The Sixth International Symposium on Operations Research and Its Applications (ISORA'06)
Xinjiang, China, August 8–12, 2006
 pp. 106–117

A Solution Method for the Quadratic Assignment Problem (QAP)

P. Ji Yongzhong Wu Haozhao Liu

Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract The classic Quadratic Assignment Problem (QAP) is one of the most interesting and challenging combinatorial optimization problems in existence. Since QAP is NP-complete, it is notoriously difficult to be solved by exact solution methods. This paper focuses on the intelligent solution methods, particularly, genetic algorithms, and related development on QAP. A hybrid genetic algorithm is devised to examine the solvability of QAP instances. Finally, advances and research trends on the solution of QAP are discussed.

1 Introduction

1.1 Problem Statement

The quadratic assignment problem (QAP) was introduced by Koopmans and Beckmann [Koo1957] in 1957 as a mathematical model for the location of indivisible economical activities. QAP is often used to describe a location problem. Let us assign n facilities to n locations with the cost being proportional to the flow between the facilities multiplied with their distances. The objective is to allocate each facility at a location such that the total cost is minimized. Thus we are given two $n \times n$ matrices, the flow matrix$A = (a_{ij})$, and the distance matrix $B = (b_{kl})$. The QAP in Koopmans-Beckmann form can now be written as

$$\min_{\pi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{\pi(i)\pi(j)} b_{ij} \tag{1}$$

where S_n is the set of permutation of $\{1, 2, \ldots, n\}$. Each individual product$a_{\pi(i)\pi(i)} b_{ij}$ is the cost caused by assigning facility $\pi(i)$ to location i and facility $\pi(j)$ to location j. An QAP instance with input matrices A and B is denoted by $QAP(A, B)$, sometimes. If any of the coefficient matrices A, B is symmetric, $QAP(A, B)$ is termed as a symmetric QAP. Otherwise, $QAP(A, B)$ is said to be asymmetric.

A slightly different problem also addressed as a QAP, and investigated by several authors is the following. Besides the two coefficient matrices A and B we are given

a third matrix $C = (c_{ij})$, whose c_{ij} is the cost of placing facility i at location j, and the problem becomes:

$$\min_{\pi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{\pi(i)\pi(j)} b_{ij} + \sum_{i=1}^{n} c_{\pi(i)i} \tag{2}$$

This problem is called the generalized Koopamans-Beckmann QAP. In the case that $c_{ij} = 0$, for all $1 \leq i, j \leq n$, we get the problems formulated in (1).

1.2 Applications

It is astonishing how many real life applications can be modeled as QAPs. A natural application in location theory was used by Dickey and Hopkins [Dic1972] in a campus planning model. The problem consists of planning the sites of n buildings on a campus, where b_{kl} is the distance from site k to site l, and a_{ij} is the traffic intensity between buildings i and j. The objective is to minimize the total weekly walking distance between the buildings.

In addition to facility location, QAPs appear in applications such as layout problems, backboard wiring, computer manufacturing, scheduling, process communications and turbine balancing. In the field of ergonomics, Burkard and Offermann [Bur1977a] showed that QAPs can be applied to typewriter keyboard design. The problem is to arrange the keys on 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 a_{ij}denotes the frequency of the appearance of the pair of symbols i and j. The entries of the distance matrix b_{kl} are the times needed to press the key in position l after pressing the key in position k. An optimal solution for this QAP minimizes the average time for writing a text. A similar application related to an ergonomic design is the development of control boards in order to minimize eye fatigue [McC1970]. Further applications concern the ranking of archeological data [Kra1978], the ranking of a team in a relay race [Hef1977], scheduling parallel production lines [Geo1976].

1.3 Complexity of Quadratic Assignment Problems

In contrast to linear assignment problems, quadratic assignment problems remain among the hardest combinatorial optimization problems. The inherent difficulty for solving QAPs is reflected by their computational complexity. Sahni and Gonzalez [Sah1976] showed that QAP is *NP*-hard and that even finding an approximate solution within some constant factor from the optimum value cannot be done in polynomial time. These results hold even for Koopmans-Beckmann QAPs with coefficient matrices fulfilling the triangle inequality, see Queyranne [Que1986].

