Analysis and Performance Measurement of Existing Solution Methods of Quadratic Assignment Problem

Morteza Karami

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the Degree of

Master of Science
in
Industrial Engineering

Eastern Mediterranean University
June 2013
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Industrial Engineering.

Asst. Prof. Dr. Gokhan İzbirak
Chair, Department of Industrial Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Industrial Engineering.

Prof. Dr. Bela Vizvari
Supervisor

Examining committee

1. Prof. Dr. Bela Vizvari
2. Assoc. Prof. Dr. Adham Mackieh
3. Asst. Prof. Dr. Sahand Daneshvar

ABSTRACT

Quadratic Assignment Problem (QAP) is known as one of the most difficult combinatorial optimization problems that is classified in the category of NP-hard problems. Quadratic Assignment Problem Library (QAPLIB) is full database of QAPs which contains several problems from different authors and with different sizes. Many exact and meta-heuristic solution methods have been introduced to solve QAP. In this thesis we focus on four previously introduced solution method of QAP e.g. Branch and Bound (B&B), Simulated Annealing (SA) Algorithm, Greedy Randomized Adaptive Search Procedure (GRASP) for dense and sparse QAPs. The codes of FORTRAN for these methods were downloaded from QAPLIB. All problems of QAPLIB were solved by the above-mentioned methods. Several results were obtained from the computational experiments part. The Results show that the Branch and Bound method is able to introduce a feasible solution for all problems while Simulated Annealing Algorithm and GRASP methods are not able to find any solution for some problems. On the other hand, Simulated Annealing and GRASP methods have shorter run time comparing to the Branch and Bound method. The performance of the methods on the objective function value is discussed also.

Keywords: Quadratic Assignment Problem, QAPLIB, Branch and Bound, Simulated Annealing, Greedy Randomized Adaptive Search Procedure (GRASP)

ÖZ

NP-zor problem kategorisinde sınıflandırılmış olan karesel Atama problemi (KAP) bilinen en zor birleşimsel optimizasyon problemidir. karesel Atama problem kütüphanesi (QAPLIB) birçok KAP veritabanı içerir ve bu veritabanlarının farklı yazar ve boyutları vardır. Birçok sezgi-ötesi çözüm yöntemleri KAP için tanımlanmıştır. Bu tezde, yoğun ve seyrek KAP için önceden tanımlanmış dört çözüm metotları olan; Dal sınır yöntemi, benzetim tavlama algoritması ve Greedy Randomized Adaptive Search Procedure (GRASP) vardır. Bu yöntemleri FORTRAN kodları KAP kütüphanesinden indirilmiştir. Bütün KAP kütüphanesindeki sorunlar bu yöntemleri çözülmüştür. Birtakım sonuçlar deneysel hesaplamalı kısımlar tarafından elde edilmiştir. Tüm problemler için Dal Sınır yöntemi, sonuçlara göre uygun bir çözümü tanıtmanın mümkün olabileceğini göstermektedir. Tavlama benzetimi algoritması ve GRASP'ın bazı sorunlar için herhangi bir uygun çözüm bulması mümkün değildir. Buna rağmen, Tavlama Benzetimi ve GRASP Metotları, Dal Sınır yöntemine göre daha kısa bir .

Anahtar Kelimeler: Karesel Atama Problemi, karesel Atama Problem Kütüphanesi, Dal Sınır Yöntemi, Tavlama Benzetim Algoritması, Greedy Randomized Adaptive Search Procedure (GRASP)

To My love

ACKNOWLEDGEMENT

There wasn't even a single step in my life that I didn't take without the support of my parents. From the first instable steps of me as a child until graduation in masters, the only thing that gave me the power and strength to overcome difficulties and the light beams of hope to find my way in darkness was the sensation of my parents being beside me. There are no words that can describe how thankful I am, and how much I want to be them beside me in this blessed day to see me not letting them down.

Being a teacher is a responsibility not many people can afford to take, and I'm proud to say I had the best teacher and guide that I could get. I want to present my special thanks to my dearest supervisor, Prof. Dr. Bela Vizvari for being so compassion and supporting. And then I really want to thank Asst. Prof. Dr. Gokhan İzbirak who was very understanding all the time and I deeply believed that I can always rely on him.

There were moments that I felt I'm lost in ignorance, hopeless and left in darkness. Then I knew the only person who can lead me through it was my dearest friend, Sadegh Niroomand, who was there for me whenever I needed his help and guidance.

I also want to specially thank my dear friend, Nima Agh, who helped me through my theses and stood up by me whenever I needed him although he was so busy with his own job and thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

The Quadratic Assignment Problem (QAP) is one of the most difficult combinatorial optimization problems which have been studied by many researchers. The aim of the problem is to assign a set of given tasks (say set 1) to another set of given tasks (say set 2) with a given assignment cost/benefit matrix in order to minimize/maximize total cost/benefit of the assignments. This problem is restricted to assign each task of set 1 to only one task of set 2 and also only one task of set 1 can be assigned to each task of set 2 by the use of binary variables (the mathematical model of QAP is described in chapter 3).

QAP has many applications in the real world where,

- the aim is to assign a set of jobs (set 1) to a set of machines/workers (set 2) with a given matrix including cost of assignment of each job to each machine/worker in order to decrease the total assignment cost.
- the aim is to assign a set of facilities (set 1) to a set of given and fixed locations (set 2) with given flow matrix of facilities and distance matrix of locations in order to decrease the cost of assignment that is calculated by multiplying the distance of a pair of location and the flow between their related facilities for all possible pairs of locations.

In addition to the above-mentioned assignment problems, QAP may be used as a mathematical formulation for the placement problem of interconnected electronic components onto an integrated circuit board or on an electronic microchip, which is a part of computer aided design in the electronics industry.

The Travelling Salesman Problem (TSP) also can be a type of QAP by the assumption that the flows join all facilities only along a single ring, all flows have the same non-zero (constant) value.

QAP is an NP-hard problem. The Branch and Bound (B&B) method is a well-known exact method for solving QAPs. The method is used in most of optimization software e.g. Lingo, Xpress, Cplex, etc. This method can be more effective for solving QAPs where some linearization techniques are used to linearize the quadratic terms of objective function. The methods were effective to reduce the running time of the problem (see He *et al*. (2012)).

On the other hand, many meta-heuristic methods were introduced to solve QAPs with large size. The meta-heuristics cannot obtain the optimal solution but they try to find a good feasible solution in a faster time comparing to exact methods. The Meta-heuristics, e.g. Simulated Annealing (SA), Genetic algorithm, Tabu search, Migrating birds, Greedy Randomized Adaptive Search Procedure (GRASP), etc. were applied on QAP by several researchers and their efficiency was proved.

