linearization strategies for a class of zero-one mixed integer programming problems, *Operations Research* **38**, 1990, 217–226.

[6] R. K. Ahuja, J. B. Orlin, and A. Tivari, A greedy genetic algorithm for the quadratic assignment problem, Working paper 3826-95, Sloan School of Management, MIT, 1995.

[7] S. Arora, A. Frieze, and H. Kaplan, A new rounding procedure for the assignment problem with applications to dense graph arrangement problems, *Proceedings of the 37-th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1996, 21–30.

[8] A. A. Assad, and W. Xu, On lower bounds for a class of quadratic 0-1 programs, *Operations Research Letters* **4**, 1985, 175–180.

[9] E. Balas and J. B. Mazzola, Quadratic 0-1 programming by a new linearization, presented at *the Joint ORSA/TIMS National Meeting*, 1980, Washington D.C.

[10] E. Balas and J. B. Mazzola, Nonlinear programming: I. Linearization techniques, *Mathematical Programming* **30**, 1984, 1–21.

[11] E. Balas and J. B. Mazzola, Nonlinear programming: II. Dominance relations and algorithms, *Mathematical Programming* **30**, 1984, 22–45.

[12] A. I. Barvinok, Computational complexity of orbits in representations of symmetric groups, *Advances in Soviet Mathematics* **9**, 1992, 161–182.

[13] R. Battiti and G. Tecchiolli, The reactive tabu search, *ORSA Journal on Computing* **6**, 1994, 126–140.

[14] M. S. Bazaraa and O. Kirca, Branch and bound based heuristics for solving the quadratic assignment problem, *Naval Research Logistics Quarterly* **30**, 1983, 287–304.

[15] M. S. Bazaraa and H. D. Sherali, Benders' partitioning scheme applied to a new formulation of the quadratic assignment problem, *Naval Research Logistics Quarterly* **27**, 1980, 29–41.

[16] M. S. Bazaraa and H. D. Sherali, On the use of exact and heuristic cutting plane methods for the quadratic assignment problem, *Journal of Operations Research Society* **33**, 1982, 991–1003.

[17] G. Birkhoff, Tres observaciones sobre el algebra lineal, *Univ. Nac. Tucumán Rev., Ser. A*, 1946, 147–151.

[18] B. Bollobás, *Extremal Graph Theory*, Academic Press, London, 1978.

[19] A. Bruengger, J. Clausen, A. Marzetta, and M. Perregaard, Joining forces in solving large-scale quadratic assignment problems in parallel, in *Proceedings of the* 11*-th IEEE International Parallel Processing Symposium (IPPS)*, 1997, 418–427.

[20] E. S. Buffa, G. C. Armour, and T. E. Vollmann, Allocating facilities with CRAFT, *Harvard Business Review* **42**, 1962, 136–158.

[21] R. E. Burkard, Die Störungsmethode zur Lösung quadratischer Zuordnungsprobleme, *Operations Research Verfahren* **16**, 1973, 84–108.

[22] R. E. Burkard, Quadratische Bottleneckprobleme, *Operations Research Verfahren* **18**, 1974, 26–41.

[23] R. E. Burkard, Locations with spatial interactions: the quadratic assignment problem, in *Discrete Location Theory*, P. B. Mirchandani and R. L. Francis, eds., Wiley, 1991.

[24] R. E. Burkard and T. Bönniger, A heuristic for quadratic boolean programs with applications to quadratic assignment problems, *European Journal of Operational Research* **13**, 1983, 374–386.

[25] R. E. Burkard and E. Çela, Heuristics for biquadratic assignment problems and their computational comparison, *European Journal of Operational Research* **83**, 1995, 283–300.

[26] R. E. Burkard, E. Çela, V. M. Demidenko, N. N. Metelski, and G. J. Woeginger, Perspectives of Easy and Hard Cases of the Quadratic Assignment Problems, SFB Report 104, Institute of Mathematics, Technical University Graz, Austria, 1997.

[27] R. E. Burkard, E. Çela, and B. Klinz, On the biquadratic assignment problem, in *Quadratic Assignment and Related Problems*, P. M. Pardalos and H. Wolkowicz, eds., *DIMACS Series on Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 117–146, AMS, Providence, RI.