1.4 Benchmark Instances

A large number of benchmark instances of QAP are available via QAPLIB, a library for research on the QAP [Bur1997b]. Approximate algorithms are commonly tested on these benchmark instances. QAPLIB contains currently over 100 instances that have been used in earlier researches and in part they stem from real applications

like hospital layout (like kra30* or els19), typewriter design (like bur26*), etc. In addition, QAPLIB also contains a number of other resources like pointers to literature, some source codes and links for QAP related research including a list of people with research interests in QAP. QAPLIB is very useful and certainly the first address for QAP related information in the internet.

2 Exact Algorithms and Lower Bounds

2.1 Exact Algorithms

There are three main exact methods used to find the global optimal solution for a given QAP: dynamic programming, cutting plane techniques, and branch and bound procedures. Research has shown that the latter is the most successful among exact algorithms for solving QAP. Even still, due to the overwhelming complexity of QAP, most problems with their sizes greater than $n = 30$ remain nearly intractable by exact algorithms. Since branch and bound procedures are generally the most helpful for solving QAPs, we will discuss more on the algorithms.

In typical branch and bound (B&B) algorithms for QAP, a heuristic procedure is used to generate a suboptimal, but suitable, initial feasible solution. Let us call this solution the incumbent. Then at any node of the tree, some bounding methods are used to find a "bound" on the best possible solution that can be expected from any descendent of that node, and the "bound" is compared with the objective value of the incumbent. If the incumbent is better than what we can ever expect from any solution resulting from that node, then it is safe to stop branching from that node. In other words, we can discard that part of the tree from further consideration. What is happening is that an optimal permutation is being constructed iteratively, one element at a time. Branch and bound techniques have evolved greatly over the past 40 years, starting with Gilmore [Gil1962] who in 1962 solved a QAP of size $n = 8$, to the solution of nug30, a QAP of size $n = 30$ in 2000 by Anstreicher, et al. [Ans2000].

2.2 Lower Bounds

Since QAPs are *NP*-hard, good lower bounds are of eminent importance for solving these problems by implicit enumeration procedures like branch and bound. A good bound is required that it is not too hard to compute, that it can easily be evaluated for subsets of the problem which occur after some branching and, finally, that it is tight. Although a lot of efforts have been done to derive tight and computationally efficient lower bounds, such bounds have not been found yet.

Until recently virtually most successful B&B algorithms for QAP were based on the Gilmore-Lawler lower bound (GLB) or closely related bounds (see for example [Brü1998, Mar1999]). Besides GLB, there are quite a lot of other bounds for QAPs, which include, Eigenvalue bounds [Ren1992], LP (linear programming) and dual–LP bounds [Ada1994, Res1995], Quadratic programming bounds [Ans2001], Polyhedral bounds [Jün2001, Kai2000], Semidefinite programming bounds (SDP) [Sot2002].

When comparing bounds for QAP the most important issues are the strength of a bound and the computational expense required to compute it. GLB can be calculated

routinely for all instances of QAPLIB. Eigenvalue bounds can also be computed efficiently for all symmetric instances, but its computation time is (by a constant factor) higher than GLB's. LP and SDP produce in general very strong bounds, but the computational effort outgrows the computation times for the other bounds. Currently, these bounds can not be considered efficient for problems of sizes larger than, say, $n = 30$.

3 Heuristics

The extreme difficulty of QAP has made it an ideal problem for the development of heuristic search methods. Local searches, simulated annealing, tabu search, genetic algorithms, GRASP (Greedy Randomized Adaptive Search Procedure), and other specialized methods have all been applied to QAP. The performance of different heuristics tends to vary with certain problem characteristics [Tat1995].

3.1 Local Search

A local search starts from some initial assignment and repeatedly tries to improve the current assignment by local changes. If in the neighborhood of the current assignment a better assignment is found, it replaces the current assignment and the local search continues.

In the QAP case, the neighborhood of a permutation J is typically defined by the set of permutations which can be obtained by exchanging two facilities. The simplest local search algorithm based on the above described neighborhood is an iterative improvement, which is referred to as 2-opt.

3.2 Simulated Annealing Methods

Simulated Annealing (SA) is one of the first available meta-heuristics. Therefore it is not astonishing that it was also the first one to be applied to QAP [Bur1984]. Following this implementation, some few others were proposed and currently the one due to Conolly [Con1990] appears to be the best in performing. Thonemann and Bölte [Tho1994] have proposed an improved SA algorithm for the QAP. A meta-heuristic closely related to SA, was also applied to QAP by Nissen and Paul [Nis1995].