A full data and information of QAP benchmarks is presented in quadratic assignment problem library (QAPLIB) webpage including the cost matrixes of all benchmarks, their related optimal or best known solutions and source code of the proposed algorithms on QAP. The data of library can be used by other researchers who want to make a study on QAP.

In this thesis all QAP benchmarks of QAPLIB is considered to be solved by four selected methods from QAPLIB. The selected methods are Branch and Bound method, Simulated Annealing algorithm, GRASP for dense QAPs and GRASP for sparse QAPs. The first method is of exact methods for QAP and the others are of meta-heuristic methods for solving QAP. Then the efficiency and performance of the methods are analyzed based on the optimal and best known solution of QAPLIB.

The thesis is organized as follow. Next chapter focuses on literature of QAP. Chapter 3 explains the QAP mathematical formulation and the above-mentioned selected methods. QAPLIB is introduced in more details in chapter 4. Chapter 5 represents computational experiments. The thesis is finished by conclusions.

Chapter 2

LITERATURE REVIEW

2.1 Practical Application for QAP

(Koopmans & Beckmann, 1957) were the first proposers of quadratic assignment problems as a mathematical model connected to the location of economic activities. After that it was shown in difference practical application: (Steinberg, 1961) used the QAP to show the optimal placement of computer element on the backboard wiring; (Dickey & Hopkins, 1972) done the Campus building arrangement by using QAP. Again the QAP is used for economic problems by (Heffley1972, 1980). (Francis & White, 1974) used QAP for assign some facilities (police posts, supermarkets, schools and so on) to best location in order to have the best service. The QAP is used to find the best place for typewriter keyboard and control panel by (Pollatschek, Gershoni, & Radday, 1976). Quadratic assignment, as a general data analysis strategy, is defined by (Hubert & Schulz, 1976). (Elshafei, 1977) used the quadratic assignment problem for the Hospital planning. (Krarup & Pruzan, 1978) applied computer aided layout design to archeology. also (Rabak & Sichman, 2003) and (Miranda *et al*. 2005) studied the best place of electronic elements. (Wess & Zeithofer, 2004) studied On the phase coupling problem between data memory layout generation and address pointer assignment. Generally, because of more benefit and importance of QAP in different industry and place, a lot of

papers have pressed and new techniques for these problems created until now and will be continue in future.

2.2 Resolution Algorithms for Solving QAP

Two general algorithms usually are used for solving optimization problems and specially QAP. The first one is exact algorithm and the second algorithm is heuristic. The most important strategies of these two methods can be explained in a short way as follows. The rest of the algorithms were not used in this project and mentioned only in order to show the vast application of QAP.

2.2.1 Exact Algorithms

- **Branch-and-Bound algorithm (B&B*)***: The B&B is one of the well-known and most frequently used methods for solving this kind of optimization problems. The brief review of this method presented in Chapter 3.
- **Dynamic programming algorithm**: This algorithm is used for some special instances that the flow matrix (matrix B) is the adjacency matrix of a tree. (Christofides & Benavent, 1989) were the first intoducers that studied and used this algorithm to the relaxed instances. After that this algorithm was improved and (Urban, 1998) used the framework of dynamic programming to obtain an optimal solution procedure.
- ***Cutting plane algorithm*:** (Bazaraa & Sherali, 1979) were the first introducer of this algorithm that at the beginning did not give a good result. This algorithm only used to small size of problems by (Kaufman & Broeckx, 1978) and (Burkard & Bonniger, 1983). In two decades later for solving on computer

motherboard design problem used Bender decomposition method by (Miranda, Luna, Mateus, & Ferreira, 2005).

- **Branch-and-cut algorithm:** It is a modified version of the B&B idea. First it was applied by (Padberg & Rinaldi, 1991) for solving symmetric matrices. (Junger & Kaibel, 2000, 2001a,b) and (Blanchard, Elloumi, Faye, & Wicker, 2003) were some of the researchers that improved and applied this algorithm.

2.2.2 Heuristic Algorithms

The heuristic method does not assure that the best solution achieved is optimal or not. This algorithm is classified in three parts: The first one is a constructive method that first time proposed by (Gilmore, 1962) and in the future developed and used by some other researchers as (Arkin, Hassin, & Sviridenko, 2001) (Gutin & Yeo, 2002). The second class is a limited enumeration that assures the obtain solution value is optimum if that obtain value go to the end of the enumerative procedure. More detail about enumeration procedure for solving quadratic assignment problems are available in (Burkard & Bonniger, 1983) and (Nissen & Paul, 1995). The third and last class of heuristic algorithms that includes most of the Quadratic Assignment Problems is improvement methods. The procedure of this method is in parallel by local search method that starts with a feasible solution and attempt to make better it. (Deinko & Woeginger, 2000) and (Mills, Tsang, & Ford, 2003) were some of the many researchers that applied and analyzed this method for solving the assignment problems.

2.2.3 Metaheuristics

The heuristic algorithms were just used for particular numerical problem before 1990s. But at the end of 1980s some general algorithms were introduced that called as

metaheuristics. Several of most important techniques that were used in assignment problem field are as follow:

- **Simulated Annealing algorithms(SA)**: the review of this method is presented and explained in Chapter 3.
- ***Genetic Algorithms***: for QAP classified in two phases, the first phase is finding the starting population by saving the best solution of each iteration. And the second phase is using Genetic Algorithms to achieve the optimal solution. For more information about the genetic procedure can see the published paper about this algorithm. (Drezner & Marcoulides, 2003) and (Wang & Sarker, 2005) were some researcher that had new idea and analyzed this algorithm.
- **Scatter search**: (Glover, 1977) was the first introducer of this method for integer programming instances. This method also contains two phases, the first phase is investigate a pleasant solution and called initial phase and the evolutionary phase (the second phase) is continue and repeat this process until achieve to a stop criterion. (Cung, Mautor, Michelon, & Tavares, 1997) studied and applied this method for the QAP instances.
- **Tabu search algorithm**: This algorithm is used to integer programming problems. (Glover, 1989a,b) was the researcher who introduced this algorithm. The more studies about this method for QAP done by some researchers same as: (Skoin-Kapov, 1990), (Misevicius, 2005) and (Drezner, 2005).
- **Greedy Randomized Adaptive Search Procedure (GRASP)**: This algorithm is one of the algorithms that used in this project, so the summery of this method present in Chapter 3. Several researchers used this algorithm to QAP same as:

(Feo & Resende, 1995), (Ahuja, Orlin, & Tiwari, 2000) and (Oliveira, Pardalos, & Resende, 2004).