[28] R. E. Burkard, E. Çela, G. Rote, and G. J. Woeginger, The quadratic assignment problem with an Anti-Monge and a Toeplitz matrix: Easy and hard cases, SFB Report 34, Institute of Mathematics, Technical University Graz, Austria, 1995. To appear in *Mathematical Programming*.

[29] R. E. Burkard and U. Derigs, *Assignment and matching problems: Solution methods with Fortran programs*, *Lecture Notes in Economics and Mathematical Systems* **184**, Springer-Verlag, Berlin, 1980.

[30] R. E. Burkard and U. Fincke, On random quadratic bottleneck assignment problems, *Mathematical Programming* **23**, 1982, 227–232.

[31] R. E. Burkard and U. Fincke, The asymptotic probabilistic behavior of the quadratic sum assignment problem, *Zeitschrift für Operations Research* **27**, 1983, 73–81.

[32] R. E. Burkard and U. Fincke, Probabilistic asymptotic properties of some combinatorial optimization problems, *Discrete Applied Mathematics* **12**, 1985, 21–29.

[33] R. E. Burkard, W. Hahn and U. Zimmermann, An algebraic approach to assignment problems, *Mathematical Programming* **12**, 1977, 318–327.

[34] R. E. Burkard, S. E. Karisch, and F. Rendl, QAPLIB - a quadratic assignment problem library, *Journal of Global Optimization* **10**, 1997, 391–403. An on-line version is available via World Wide Web at the following URL: `http://www.opt.math.tu-graz.ac.at/~karisch/qaplib/`

[35] R. E. Burkard, B. Klinz, and R. Rudolf, Perspectives of Monge properties in optimization, *Discrete Applied Mathematics* **70**, 1996, 95–161.

[36] R. E. Burkard and J. Offermann, Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme, *Zeitschrift für Operations Research* **21**, 1977, B121–B132, (in German).

[37] R. E. Burkard and F. Rendl, A thermodynamically motivated simulation procedure for combinatorial optimization problems, *European Journal Operational Research* **17**, 1984, 169–174.

[38] R. E. Burkard and U. Zimmermann, Combinatorial optimization in linearly ordered semimodules: a survey, in *Modern Applied Mathematics*, B. Korte, ed., North Holland, Amsterdam, 1982, 392–436.

[39] P. Carraresi and F. Malucelli, A new lower bound for the quadratic assignment problem, *Operations Research* **40**, 1992, Suppl. No. **1**, S22–S27.

[40] P. Carraresi and F. Malucelli, A reformulation scheme and new lower bounds for the QAP, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 147–160, AMS, Providence, RI.

[41] E. Çela, *The Quadratic Assignment Problem: Theory and Algorithms*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[42] V. Černý, Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm, *Journal of Optimization Theory and Applications* **45**, 1985, 41–51.

[43] J. Chakrapani and J. Skorin-Kapov, Massively parallel tabu search for the quadratic assignment problem, *Annals of Operations Research* **41**, 1993, 327–342.

[44] J. Chakrapani and J. Skorin-Kapov, A constructive method to improve lower bounds for the quadratic assignment problem, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 161–171, AMS, Providence, RI.

[45] P. Chretienne, A polynomial algorithm to optimally schedule tasks on a virtual distributed system under tree-like precedence constraints, *European Journal of Operational Research* **43**, 1989, 225–230.

[46] N. Christofides, Worst case analysis of a new heuristic for the traveling salesman problem, Technical Report 338, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976.

[47] N. Christofides and E. Benavent, An exact algorithm for the quadratic assignment problem, *Operations Research* **37**, 1989, 760–768.

[48] N. Christofides and M. Gerrard, A graph theoretic analysis of bounds for the quadratic assignment problem, in *Studies on Graphs and Discrete Programming*, P.Hansen, ed., North Holland, 1981, pp. 61–68.

[49] J. Clausen, S. E. Karisch, M. Perregaard, and F. Rendl, On the applicability of lower bounds for solving rectilinear quadratic assignment problems in parallel, *Computational Optimization and Applications* **10**, 1998, 127–147.

[50] J. Clausen and M. Perregaard, Solving large quadratic assignment problems in parallel, *Computational Optimization and Applications* **8**, 1997, 111–127.