3.3 Tabu Search

One of tabu search applications to QAP was due to Skorin-Kapov [Sko990], but probably the best known tabu search algorithm for QAP is the robust tabu search (RoTS) algorithm of Taillard [Tai1991]. This algorithm is based on the 2-opt best-improvement local search algorithm. As tabu attributes, the algorithm uses assignment of facilities to specific objects, that is, a tabu attribute $t(i, j)$ refers to the fact that it is forbidden to assign facility i to location j.

3.4 Genetic Algorithms

Genetic algorithms also receive their name from an intuitive explanation of the manner in which they behave. This explanation is based on Darwin's theory of nat-

ural selection. Genetic algorithms store a set of solutions and then work to replace these solutions with better ones based on some fitness criterion, usually the objective function value.

Conventional genetic algorithms did not find the best known solution for the Nugent's problems of sizes 20 and 30. For larger problems of size up to 100, they seldom really compete with tabu search procedures. In 2000, Ahuja, Orlin and Tiwari [Ahu2000] obtained very promising results on large scale QAPs in QAPLIB by applying a version of GA called *a greedy genetic algorithm*. Recently, Drezner [Dre2003] designed a new GA with a problem-specific crossover rule and a tabu search, and obtained even better results than those obtained by Ahuja *et al.* This new genetic algorithm is currently one of the best heuristics to solve QAPs.

3.5 Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP is a relatively new heuristic used to solve combinatorial optimization problems. GRASP was first applied to QAP in 1994 by Li, Pardalos, and Resende [Li1994]. They applied GRASP to 88 QAP instances, found the best known solution in almost every case.

3.6 Other Meta-heuristics for QAP

There are a number of applications of other heuristics to QAP. These heuristics include ant systems [Gam1999], iterated local search [Stü1999], and other specialized methods.

Today it is widely agreed that the performance of metaheuristics depends strongly on the shape of the underlying search space. Central to the search space analysis of combinatorial optimization problems is the notion of fitness landscape [Sta1995]. Intuitively, the fitness landscape can be imagined as a mountainous region with hills, craters, and valleys. The performance of metaheuristics strongly depends on the ruggedness of the landscape, the distribution of the valleys, craters and the local minima in the search space, and the overall number of the local minima. While the local properties have a large impact on the effectiveness of a local search algorithm, the global structure can be exploited by a population-based search.

4 A New Hybrid Genetic Algorithm

In order to test the solvability of QAPs, we propose a new hybrid genetic algorithm, which combines a new selection scheme and a descent local search algorithm. The instances in QAPLIB are tried.

4.1 The Algorithm

In our GA, the representation scheme is not much different from other GAs . We use an array to represent the solution, as shown in Figure 1.

The uniform crossover scheme is used in our GA. In every generation, all the individuals in the population will be applied by the crossover operator. For every pair of randomly selected parents, a small proportion of randomly selected genes are

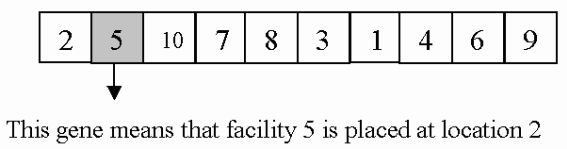

Figure 1: Representation scheme of the genetic algorithm

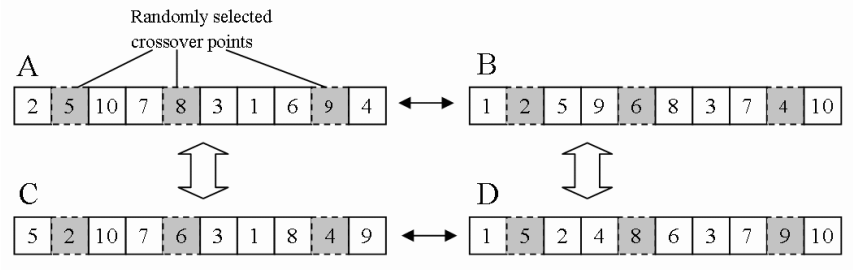

Figure 2: Crossover scheme in the GA

exchanged. The crossover process is illustrated in Figure 2. Individuals A and B produce C and D after applying the crossover. We define A as the direct parent of C, and B as the direct parent of D.

The main difference between our GA and other GAs is the selection scheme: in every generation, after crossover, every child will compare with its direct parent only, and the worst ones will be abandoned. This process is called the intra-kindred selection. And the culling process will be applied only once in every certain amount of generations. This culling process is called the inter-kindred selection, and is done by replacing a certain proportion of the individuals with the worst fitness values by those with the best fitness values. In this way, every individual is allowed to have certain time to evolve, and the potential schemata in every individual can be exploited before it is removed out of the population.