Chapter 3

SELECT METHOD FOR SOLVING QAP

3.1 Branch and Bound

The optimization of NP-hard problems needs to involve capable algorithms. Branch and bound algorithm (B&B) which search the entire space of solutions is one of the well-known methods for solving these kinds of optimization problems.

In this method the solution status at any point is described by an unexplored subset and the best solution which founded so far. Only one subset exist initially (say complete solution space) and the best solution value obtained so far is infinity. The nodes in search tree represent unexplored subspaces and they contain the root initially. Three main step of iteration can be classified as: Choosing the node, computing the bound and final branching. Figure 1; represent the first step and initial condition. (Clausen, 1999)

The procedure continues with choosing the next node. The first action will be branching if subproblem selected according to the bound value. For each subspace which had single solution we keep this solution and then make comparison with the current solution and save the best one. If it does not contain any single solution, bounding function calculation and comparison with current best solution should be made. The subspace can

be discarded completely if it is founded that the optimal solution cannot be reached through the subspace, otherwise it must be saved. (Clausen, 1999)

When the solution space explored completely the procedure ends and what saved in "current best" will be the optimal solution.

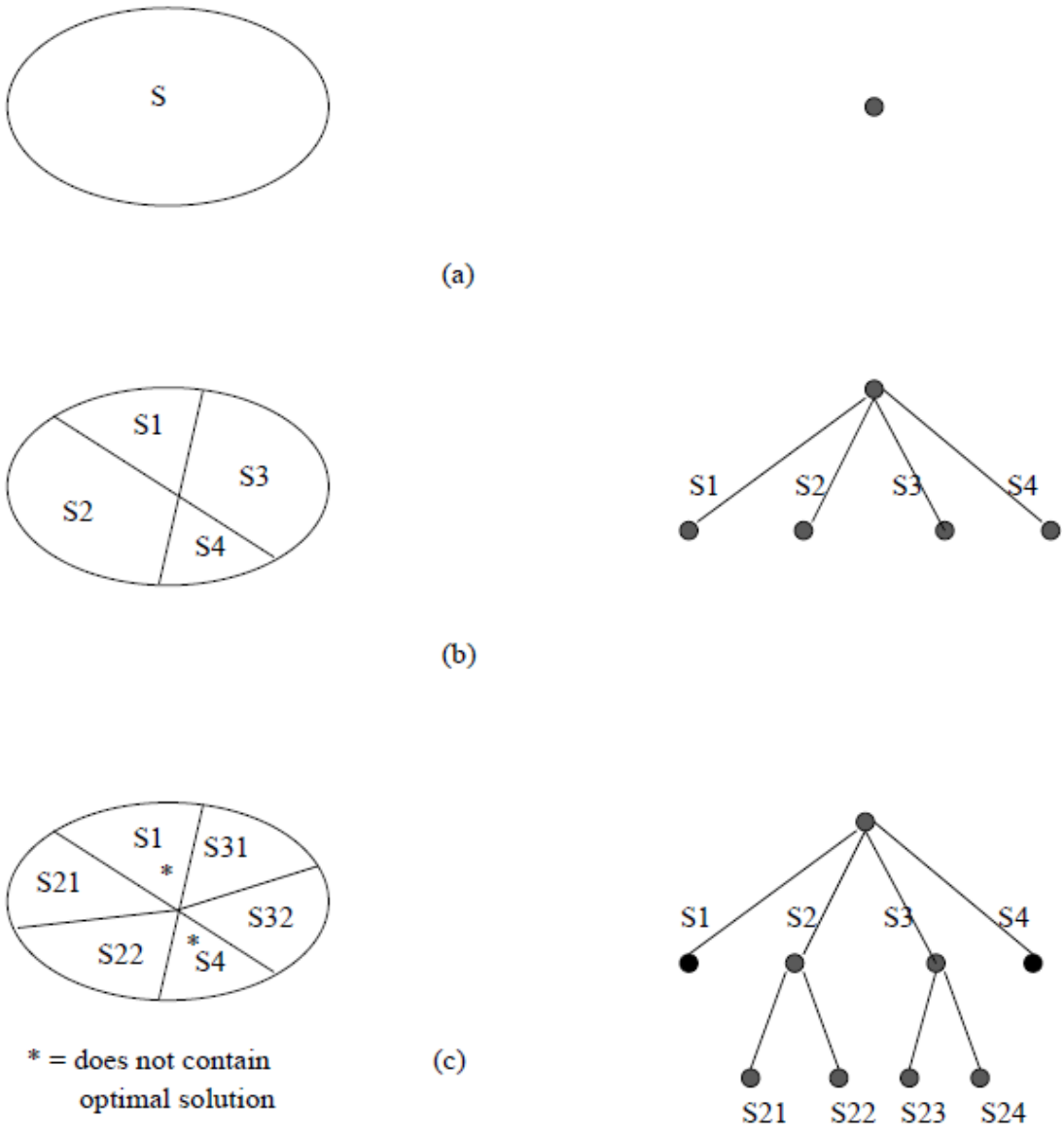

Figure 1: Illustration of the search space of B&B

Branch and Bound method for QAP classified in three steps:

Single assignment problem ((Gilmore, 1962) , (Lawler, 1963))

Pair assignment algorithms ((Gavett & Plyter, 1994), (Land, 1963), (Nugent, Vollmann, & Ruml, 1969))

Relative positioning algorithm ((Mirchandani & Obata, 1979))

(Roucairol, 1987) and (Pardalos and Grouse, 1989) were the first researchers who introduced B&B algorithms for quadratic assignment problem, but they just worked on instances of size $n \leq 15$. This algorithm was developed further on in one decade later by many researchers, for example (Gendronu & Grainic, 1994), (Feo & Resende, 1995) and (Pardalos, Resende and Ramakrishnan, 1995).

3.2 Simulated Annealing (SA)

Simulated Annealing (SA) is one of the good and useful methods for solving NP hard problems, like as QAP. The SA method is a meta-heuristic optimization method to solve combinatorial optimization problems. This method was introduced by (Kirkpatrick et al. 1983). This algorithm has been applied to many layout problems e.g. (Peng, Huanchen, & Dongme, 1996) and (Jingwei, Ting, Husheng, Jinlin, & Ming, 2012), *etc*. The SA algorithm is based on randomization techniques and works as a probabilistic local search method. Generally the SA applies an iterative improvement algorithm to a predetermined initial solution and its objective function value. The SA with continues stage can be used to find and improve the local optimization algorithm. Also we can achieve to the local optimization algorithm by generating of all solutions in the neighborhood of the current solution. Local Search evaluates the neighbor according to

the optimization criterian and selects it if it is better than the current solution, otherwise rejects the neighbor. This procedure continues until that the all or maximum number of the trials was checked and is not able to improve the solution. The SA is differ from local optimization, and the different is that in SA we can accept solution Á if it is better than solution A or worse than A. It depends on Boltzmann's law that uses to determine this acceptance probability that is shown as follow: (Singh & Sharma, 2006)

p(accept) = $e^{-\Delta z/bt}$

The objective function value of the solutions obtained by SA algorithm usually is close to optimal value.