[51] A. Colorni, M. Dorigo, and V. Maniezzo, The ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics -Part B* **26**, 1996, 29–41.

[52] A. Colorni and V. Maniezzo, The ant system applied to the quadratic assignment problem, to appear in *IEEE Transactions on Knowledge and Data Engineering*, 1998.

[53] D. T. Connolly, An improved annealing scheme for the QAP, *European Journal of Operational Research* **46**, 1990, 93–100.

[54] K. Conrad, *Das Quadratische Zuweisungsproblem und zwei seiner Spezialfälle*, Mohr-Siebeck, Tübingen, 1971.

[55] D. Cyganski, R. F. Vaz, and V. G. Virball, Quadratic assignment problems with the Palubeckis' algorithm are degenerate, *IEEE Transactions on Circuits and Systems-I* **41**, 1994, 481–484.

[56] L. Davis, *Genetic Algorithms and Simulated Annealing*, Pitman, London, 1987.

[57] V. G. Deĭneko and G. J. Woeginger, A solvable case of the quadratic assignment problem, SFB Report 88, Institute of Mathematics, Technical University Graz, Austria, 1996.

[58] J. W. Dickey and J. W. Hopkins, Campus building arrangement using TOPAZ, *Transportation Research* **6**, 1972, 59–68.

[59] M. Dorigo, *Optimization, Learning, and Natural algorithms*, Ph.D. Thesis, Dipartimento die Elettronica e Informazione, Politecnico di Milano, Milano, Italy, 1992, (in Italian).

[60] M. E. Dyer, A. M. Frieze, and C. J. H. McDiarmid, On linear programs with random costs, *Mathematical Programming* **35**, 1986, 3–16.

[61] C. S. Edwards, The derivation of a greedy approximator for the Koopmans-Beckmann quadratic assignment problem, *Proceedings of the 77-th Combinatorial Programming Conference (CP77)*, 1977, 55–86.

[62] C. S. Edwards, A branch and bound algorithm for the Koopmans-Beckmann quadratic assignment problem, *Mathematical Programming Study* **13**, 1980, 35–52.

[63] A. N. Elshafei, Hospital layout as a quadratic assignment problem, *Operations Research Quarterly* **28**, 1977, 167–179.

[64] T. Espersen, S. E. Karisch, E. Çela, and J. Clausen, `QAPPACK`- a JAVA package for solving quadratic assignment problems, working paper, Department of Mathematical Modelling, Technical University of Denmark, Denmark, and Institute of Mathematics, Technical University Graz, Austria.

[65] T. A. Feo, M. G. C. Resende, and S. H. Smith, *A greedy randomized adaptive search procedure for the maximum independent set*, Technical report, AT&T Bell Laboratories, Murray Hill, NJ, 1989. To appear in *Operations Research*.

[66] T. A. Feo and M. G. C. Resende, Greedy randomized adaptive search procedures, *Journal of Global Optimization* **6**, 1995, 109–133.

[67] G. Finke, R. E. Burkard, and F. Rendl, Quadratic assignment problems, *Annals of Discrete Mathematics* **31**, 1987, 61–82.

[68] C. Fleurent and J. Ferland, Genetic hybrids for the quadratic assignment problem, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 173–187, AMS, Providence, RI.

[69] J. B. G. Frenk, M. van Houweninge, and A. H. G. Rinnooy Kan, Asymptotic properties of the quadratic assignment problem, *Mathematics of Operations Research* **10**, 1985, 100–116.

[70] A. M. Frieze and J. Yadegar, On the quadratic assignment problem, *Discrete Applied Mathematics* **5**, 1983, 89–98.

[71] A. M. Frieze, J. Yadegar, S. El-Horbaty, and D. Parkinson, Algorithms for assignment problems on an array processor, *Parallel Computing* **11**, 1989, 151–162.

[72] L. M. Gambardella, E. D. Taillard, and M. Dorigo, Ant colonies for the QAP, Technical Report IDSIA-4-97, 1997, Istituto dalle Molle Di Studi sull' Intelligenza Artificiale, Lugano, Switzerland.

[73] M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W. H. Freeman and Company, New York, 1979.

[74] J. W. Gavett and N. V. Plyter, The optimal assignment of facilities to locations by branch and bound, *Operations Research* **14**, 1966, 210–232.