To help improve the evolution of all the individuals, a post-crossover heuristic is applied to the newborn offspring. The combined heuristic used in our GA is the descent local search heuristic, which is defined as follows:

Step 1. Examine the change in the fitness value for all pairwise exchanges of genes.

Step 2. The best improved exchange is executed and go back to Step 1 again.

Step 3. If no further improving exchange is found, the heuristic terminates.

The framework of the proposed algorithm is described as follows:

begin

generate an initial population randomly;

repeat

in every generation, do

apply the uniform crossover to each pair of individuals to generate offspring;

apply the descent local search on every child;

compare every child with its direct parent and remove the worse one;

in every T_c generations, do

culling;

until certain stopping criterion is met.

end;

The inter-kindred selection period, defined by the number of generations or T_c, is important. If T_c is too small, the chance of premature convergence is high. On the other hand, if T_c is too large, the computation effort will be great.

4.2 Computational Experiments

In this section, our genetic algorithm was tested on all instances with $30 \leq n \leq 100$ in QAPLIB, most of which still can not be solved to optimality. The results were compared with the results obtained by Ahuja *et al.* [Ahu2000], and by Drezner [Dre2003].

The population size in our GA was set to 100. The inter-kindred selection period T_c was set to be equal to the problem size n. The removal rate of the inter-kindred selection was 20%. The crossover rate was set to be 0.2, which means in every crossover process, two parents exchanged 20% of their alleles, which were selected randomly. And the algorithm was set to run until there was no improvement during T_c generations.

The program was coded in Microsoft Visual C++ 6.0, and ran on a desktop computer with Pentium III 866 CPU and 256 RAM, which was similar to the computer used by Ahuja *et al*. Ahuja *et al*. ran the algorithm only once for each problem. Drezner ran his algorithm 200 times for each problem and we ran our GA 20 times for each problem. And the results shown in Table 1 are the averages. The numeric digits in a problem name state the problem size. The percentage deviations from the best known solutions are given in the "Gap" columns.

Note: *: Number of times out of 20 runs that the best known solutions were obtained.

Gap (%): Percentage deviations over the best known solutions

For all these 44 instances, our GA found the best known solution at least once in 20 runs. It can be found that our GA performed much better than the greedy genetic algorithm designed by Ahuja *et al.* (2000), in terms of both objective value and computation time. And our GA performed as good as the genetic algorithm designed by Drezner (2003) when the problem size is not so large. But Drezner's algorithm seems to perform slightly better when the problem size is larger than 50. The main

Table 1: Comparison between the our results and the results by Ahuja et al. (2000) and Drezner (2003)

problem	Our GA			Ahuja et al (2000)		Drezner (2003)	
	*	gap(%)	time(min)	gap(%)	time (min)	gap(%)	time (min)
esc32a	20	0	0.37	0	6.36	0	0.35
esc32b	20	0	0.24	0	6.67	0	0.30
esc32c	20	0	0.20	0	6.49	0	0.27
esc32d	20	0	0.15	0	5.88	0	0.28
esc32e	20	0	0.23	0	6.16	/	/
esc32f	20	0	0.17	0	6.14	/	/
esc32g	20	0	0.33	0	6.18	/	/
esc32h	20	0	0.26	0	5.82	0	0.29
esc64a	20	0	4.13	0	43.85	/	/
kra30a	20	0	0.29	0	5.02	0	0.33
kra30b	20	0	0.34	0	5.51	0	0.33
lipa30a	20	0	0.16	0	5.74	/	/
lipa30b	20	0	0.16	0	5.62	/	/
lipa40a	20	0	1.02	0.960	17.03	/	/
lipa40b	20	0	0.90	0	17.10	/	/
lipa50a	20	0	3.31	0.950	24.77	/	/
lipa50b	20	0	2.98	0	25.14	/	/
lipa60a	20	0	10.35	0.770	50.95	/	/
lipa60b	20	0	9.97	0	50.79	/	/
lipa70a	6	0.127	30.08	0.710	102.47	/	/
lipa70b	20	0	28.23	0	102.05	/	/
lipa80a	2	0.242	47.82	0.610	158.55	/	/
lipa80b	20	0	45.25	0	158.31	/	/
lipa90a	1	0.187	61.88	0.580	205.97	/	/
lipa90b	20	0	60.08	0	205.32	/	/
nug30	20	0	0.36	0.070	5.9033	0	0.37
sko42	20	0	1.57	0.250	16.77	0	1.15
sko49	10	0.038	3.78	0.210	20.87	0.009	2.13
sko56	20	0	7.36	0.020	49.6	0.001	3.24
sko64	20	0	12.11	0.220	63.14	0	5.85
sko72	3	0.042	35.39	0.290	84.63	0.014	8.36
sko81	2	0.067	57.28	0.200	182.74	0.014	13.30
sko90	1	0.073	102.50	0.270	211.63	0.011	22.35
sko100a	2	0.051	174.13	0.210	276.80	0.018	33.55
sko100b	7	0.039	165.50	0.140	245.49	0.011	34.05
sko100c	16	0.015	158.54	0.200	338.57	0.003	33.80
sko100d	11	0.022	184.41	0.170	338.37	0.049	33.90