The Simulated Annealing algorithm consists of the following steps:

Step 1: For first solution of SA we choose initial solution 'i' randomly.

Step 2: An initial temperature should be chosen (T_i>0).

Step 3: Then we should select the temperature updating function i.e. cooling (or annealing) schedule.

Step 4: The epoch length function should be selected.

Step 5: Put temperature change counter (t) and epoch length counter (l) equal zero.

Step 6: By replacing two facilities compute solution Á in the neighborhood of A.

Step 7: Compute $\Delta z = z(\acute{A}) - z(A)$

Step 8: If $\Delta z \geq 0$ go to step 9, otherwise replace a with Á and go to step 10.

Step 9: If random (0 ,1) < $\exp(-\Delta z/bt)$ then Á=A

Step 10: 3 last step (7 to 9) repeat until l=Q (the maximum Q (the number of trial) for which the temperature is 't')

Step 11: Generate and find the next temperature according to step 3 so that the function will be change and again repeat step 6 till 9 for the new temperature.

Step 12: We should do all of steps again until the stopping criteria becomes true.

In SA procedure b is a Botzmann's constant and parameter, t is called temperature that can change according to some annealing schedule and also $T_I < t < T_F$ (T_I is the initial and T_F is final temperatures respectively).

For using SA method four factors must be selected: Initial temperature, Epoch length, Annealing (Cooling) schedule and finally termination criterion. (Singh & Sharma, 2006)

3.2.1 Initial Temperature

A great initial temperature introduced by (Kirkpatrick, Gelatt Jr, & Vecchi, 1983) in optimizing by simulated annealing article that with probability 8% most of the solutions are accepted at the first stage of the SA procedure.

3.2.2 Epoch Length

If we assume N_k be the epoch length parameter (i.e. the amount of trials to be done with the same temperature value). Some functions that refer to them are as follows:

Constant function: $N_k =$ constant (k = 0,1,2,…,Q)

Arithmetic function: : $N_k = N_{k-1} +$ constant (k = 0,1,2,…,Q)

Geometric function: $N_k = \frac{N_{k-1}}{a}$ (a<1 and constant, k = 0,1,2,…,Q)

Exponential function: $N_k = \frac{constant}{\log(T_k)}$ (k = 0,1,2,…,Q)

Logarithmic function: $N_k = (N_{k-1})^{\frac{1}{a}}$ (a<1 and constant, k = 0,1,2,…,Q)

3.2.3 Cooling Schedule

A lot of functions for updating temperature are available in literature, but sum of the most important ones are as follows:

Arithmetic function: : $t_{k+1} = t_k -$ constant (k = 0,1,2,…,Q)

Geometric function: $t_{k+1} = \alpha . t_k$ (α<1, $t_0 = T_I(constant)$, k = 0,1,2,…,Q)

Inverse function: $t_{k+1} = \frac{t_k}{1+\alpha . t_k}$ (α $< t_0$, $t_0 = T_I(constant)$, $t_k = \frac{constant}{t_{k+1}}$, k=0,1,2,…,Q)

Logarithmic function: $t_k = \frac{constant}{\log(t_{k+2})}$ (k = 0,1,2,…,Q)

3.2.4 Termination Criterion

In the literature lot of tests available and explained that some of them are as follows:

If we can't find any improvement in a certain amount of iterations;

If all of the iterations have been done;

If the previously defined number of acceptance for a given number of trials has not been obtained;

If we reached the target temperature;

3.3 GRASP Method

Greedy Randomized Adoptive Search Procedure (GRASP) is a good method that mixes lot of favorable properties of other heuristics. In GRASP can select number of iterations as each iteration includes two steps, a first one is construction and the second is local search. Each iteration gives a special solution and the best solution from all iterations is selected for the final solution of the main problem. The GRASP procedure is shown on figure 2: (Pardalos & Crouse, 1989)

```
procedure grasp()
1       InputInstance();
2       do stopping criterion not satisfied →
3           ConstructGreedyRandomizedSolution(solution);
4           LocalSearch(solution);
5           SaveSolution(solution,bestsolution);
6       od;
7       return(bestsolution)
end grasp;
```

Figure 2: General case for GRASP pseudo-code

In row number one we have inputs that consists two matrices, the flow matrix (F) and distance matrix (D). The stopping criterion in line 2 is a maximum number of iteration that we select or a solution value that we are satisfied or anything else. Line 3 is the construction step (shown in Figure 3) and line 4 is the local search step of GRASP procedure. In line 5 records the best solution until now and continue. (Pardalos & Crouse, 1989)

Figure 3 illustrate the summary of construction step. In this figure line 1 means that the construction step start with empty solution. In line 3 create a Restricted Candidate list (RCL). RCL means a list of best components, from that a selection will be created. The selecting random elements from RCL list is done in line 4. In line 5 the new solution is considered as the last solutions set with added s. The last elements (without added s) are computed in the greedy function, where the loop starting with a new RCL will be constructed.

```
procedure ConstructSolution(solution)
1       solution={};
2       do Solution is not complete →
3           MakeRCL(RCL);
4           s=SelectRandomElement(RCL);
5           solution=solution ∪{s};
6           AdaptGreedy(s,RCL);
7       od;
8       return(solution)
end ConstructSolution;
```

Figure 3: GRASP construction step pseudo-code

The construction step can be implemented in two ways, one of them is sparse procedure and another one is dense procedure. Both of these procedures have similar function, but the most important thing is that the sparse procedure is considerably faster rather than dense procedure. This fasting is perceptible in our computations.

For see the literature survey of a GRASP can refer to (Feo & Resende, 1995).