[75] A. M. Geoffrion, Lagrangean relaxation and its uses in integer programming, *Mathematical Programming Study* **2**, 1974, 82–114.

[76] A. M. Geoffrion and G. W. Graves, Scheduling parallel production lines with changeover costs: Practical applications of a quadratic assignment/LP approach. *Operations Research* **24**, 1976, 595–610.

[77] P. C. Gilmore, Optimal and suboptimal algorithms for the quadratic assignment problem, *SIAM Journal on Applied Mathematics* **10**, 1962, 305–313.

[78] F. Glover, Improved linear integer programming formulations of nonlinear integer problems, *Management Science* **22**, 1975, 455–460.

[79] F. Glover, Tabu search - Part I, *ORSA Journal on Computing* **1**, 1989, 190–206.

[80] F. Glover, Tabu search - Part II, *ORSA Journal on Computing* **2**, 1989, 4–32.

[81] F. Glover, M. Laguna, E. Taillard, and D. de Werra, eds., Tabu search, *Annals of Operations Research* **41**, 1993.

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

[83] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Wokingham, England, 1989.

[84] A. Graham, *Kronecker Products and Matrix Calculus with Applications*, Halsted Press, Toronto, 1981.

[85] H. Greenberg, A quadratic assignment problem without column constraints, *Naval Research Logistic Quarterly* **16**, 1969, 417–422.

[86] S. W. Hadley, *Continuous Optimization Approaches for the Quadratic Assignment Problem*, PhD thesis, University of Waterloo, Ontario, Canada, 1989.

[87] S. W. Hadley, F. Rendl, and H. Wolkowicz, Bounds for the quadratic assignment problem using continuous optimization techniques, *Proceedings of the 1-st Integer Programming and Combinatorial Optimization Conference (IPCO)*, University of Waterloo Press, 1990, 237–248.

[88] S. W. Hadley, F. Rendl, and H. Wolkowicz, A new lower bound via projection for the quadratic assignment problem, *Mathematics of Operations Research* **17**, 1992, 727–739.

[89] S. W. Hadley, F. Rendl, and H. Wolkowicz, Nonsymmetric quadratic assignment problems and the Hoffman-Wielandt inequality, *Linear Algebra and its Applications* **58**, 1992, 109–124.

[90] P. Hahn and T. Grant, Lower bounds for the quadratic assignment problem based upon a dual formulation, to appear in *Operations Research*.

[91] P. Hahn, T. Grant, and N. Hall, Solution of the quadratic assignment problem using the Hungarian method, to appear in *European Journal of Operational Research*.

[92] G. G. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*, Cambridge University Press, London and New York, 1952.

[93] D. R. Heffley, Assigning runners to a relay team, in *Optimal Strategies in Sports*, S. P. Ladany and R. E. Machol, eds., North-Holland, Amsterdam, 1977, 169–171.

[94] C. H. Heider, A computationally simplified pair exchange algorithm for the quadratic assignment problem, Paper No. **101**, Center for Naval Analysis, Arlington, Virginia, 1972.

[95] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[96] B. Jansen. A note on lower bounds for the QAP, Technical report, Mathematics and Computer Science, Delft University of Technology, The Netherlands, December 1993.

[97] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis, How easy is local search, *Journal of Computer and System Sciences* **37**, 1988, 79–100.

[98] T. A. Johnson, New linear programming-based solution procedures for the quadratic assignment problem, Ph.D. Thesis, Clemson University, SC, 1992.

[99] M. Jünger, *Polyhedral Combinatorics and the Acyclic Subdigraph Problem*, Heldermann Verlag, Berlin, Germany, 1985.

[100] M. Jünger and V. Kaibel, A basic study of the QAP polytope, Technical Report 96.215, Institut für Informatik, Universität zu Köln, Germany, 1996.

[101] M. Jünger and V. Kaibel, On the SQAP polytope, Technical Report 96.241, Institut für Informatik, Universität zu Köln, Germany, 1996.

[102] V. Kaibel, Polyhedral Combinatorics of the Quadratic Assignment Problem, Ph.D. Thesis, Universität zu Köln, Germany, 1997.