Continued.

Table 1: Comparison between the our results and the results by Ahuja et al. (2000) and Drezner (2003) (Continued)

problem	Our GA			Ahuja et al (2000)		Drezner (2003)	
	*	gap(%)	time(min)	gap(%)	time (min)	gap(%)	time (min)
sko100e	10	0.030	167.31	0.240	352.12	0.002	30.67
sko100f	5	0.017	170.88	0.290	357.98	0.032	35.74
ste36a	18	0.025	0.84	0.270	11.827	0.005	0.55
tho30	20	0	0.31	0	6.59	0	0.35
tho40	9	0.041	1.98	0.32	15.97	0.010	0.98
wil50	4	0.028	5.06	0.070	35.25	0.002	1.99
wil100	2	0.041	176.28	0.200	342.40	0.002	33.11

reason is that Drezner's algorithm incorporated a highly efficient tabu search and a crossover method specifically designed for QAP. In contrast, there is no specific crossover operator in our GA.

At present, most of the best solutions for the large instances in QAPLIB are obtained through meta-heuristics. The above test results indicate that even without problem-specific operators, some meta-heuristics like the hybrid genetic algorithm can perform very well in solving large QAP instances.

5 Concluding Remarks

For a long time, the extreme difficulty of QAP has made it an ideal problem for the development of heuristic search methods. However, some large problems are still challenging. On the other hand, new exact algorithms and novel computing structures developed in recent years make it possible to solve a number of long-open QAPs to optimality, including those posed by Steinberg (1961), Nugent et al. (1968) and Krarup (1972). For this reason, there has been a large amount of research on both fields of exact algorithms and heuristics.

Our opinions on advances and recent trends in the development of exact algorithms are:

- Development of tight and computationally efficient bounding procedures.
- Development of new strong branching methods for B&B algorithms.
- Utilization of distributed computation which uses multiple machines interfaced with some form of communication network.

Advances and recent trends in the development of heuristics are:

- Development of new hybrid heuristics which preferably exploit the special structure of QAP.
- Fitness landscape analysis of QAP that may help the design or parameter tuning of heuristics.

QAP still remains very challenging. More researches along these two lines are still needed. Further advances and development in these areas will certainly lead to the solution of even more difficult instances in the near future.

Acknowledgement

The work described in this paper has been supported by the Research Grants Council of Hong Kong, China (Project no.: PolyU 5259/04E).

References

[Ada1994] W. P. Adams and T. Johnson. Improved linear programming based lower bounds for the quadratic assignment problem quadratic assignment and related problems. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 16, 43–77, 1994.

[Ahu2000] R. K. Ahuja, J. B. Orlin and A. Tiwari. A descent genetic algorithm for the quadratic assignment problem. *Computers and Operations Research*, 27, 917–934, 2000.

[Ans2000] K. M. Anstreicher, N. W. Brixius, J. Linderoth and J.-P. Goux. Solving large quadratic assignment problems on computational grids. *Mathematical Programming, Series B*, 91, 563–588, 2002.

[Ans2001] K. M. Anstreicher and N. W. Brixius. A new bound for the quadratic assignment problem based on convex quadratic programming. *Mathematical Programming*, 89, 341–357, 2001.

[Brü1998] A. Brüngger, A. Marzetta, J. Clausen and M. Perregaard. Solving large-scale QAP problems in parallel with the search library ZRAM. *Journal of Parallel and Distributed Computing*, 50, 157–169, 1998.

[Bur1984] R. E. Burkard and F. Rendl. A thermodynamically motivated simulation procedure for combinatorial optimization problems. *European Journal of Operational Research*, 17, 169–174, 1984.