3.3.1 Initial Construction Phase for QAP

Each QAP instance consists of two $n \times n$ matrices which are the flow matrix F= (f_{ij}) and the distance matrix D=(d_{ij}). $f_{ij}d_{kl}$ is the transportation cost between facilities i and j if they are assigned to locations k and l. α and β are assumed the candidate list restriction parameters. Then the distance matrix elements classified in increasing order and flow matrix element classified in decreasing order that shown as follow:

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

$$f_{i_1 j_1} \geq f_{i_2 j_2} \geq \ \dots \ \geq \ f_{i_m j_m}$$

If $\lfloor \beta_m \rfloor$ is the minimum of d_{kl} elements and also $\lfloor \beta_m \rfloor$ is the maximum of f_{ij} elements, the corresponding cost is as follows:

$f_{i_1 j_1} d_{k_1 l_1}$, $f_{i_2 j_2}\ d_{k_2 l_2}$, … , $f_{i_{[\beta_m]}\, j_{[\beta_m]}}\ d_{k_{[\beta_m]}\, l_{[\beta_m]}}$

These costs are classified in increasing order and the cost corresponding to some couple assignment shown with each element.

In in this phase according to two couple assignment, two selection elements take from the previous cost set. For complete implementation details about this phase can refer to (Li, Pardalos, & Resende, 1994).

3.3.2 Local Search for QAP

In this step the current solution (s) is improved by investigating the neighborhood N(s) of that solution. QAP include n! permutation. First of all assume $\delta(p,q) = \{i | p(i) \neq q(i)\}$ is the difference between permutation p and permutation q and $d(p,q) = |\delta(p,q)|$ is the distance of this this two permutations. For different problems, many different ways are available to construct N(s). Some of the general ones are as follows (Li (1992)):

The reasonable neighborhood size: could be possible to investigate in the rational number of calculation.

Large variance in the neighborhood: show that the maximum distance of all of the permutations is large if a special neighborhood conclude all of the permutations $p_1, p_2, \dots, p_n$.

Connectivity in the neighborhood: it means that the sequential of permutation $\{p_k\}$ with small $\delta(p_k, p_{k+1})$ are available while we moving from permutation $p_i \; to \; p_j \; (k = i, i + 1, \dots, j - 1)$.

The k-exchange neighborhood is used for GRASP local search, which is as follows:

$$N_k(p) = \{q | d(p, q) \leq k, 2 \leq k \leq n\}.$$

The initial permutation p is used to start the k-exchange local search, and then all of the permutations generated by exchanging of k between each other. The local optimum recorded according to the best permutation. The size of this neighborhood is p_k^n.

If $k = 2$ then the neighbors are as follows:

$$\begin{aligned}
m &= 1 && q = (2,1,3, \dots, n-2, n-1, n) \\
m &= 2 && q = (3,2,1, \dots, n-2, n-1, n) \\
&\cdot && \cdot \\
&\cdot && \cdot \\
&\cdot && \cdot \\
m &= p_k^n - 1 && q = (1,2,3, \dots, n-1, n-2, n) \\
m &= p_k^n && q = (1,2,3, \dots, n-2, n, n-1)
\end{aligned}$$

In looking for the neighborhood, the k-exchange local search processing mentioned above ceases when a better solution is achieved, and starts penetrating in the neighborhood until the most valued solution is found at the same manner. The First Decrement search is comparing to Complete Enumeration search, but Complete Enumeration checks all of the solution in neighborhood, and this procedure continues for the best outcome is suggested as before. However in both case a neighborhood completely search (according to p_k^n solution). But in GRASP procedure generally the

initial permutation is in the good condition and the local search just search in few percent of the neighborhood. (Pardalos & Crouse, 1989)

Chapter4

QUADRATIC ASSIGNMENT PROBLEM LIBRARY

In order to create a standard test set for QAP, in 1991 QAPLIB was established which is accessible for all researchers and all QAP instances that were available to the authors at that time, are included in it. Following the huge positive feedback and continuing demands from scientific community in 1994 Bukard, Karisch and Rendl provided a major update which was even accessible through anonymous ftp as well. Many new problem instances which were generated by researchers for their own testing purposed, were included in that update.

In April 1996 QAPLIB was updated again. On one hand this update reflected on the big changes in electronic communication, which means, QAPLIB became a World Wide Web site, the QAPLIB Home Page. On the other hand, research activities around QAP were increased so much that another update was necessary and some recent (at that time) dissertations were added to the library.

Some of the test instances were not solved optimally before, so another update released in June 2000 which contained new test instances and the optimal solutions for non-optimally solved old problems.

The next update was in January 2002 that again consists of new test instances and improvements on the best known feasible solutions, especially test the instances did not achieve to the optimal solution yet.

4.1 Problem Instances

In this part we discuss the problem instances that are available in the QAPLIB. The size of instances are $12 \leq n \leq 256$, and four largest ones of size 128, 150(two instances) and 256. Instances of size $n < 12$ are not considered in the QAPLIB, because this size of problems can be solved so easily even without using any software.

The problems that solved in the QAPLIB consists different parts:

n: is the size of the instance

A and B: are distance and flow matrices

P: is a corresponding permutation

Sol: is an objective function value

Also the solution of the problems classified to two groups:

1) The problem instances where optimality is. Their optimal permutation is also provided.
2) The problem instances where optimality is not achieved. Their best known feasible solution and the gap relative to the best known lower bound are also provided.

The most important lower bounds that used for QAPLIB are as follow:

- The Elimination Bound (ELI) ((Hadley, Rendl, & Wolkowicz, 1992))
- The Gilmore-Lawer Bound (GLB)((Gilmore, 1962), (Lawler, 1963))
- An Interior Point Based Linear Programming (IPLP) ((Resende, Ramakrishman, & Drezner, 1995))
- A Semidefinite Programming Bound (SDP) ((Zhoa, Karisch, Rendl, & Wolkowicz, 1996))
- A Triangle Decomposition Bound (TDB) ((Karisch & Rendl, 1995))

The lower bound for all of the instances can be regularly computed by GLB. The ELI is only used to symmetric matrices. The TDB can be used for instances such that the distance matrix has metric structure. The bound SDP and IPLP are very strong bounds, but only for size $n \leq 30$.

The available problems in QAPLIB are as follows:

1) (A. N. Elshafei, 1977): The data describes the hospital layout between patient's locations according to the distances.
2) (Eschermann & Wunderlich, 1990): The purpose is to optimized synthesis of self-testable finite state machines.
3) (Roucairol, 1987): The intervals are used to matrices data.
4) (Nugent, Vollmann, & Ruml, 1969): The distance matrix provided by this author contains the most frequently used problem distances (Manhattan distances of rectangular grids).