[103] S. E. Karisch, Nonlinear Approaches for Quadratic Assignment and Graph Partition Problems, *Ph.D. Thesis*, Technical University Graz, Austria, 1995.

[104] S. E. Karisch, E. Çela, J. Clausen, and T. Espersen, A dual framework for lower bounds of the quadratic assignment problem based on linearization, SFB Report 120, Institute of Mathematics, Technical University Graz, Austria, 1997.

[105] S. E. Karisch and F. Rendl, Lower bounds for the quadratic assignment problem via triangle decompositions, *Mathematical Programming* **71**, 1995, 137–151.

[106] S. E. Karisch, F. Rendl, and H. Wolkowicz, Trust regions and relaxations for the quadratic assignment problem, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 199–220, AMS, Providence, RI.

[107] R. M. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, eds., Plenum, New York, 1972, 85–103.

[108] L. Kaufmann and F. Broeckx, An algorithm for the quadratic assignment problem using Benders' decomposition, *European Journal of Operational Research* **2**, 1978, 204–211.

[109] B. Kernighan and S. Lin, An efficient heuristic procedure for partitioning graphs, *Bell Systems Journal* **49**, 1972, 291–307.

[110] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by simulated annealing, *Science* **220**, 1983, 671–680.

[111] J. G. Klincewicz, Avoiding local optima in the p-hub location problem using tabu search and GRASP, *Annals of Operations* **40**, 1992, 283–302.

[112] J. G. Klincewicz and A. Rajan, Using GRASP to solve the component grouping problem, Technical report, AT&T Bell Laboratories, Holmdel, NJ, 1992.

[113] T. C. Koopmans and M. J. Beckmann, Assignment problems and the location of economic activities, *Econometrica* **25**, 1957, 53–76.

[114] J. Krarup and P. M. Pruzan, Computer-aided layout design, *Mathematical Programming Study* **9**, 1978, 75–94.

[115] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, D. Reidel Publishing Company, Dordrecht, 1988.

[116] A. M. Land, A problem of assignment with interrelated costs, *Operations Research Quarterly* **14**, 1963, 185–198.

[117] G. Laporte and H. Mercure, Balancing hydraulic turbine runners: A quadratic assignment problem, *European Journal of Operational Research* **35**, 1988, 378–382.

[118] E. L. Lawler, The quadratic assignment problem, *Management Science* **9**, 1963, 586–599.

[119] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, eds., *The Traveling Salesman Problem*, Wiley, Chichester, 1985.

[120] T. Lengauer, *Combinatorial Algorithms for Intergrated Circuit Layout*, Wiley, Chichester, 1990.

[121] W. Leontief, *Input-Output Economics*, Oxford University Press, New York, 1966.

[122] Y. Li and P. M. Pardalos, Generating quadratic assignment test problems with known optimal permutations, *Computational Optimization and Applications* **1**, 1992, 163–184.

[123] Y. Li, P. M. Pardalos, K. G. Ramakrishnan, and M. G. C. Resende, Lower bounds for the quadratic assignment problem, *Annals of Operations Research* **50**, 1994, 387–410.

[124] Y. Li, P. M. Pardalos, and M. G. C. Resende, A greedy randomized adaptive search procedure for the quadratic assignment problem, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 237–261, AMS, Providence, RI.

[125] L. Lovász and A. Schrijver, Cones of matrices and set functions and 0-1 optimization, *SIAM Journal on Optimization* **1**, 1991, 166–190.

[126] E. J. McCormick, *Human Factors Engineering*, McGraw-Hill, New York, 1970.

[127] T. Magnanti, R. Ahuja, and J. Orlin. *Network flows: theory, algorithms, and applications*, Prentice Hall, Englewood-Cliffs, New Jersey, 1993.

[128] F. Malucelli, Quadratic Assignment Problems: Solution Methods and Applications, Ph.D. Thesis, Dipartimento di Informatica, Universitá di Pisa, Italy, 1993.

[129] F. Malucelli and D. Pretolani, Lower bounds for the quadratic semi-assignment problem, Technical Report 955, Centre des Recherches sur les Transports, Université de Montréal, Canada, 1993.

[130] A. Marzetta, Dynamic programming for the quadratic assignment problem, presented at the *2-nd Aussois Workshop on Combinatorial Optimization*, 1998, Aussois, France.