[Bur1977a] R. E. Burkard and J. Offermann. Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme, Z. *Operations Research*, 21, B121–B132, 1997.

[Bur1997b] R. E.Burkard, S. E. Karisch and F. Rendl. QAPLIB—a quadratic assignment problem library. *Journal of Global Optimization*, 10, 391–403, 1997.

[Con1990] D. T. Connolly. An improved annealing scheme for the QAP. *European Journal of Operational Research*, 46, 93–100, 1990.

[Dic1972] J. W. Dickey and J. W. Hopkins. Campus building arrangement using TOPAZ. *Transportation Research*, 6, 59–68, 1972.

[Dre2003] Z. Drezner. A new genetic algorithm for the quadratic assignment problem. *INFORMS Journal on Computing*, 115, 320–330, 2003.

[Gam1999] L. M. Gambardella, É. D. Taillard and M. Dorigo. Ant colonies for the quadratic assignment problem. *Journal of Operational Research Society*, 50, 167–176, 1999.

[Geo1976] 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, 595–610, 1976.

[Gil1962] P. C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM Journal on Applied Mathematics*, 10, 305–313, 1962.

[Hef1977] D. R. Heffey. Assigning runners to a relay team. *Optimal Strategies in Sports*, North-Holland, Amsterdam, 169–171, 1977.

[Jün2001] M. Jünger and V. Kaibel. On the SQAP-polytope. *SIAM Journal on Optimization*, 11, 444–468, 2001.

[Kai2000] V. Kaibel. Polyhedral methods for the QAP. *Nonlinear Assignment Problems: Algorithms and Applications*, Kluwer, 109–141, 2000.

[Koo1957] T. C. Koopmans and M. J. Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25, 53-76, 1957.

[Kra1978] J. Krarup and P. M. Pruzan. Computer-aided layout design. *Mathematical Programming Study*, 9, 75–94, 1978.

[Li1994] Y. Li, P. M. Pardalos and M. Resende. A greedy randomized adaptive search procedure for the quadratic assignment problem. In *Quadratic Assignment and Related Problems*, P. M. Pardalos and H. Wolkowicz, eds., DIMACS Series in Discrete Mathematics and Theoretical Computer Science, *AMS*, 16, 237–261, 1994.

[Mar1999] A. Marzetta and A. Brüngger. A dynamic-programming bound for the quadratic assignment problem. *Lecture Notes in Computer Science*, 1627, 339–348, 1999.

[McC1970] E. J. McCormick. *Human Factors Engineering*, McGraw-Hill, New York 1970.

[Nis1995] V. Nissen and H. Paul. A modification of threshold accepting and its application to the quadratic assignment problem. *OR Spektrum*, 17, 205–210, 1995.

[Que1986] M. Queyranne. Performance ratio of heuristics for triangle inequality quadratic assignment problems. *Operations Research Letters*, 4, 231–234, 1986.

[Ren1992] F. Rendl and H. Wolkowicz. Applications of parametric programming and eigenvalue maximization to the quadratic assignment problem. *Mathematical Programming*, 53, 63–78, 1992.

[Res1995] 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, 781–791, 1995.

[Sah1976] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the Association for Computing Machinery*, 23, 555–565, 1976.

[Sko1990] J. Skorin-Kapov. Tabu search applied to the quadratic assignment problem. *ORSA. Journal on Computing*, 2, 33–45, 1990.

[Sot2002] R. Sotirov and F. Rendl. Semidefinite relaxations for the quadratic assignment problem. Presentation in *SIAM Optimization Conference*, Toronto, Canada, 2002.

[Sta1995] P.F. Stadler. Towards a theory of landscapes. *Technical Report SFI-95-03-030*, Santa Fe Institute, 1995.

[Stü1999] T. Stützle. Iterated local search for the quadratic assignment problem. *Technical Report AIDA-99-03*, FG Intellektik, FB Informatik, TU Darmstadt, Darmstadt, Germany, March 1999.

[Tai1991] É. D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17, 443–455, 1991.

[Tat1995] D. D. Tate and A. E. Smith. A genetic approach to the quadratic assignment problem. *Computers and Operations Research*, 1, 855–865, 1995.

[Tho1994] U. W. Thonemann and A. Bolte. An improved simulated annealing algorithm for the quadratic assignment problem. *Technical Report*, Department of Production and Operations Research, University of Paderborn, Allemagne, Germany, 1994.