5) (E.D. Taillard 1991, 1994): The instances Tai(size)a are uniformly created, the Instances Tai(size)b are symmetric with randomly created and the instances Tai(size)c happen in the creation of grey pattern.
6) (J. Krarup and P.M. Pruzan 1978): These problems consists real data that used in the design of the layout in Germany for Klinikum Regensburg.
7) (Skoin-Kapov, 1990): The distances and flow matrices of this problem are respectively rectangular and pseudorandom.
8) (Steinberg, 1961): Contain a placement algorithm about the backboard wiring problem.
9) (Scriabin & Vegin, 1975): This problem was the comparison of computer algorithms and visual based method for plant layout. Matrix A is rectangular matrices.
10) (Wilhelm & Ward, 1987): The distance matrices are rectangular.
11) (Christofides & Benavent, 1989): Matrix A is the being adjacent matrix of a weighted tree the other that of a complete graph.
12) (R.E. Burkard and J. Offermann, 1977): The typing-time of an average stenotypist collected in matrix A and matrix B described the frequency of couples of letters in different languages.
13) (Hadley, Rendl, & Wolkowicz, 1992): The data in matrix A describes the Manhattan distance of a joined cellular complex in the flat, and the data in matrix B(flow matrix) are uniformly attracted from the interval [1,n].
14) (Thonemann & Bolte, 1994): Again the matrix A is rectangular.
15) (Y. Li and P.M. Pardalos, 1992): These researchers generating quadratic assignment test problems with known optimal solutions and permutations.

Chapter 5

DESIGNING EXPERIMENTS

5.1 Recognition of System

The studied case in this project is solving the instances that available in QAPLIB homepage with several methods and then make comparison the performance of each method. This depends on the amount of the feasible solution and also the efficacy of algorithm for solving this kind of the optimization problems. In such comparisons we need two values. The first value was taken from QAPLIB home page and the second value is obtained by solving the instance problems by different algorithms. The methods were coded by FORTRAN 6. For this experiment, desktop computers with 3.00 GHz processor, 960 Mb RAM with windows XP 2008-32 were used.

5.2 Choice of Factor and Selection of the Response Variable

- In solving these instances the treatment levels of the following factors were selected by randomization techniques:
- Selected method
- software
- hardware

The selecting method is the aim of this project. Generally we have two kinds of algorithms for solving optimization problem, exact and heuristic algorithm. So we should select methods from these algorithms that yields better solution. In exact algorithm the only choice for solving this kind of problem is B&B, so the first level of this part is B&B method. In the heuristic algorithm we selected three methods and the purpose of this selection is that, these three heuristic methods are the most popular methods that frequently used by operation researchers. The feasible solution of each instance shows the accuracy of the methods that used for performance comparison.

The software that used for solving the problem is FORTRAN 6 that installed on all of the computers at the same time. The reason for using this software is that, this software is programmable for a lot of method. So the FORTRAN was one of the best choices that can be used for these four methods.

In hardware part the important point is the specification, operation and utilization of the computers. The specification of all computers are same. In hardware part two parameters are more effective:

Warm-up effect: that can affect the runtime of the instances. Because this factor leads to some delay for solving the problems and maybe some problem can achieve the optimum in one week because of this delay.

Age of operation system: this factor also can affect the runtime of the instances. Even though, we are using the same operating system (windows XP), it is known that the

performance may drop down as time passes. Therefore, to schedule which PC to use in every experiment randomization techniques were used.

So the effects of the software and hardware were completely minimized by randomization.

5.3 Experiment Design

This project is one-way or single-factor analysis of variance model because only one factor is investigated (the selection method) and $a = 4$ levels of factor (or treatment or the number of methods that used).

5.4 Performing the Experiment

The computation was made by FORTRAN 6 and 10 desktop computers with 3.00 GHz processor and 960 Mb RAM with windows XP 2008-32 as the operation system were used. At first we run all of the heuristic methods, because it is obvious for all operation researchers that these methods solve the problems in very short time. So the perform start with computer #1 and SA method. Because this solving Length in few second, so after solving the instance, the feasible solution and runtime recorded and then another instance run. This process continues until all of the instances run with this method. When this method finished, the GRASP sparse method that planned in FORTRAN use for solving the instances and after run all of the instances the GRASP dense used for solving the instances. After I finish running all of the heuristic methods, the B&B method was applied. For solving the instances by this method 10 computers were used and 10 instances randomly run on computers. In this step if solving process by computer

was finished before 1week, drop it and record the optimum solution and run time and after 1day another instance should be applied on that computer. If solution was not obtained during 1week, we drop it and record the final feasible solution and run time; This process continue till all the instances applied.

5.5 Statistical Analysis of the Data

This part will be explained in Chapter 6.

Chapter 6

EXPERIMENTATION

This chapter presents certain numerical results for all of the instances in the QAPLIB (134 instances) were calculated with 4 different algorithms which have been explained in chapter 3 in extensively. The calculations were made by FORTRAN software which is available in QAPLIB home page and FORTRAN codes were modified because the original codes was not appropriate for solving general sizes in the instances so the new codes supported different input sizes of the matrices. Maximum number of iterations for GRASP method was set to 100 times and the parameters $\alpha = 0.25$ and $\beta = 0.5$ were initialized. 12 desktop computers with processors with Pentium® D 3.00 GHz and 960 Mb RAM with windows XP 32 bit as the operating system, were connected to a network and all runtimes correspond to the parallelized processors.

Summary of all the experiments with 4 mentioned methods and the existing results in the literature are shown and compared in table 1.