[131] T. Mautor and C. Roucairol, A new exact algorithm for the solution of quadratic assignment problems, *Discrete Applied Mathematics* **55**, 1992, 281–293.

[132] T. Mavridou, P. M. Pardalos, L. S. Pitsoulis, and M. G. C. Resende, A GRASP for the biquadratic assignment problem, *European Journal of Operations Research* **105**, 1998, 613–621.

[133] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, Equations of state calculations by fast computing machines, *Journal of Chemical Physics* **21**, 1953, 1087–1092.

[134] I. Z. Milis and V. F. Magirou, A Lagrangean relaxation algorithm for sparse quadratic assignment problems, *Operations Research Letters* **17**, 1995, 69–76.

[135] P. B. Mirchandani and T. Obata, Locational decisions with interactions between facilities: the quadratic assignment problem a review, Working Paper Ps-79-1, Rensselaer Polytechnic Institute, Troy, New York, May 1979.

[136] L. Mirsky, The spread of a matrix, *Mathematika* **3**, 1956, 127–130.

[137] J. Mosevich, Balancing hydraulic turbine runners — a discrete combinatorial optimization problem, *European Journal of Operational Research* **26**, 1986, 202–204.

[138] K. A. Murthy, P. Pardalos, and Y. Li, A local search algorithm for the quadratic assignment problem, *Informatica* **3**, 1992, 524–538.

[139] K. G. Murty, An algorithm for ranking all the assignments in order of increasing cost, *Operations Research* **16**, 1968, 682–287.

[140] H. Müller-Merbach, *Optimale Reihenfolgen,* Springer-Verlag, Berlin, Heidelberg, New York, 1970, pp. 158–171.

[141] C. E. Nugent, T. E. Vollmann, and J. Ruml, An experimental comparison of techniques for the assignment of facilities to locations, *Journal of Operations Research* **16**, 1969, 150–173.

[142] M. W. Padberg and M. P. Rijal, *Location, Scheduling, Design and Integer Programming*, Kluwer Academic Publishers, Boston, 1996.

[143] M. W. Padberg and G. Rinaldi, Optimization of a 532-city symmetric traveling salesman problem by a branch and cut algorithm, *Operations Research Letters* **6**, 1987, 1–7.

[144] G. S. Palubeckis, Generation of quadratic assignment test problems with known optimal solutions, *U.S.S.R. Comput. Maths. Math. Phys.* **28**, 1988, 97–98, (in Russian).

[145] C. H. Papadimitriou and D. Wolfe, The complexity of facets resolved, *Proceedings of the 25-th Annual IEEE Symposium on the Foundations of Computer Science (FOCS)*, 1985, 74–78.

[146] P. Pardalos and J. Crouse, A parallel algorithm for the quadratic assignment problem, *Proceedings of the Supercomputing Conference 1989*, ACM Press, 1989, 351–360.

[147] P. Pardalos, F. Rendl, and H. Wolkowicz, The quadratic assignment problem: A survey and recent developments, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **16**, 1994, 1–42, AMS, Providence, RI.

[148] P. M. Pardalos, L. S. Pitsoulis, and M. G. C. Resende, A parallel GRASP implementation for solving the quadratic assignment problem, in *Parallel Algorithms for Irregular Problems: State of the Art*, A. Ferreira and José D.P. Rolim, eds., Kluwer Academic Publishers, 1995, 115–133.

[149] P. M. Pardalos, L. S. Pitsoulis, and M. G. C. Resende, Fortran subroutines for approximate solution of sparse quadratic assignment problems using GRASP, *ACM Transcations on Mathematical Software* **23**, 1997, 196–208.

[150] P. M. Pardalos, K. G. Ramakrishnan, M. G. C. Resende, and Y. Li, Implementation of a variable reduction based lower bound in a branch and bound algorithm for the quadratic assignment problem, *SIAM Journal on Optimization* **7**, 1997, 280–294.

[151] P. M. Pardalos and J. Xue, The maximum clique problem, Research Report 93-1, Department of Industrial and System Engineering, University of Florida, Fl, 1993.

[152] M. Queyranne, Performance ratio of heuristics for triangle inequality quadratic assignment problems, *Operations Research Letters* **4**, 1986, 231–234.

[153] G. Reinelt, *The Linear Ordering Problem: Algorithms and Applications*, Heldermann Verlag, Berlin, Germany, 1985.

[154] F. Rendl, Ranking scalar products to improve bounds for the quadratic assignment problem, *European Journal of Operations Research* **20**, 1985, 363–372.

[155] F. Rendl and H. Wolkowicz, Applications of parametric programming and eigenvalue maximization to the quadratic assignment problem, *Mathematical Programming* **53**, 1992, 63–78.

[156] M. G. C. Resende, P. M. Pardalos, and Y. Li, Fortran subroutines for approximate solution of dense quadratic assignment problems using GRASP, *ACM Transcations on Mathematical Software* **22**, 1996, 104–118.

[157] M. G. C. Resende, L. S. Pitsoulis, and P. M. Pardalos, Approximate solution of weighted max-sat problems using GRASP, in *The Satisfiability Problem*, P. M. Pardalos, M. G. C. Resende and D. Z. Du, eds., *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **35**, 1997, 393–405, AMS, Providence, RI.

[158] 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**, 1995, 781–791.

[159] W. T. Rhee, A note on asymptotic properties of the quadratic assignment problem, *Operations Research Letters* **7**, 1988, 197–200.

[160] W. T. Rhee, Stochastic analysis of the quadratic assignment problem, *Mathematics of Operations Research* **16**, 1991, 223–239.

[161] M. P. Rijal, Scheduling, Design and Assignment Problems with Quadratic Costs, Ph.D. Thesis, New York University, NY, 1995.

[162] C. Roucairol, A reduction method for quadratic assignment problems, *Operations Research Verfahren* **32**, 1979, 183–187.

[163] C. Roucairol, A parallel branch and bound algorithm for the quadratic assignment problem, *Discrete Applied Mathematics* **18**, 1987, 221–225.

[164] S. Sahni and T. Gonzalez, P-complete approximation problems, *Journal of the Association of Computing Machinery* **23**, 1976, 555–565.

[165] A. Schäffer and M. Yannakakis, Simple local search problems that are hard to solve, *SIAM Journal on Computing* **20**, 1991, 56–87.

[166] J. Skorin-Kapov, Tabu search applied to the quadratic assignment problem, *ORSA Journal on Computing* **2**, 1990, 33–45.

[167] J. Skorin-Kapov, Extensions of tabu search adaptation to the quadratic assignment problem, to appear in *Computers and Operations Research*.

[168] L. Steinberg, The backboard wiring problem: A placement algorithm, *SIAM Review* **3**, 1961, 37–50.

[169] H. S. Stone, Multiprocessor scheduling with the aid of network flow algorithms, *IEEE Transactions on Software Engineering* **3**, 1977, 85–93.

[170] W. Szpankowski, Combinatorial optimization problems for which almost every algorithm is asymptotically optimal!, *Optimization* **33**, 1995, 359–367.

[171] E. Taillard, Robust tabu search for the quadratic assignment problem, *Parallel Computing* **17**, 1991, 443–455.

[172] D. M. Tate and A. E. Smith, A genetic approach to the quadratic assignment problem, *Computers and Operations Research* **22**, 1995, 73–83.

[173] I. Ugi, J. Bauer, J. Friedrich, J. Gasteiger, C. Jochum, and W. Schubert, Neue Anwendungsgebiete für Computer in der Chemie, *Angewandte Chemie* **91**, 1979, 99–111.

[174] D. H. West, Algorithm 608: Approximate solution of the quadratic assignment problem, *ACM Transactions on Mathematical Software* **9**, 1983, 461–466.

[175] M. R. Wilhelm and T. L. Ward, Solving quadratic assignment problems by simulated annealing, *IEEE Transactions* **19**, 1987, 107–119.

[176] Q. Zhao, Semidefinite Programming for Assignment and Partitioning Problems, *Ph.D. Thesis*, University of Waterloo, Ontario, Canada, 1996.

[177] Q. Zhao, S. E. Karisch, F. Rendl, and H. Wolkowicz, Semidefinite relaxations for the quadratic assignment problem, *Journal of Combinatorial Optimization* **2**, 1998, 71–109.