Table 1: Final results

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
1	Els19	17212548	Opt	**17212548**	**0.08**	1.0	18059476	0.03	1.0	17937024	0.09	1.0	**17212548**	0.09	1.0
2	Esc16a	68	Opt	**68**	2093.97	1.0	**68**	**0.01**	1.0	**68**	0.04	1.0	**68**	0.04	1.0
3	Esc16b	292	Opt	**292**	397366.63	1.0	**292**	**0.03**	1.0	**292**	0.06	1.0	**292**	0.05	1.0
4	Esc16c	160	Opt	**160**	10267.84	1.0	**160**	**0.03**	1.0	**160**	0.06	1.0	**160**	0.05	1.0
5	Esc16d	16	Opt	**16**	1785.50	1.0	**16**	**0.03**	1.0	**16**	0.04	1.0	**16**	0.03	1.0
6	Esc16e	28	Opt	**28**	7403.00	1.0	**28**	**0.02**	1.0	**28**	0.04	1.0	**28**	0.03	1.0
7	Esc16f	0	Opt	**0**	0.01	1.0	**0**	**0.03**	1.0	**0**	0.04	1.0	**0**	0.03	1.0
8	Esc16g	26	Opt	**26**	3343.61	1.0	**26**	**0.02**	1.0	**26**	0.06	1.0	**26**	0.03	1.0
9	Esc16h	996	Opt	**996**	35319.78	1.0	**996**	**0.02**	1.0	**996**	0.06	1.0	**996**	0.05	1.0
10	Esc16i	14	Opt	**14**	3090.97	1.0	**14**	**0.03**	1.0	**14**	0.06	1.0	**14**	0.05	1.0
11	Esc16j	8	Opt	**8**	4994.17	1.0	**8**	**0.02**	1.0	**8**	0.04	1.0	**8**	0.03	1.0
12	Esc32a	130	Opt	148	1w	1.1	154	0.03	1.2	**136**	**0.40**	1.0	140	0.25	1.1
13	Esc32b	168	Opt	**168**	1w	1.0	192	0.05	1.1	192	0.42	1.1	184	0.25	1.1
14	Esc32c	642	Opt	648	1w	1.0	**642**	**0.05**	1.0	**642**	0.40	1.0	**642**	0.23	1.0
15	Esc32d	200	Opt	210	1w	1.1	**200**	**0.03**	1.0	**200**	0.39	1.0	**200**	0.23	1.0
16	Esc32e	2	Opt	**2**	1w	1.0	**2**	**0.03**	1.0	**2**	0.29	1.0	**2**	0.19	1.0
17	Esc32g	6	Opt	10	1w	1.7	**6**	**0.03**	1.0	**6**	0.31	1.0	**6**	0.20	1.0
18	Esc32h	438	Opt	470	1w	1.1	**438**	**0.03**	1.0	**438**	0.40	1.0	440	0.27	1.0
19	Esc64a	116	Opt	**116**	1w	1.0	**116**	**0.09**	1.0	**116**	3.57	1.0	**116**	1.48	1.0
20	Esc128	64	Opt	72	1w	1.1	**68**	0.04	1.1	N	N	N	N	N	N
21	Rou12	235528	Opt	**235528**	0.30	1.0	246282	0.02	1.0	**235852**	0.05	1.0	**235852**	**0.03**	1.0

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
22	Rou15	354210	Opt	**354210**	22.77	1.0	356874	0.03	1.0	**354210**	0.06	1.0	**354210**	**0.05**	1.0
23	Rou20	725522	Opt	**725522**	**155702.50**	1.0	750154	0.03	1.0	736360	0.13	1.0	736360	0.13	1.0
24	Nug12	578	Opt	**578**	0.34	1.0	600	0.03	1.0	**578**	**0.03**	1.0	582	0.05	1.0
25	Nug14	1014	Opt	**1014**	**5.25**	1.0	1046	0.03	1.0	1018	0.05	1.0	1026	0.05	1.0
26	Nug15	1150	Opt	**1150**	17.70	1.0	1170	0.02	1.0	**1150**	0.06	1.0	**1150**	**0.05**	1.0
27	Nug16a	1610	Opt	**1610**	**78.02**	1.0	1642	0.03	1.0	1612	0.06	1.0	1622	0.05	1.0
28	Nug16b	1240	Opt	**1240**	97.39	1.0	1270	0.03	1.0	**1240**	**0.06**	1.0	**1240**	**0.06**	1.0
29	Nug17	1732	Opt	**1732**	**1059.72**	1.0	1776	0.03	1.0	1750	0.08	1.0	1750	0.06	1.0
30	Nug18	1930	Opt	**1930**	**6429.63**	1.0	1964	0.03	1.0	1946	0.08	1.0	1936	0.09	1.0
31	Nug20	2570	Opt	**2570**	**142231.22**	1.0	2598	0.02	1.0	2500	0.13	1.0	2500	0.09	1.0
32	Nug21	2438	Opt	**2438**	**1w**	1.0	2476	0.03	1.0	2444	0.13	1.0	2458	0.13	1.0
33	Nug22	3596	Opt	3612	1w	1.0	3642	0.03	1.0	3604	0.16	1.0	**3596**	**0.14**	1.0
34	Nug24	3488	Opt	3596	1w	1.0	3592	0.03	1.0	**3516**	**0.20**	1.0	3534	**0.20**	1.0
35	Nug25	3744	Opt	3806	1w	1.0	3762	0.03	1.0	3788	0.23	1.0	**3754**	**0.22**	1.0
36	Nug27	5234	Opt	5376	1w	1.0	5350	0.05	1.0	**5272**	0.30	1.0	**5272**	**0.28**	1.0
37	Nug28	5166	Opt	5368	1w	1.0	5304	0.05	1.0	**5240**	**0.31**	1.0	5242	**0.31**	1.0
38	Nug30	6124	Opt	6344	1w	1.0	6336	0.05	1.0	**6216**	0.42	1.0	6286	**0.41**	1.0
39	Tai12a	224416	Opt	**224416**	**0.06**	1.0	229982	0.03	1.0	229092	0.03	1.0	229092	0.03	1.0
40	Tai12b	39464925	Opt	**39464925**	**12.22**	1.0	N	N	N	N	N	N	N	N	N
41	Tai15a	388214	Opt	**388214**	**39.86**	1.0	393754	0.03	1.0	388988	0.05	1.0	388988	0.05	1.0
42	Tai15b	51765268	Opt	**51765268**	**13.72**	1.0	N	N	N	N	N	N	N	N	N
43	Tai17a	491812	Opt	**491812**	**476.56**	1.0	503348	0.02	1.0	501556	0.06	1.0	502840	0.08	1.0

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
44	Tai20a	703482	Opt	**703482**	**85169.50**	1.0	730074	0.03	1.0	714464	0.13	1.0	714464	0.13	1.0
45	Tai20b	122455319	Opt	**135725519**	1w	1.1	N	N	N	N	N	N	N	N	N
46	Tai25a	1167256	Opt	**1193888**	**1w**	1.0	1221360	0.05	1.0	1197489	0.23	1.0	1197489	0.23	1.0
47	Tai25b	344355646	Opt	**434386511**	1w	1.3	N	N	N	N	N	N	N	N	N
48	Tai30a	1818146	6.12%	**1847984**	**1w**	1.0	1880396	0.05	1.0	1851736	0.42	1.0	1851736	0.42	1.0
49	Tai30b	637117113	Opt	**7738220240**	1w	12.1	N	N	N	N	N	N	N	N	N
50	Tai35a	2422002	8.48%	2502538	1w	1.0	2537576	0.06	1.0	**2486540**	**0.69**	1.0	**2486540**	0.70	1.0
51	Tai35b	283315445	14.52%	**338594939**	1w	1.2	N	N	N	N	N	N	N	N	N
52	Tai40a	3139370	9.43%	3230752	1w	1.0	3275670	0.06	1.0	3224270	1.08	1.0	**3222338**	**1.13**	1.0
53	Tai40b	637250948	11.43%	**810247590**	1w	1.3	N	N	N	N	N	N	N	N	N
54	Tai50a	4938796	11.09%	5106178	1w	1.0	5122220	0.14	1.0	**5096724**	**2.39**	1.0	**5096724**	**2.39**	1.0
55	Tai50b	458821517	13.79%	**532777736**	1w	1.2	N	N	N	N	N	N	N	N	N
56	Tai60a	7205962	22.91%	**7427476**	**1w**	1.0	7466130	0.22	1.0	7446714	4.59	1.0	7439286	4.66	1.0
57	Tai60b	608215054	10.82%	**749514850**	1w	1.2	N	N	N	N	N	N	N	N	N
58	Tai64c	1855928	Opt	1871994	1w	1.0	1871994	0.03	1.0	**1855928**	3.13	1.0	**1855928**	**1.41**	1.0
59	Tai80a	13499184	22.20%	**13778682**	**1w**	1.0	13948756	0.56	1.0	13874024	12.98	1.0	13874024	12.98	1.0
60	Tai80b	818415043	12.28%	**974280375**	1w	1.2	N	N	N	N	N	N	N	N	N
61	Tai100a	21052466	24.86%	**21436242**	**1w**	1.0	21545924	1.34	1.0	21643136	29.47	1.0	21616536	29.63	1.0
62	Tai100b	1185996137	10.78%	**—**	1w	N	N	N	N	N	N	N	N	N	N
63	Tai 150b	498896643	11.45%	**552849458**	1w	1.1	N	N	N	N	N	N	N	N	N
64	Tai256c	44759294	2.03%	**44951708**	1w	1.0	45069154	2.86	1.0	N	N	N	N	N	N
65	Kra30a	88900	Opt	97820	1w	1.1	94270	0.05	1.1	91160	0.39	1.0	**88900**	**0.33**	1.0
66	Kra30b	91420	Opt	95920	1w	1.0	95080	0.05	1.0	**92150**	**0.39**	1.0	92990	0.33	1.0

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
67	Kra32	88700	Opt	96040	1w	1.1	93550	0.05	1.1	91120	0.47	1.0	**90600**	**0.41**	1.0
68	Sko42	15812	5.56%	16112	1w	1.0	16058	0.13	1.0	**15888**	**1.30**	1.0	15938	1.30	1.0
69	Sko49	23386	5.91%	24280	1w	1.0	23880	0.16	1.0	**23600**	**2.30**	1.0	23722	2.23	1.0
70	Sko56	34458	5.37%	37034	1w	1.1	35030	0.25	1.0	34902	3.67	1.0	**34854**	**3.64**	1.0
71	Sko64	48498	5.70%	50060	1w	1.0	49294	0.34	1.0	**49138**	**6.14**	1.0	49144	6.07	1.0
72	Sko72	66256	5.38%	68614	1w	1.0	67234	0.45	1.0	**66802**	**8.92**	1.0	67088	9.06	1.0
73	Sko81	90998	5.41%	94422	1w	1.0	**91894**	**0.70**	1.0	92042	13.77	1.0	91944	13.78	1.0
74	Sko90	115534	5.63%	119040	1w	1.0	**116542**	**1.02**	1.0	116704	20.08	1.0	116850	20.61	1.0
75	Sko100a	152002	5.37%	157686	1w	1.0	153594	1.47	1.0	**153502**	**29.69**	1.0	153576	29.52	1.0
76	Sko100b	153890	5.44%	158818	1w	1.0	155844	2.19	1.0	**155430**	**29.31**	1.0	155588	29.33	1.0
77	Sko100c	147862	5.54%	151972	1w	1.0	150054	1.59	1.0	149654	29.94	1.0	**149436**	**29.97**	1.0
78	Sko100d	149576	5.54%	153980	1w	1.0	150882	1.59	1.0	151164	29.41	1.0	**150878**	**29.33**	1.0
79	Sko100e	149150	5.54%	155778	1w	1.0	150508	1.73	1.0	**150422**	**30.80**	1.0	150936	29.81	1.0
80	Sko100f	149036	5.60%	154660	1w	1.0	**150184**	**1.63**	1.0	150500	29.92	1.0	150738	29.02	1.0
81	Ste36a	9526	Opt	10706	1w	1.1	10100	0.08	1.1	**9860**	**0.75**	1.0	9922	0.59	1.0
82	Ste36b	15852	Opt	21814	1w	1.4	17442	0.08	1.1	16760	0.75	1.1	**16286**	**0.64**	1.0
83	Ste36c	8239110	Opt	9057516	1w	1.1	8837824	0.08	1.1	8574764	0.75	1.0	**8483344**	**0.63**	1.0
84	Scr12	31410	Opt	**31410**	0.09	1.0	**31410**	**0.02**	1.0	**31410**	0.03	1.0	**31410**	0.05	1.0
85	Scr15	51140	Opt	**51140**	0.81	1.0	54454	0.02	1.1	53108	0.06	1.0	**51140**	**0.05**	1.0
86	Scr20	110030	Opt	**110030**	**1519.03**	1.0	113946	0.03	1.0	111732	0.11	1.0	111420	0.11	1.0
87	Wil50	48816	3.52%	49756	1w	1.0	49190	0.19	1.0	**49110**	**2.41**	1.0	49122	2.41	1.0
88	Wil100	273038	3.52%	278168	1w	1.0	275078	1.86	1.0	**274646**	**29.94**	1.0	274798	30.23	1.0

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
89	Chr12a	9552	Opt	N	N	N	N	N	N	N	N	N	N	N	N
90	Chr12b	9742	Opt	**9742**	**0.02**	1.0	10398	0.02	1.1	**9742**	0.03	1.0	**9742**	0.03	1.0
91	Chr12c	11156	Opt	**11156**	**0.03**	1.0	12984	0.03	1.2	11926	0.05	1.1	11770	0.03	1.1
92	Chr15a	9896	Opt	**9896**	**0.07**	1.0	12052	0.03	1.2	10106	0.05	1.0	10864	0.05	1.1
93	Chr15b	7990	Opt	**7990**	**0.09**	1.0	11992	0.02	1.5	9424	0.05	1.2	9164	0.05	1.1
94	Chr15c	9504	Opt	**9504**	**0.62**	1.0	12344	0.02	1.3	10282	0.05	1.1	11074	0.05	1.2
95	Chr18a	11098	Opt	**11098**	**0.36**	1.0	16054	0.02	1.4	13276	0.08	1.2	12886	0.06	1.2
96	Chr18b	1534	Opt	**1534**	**0.03**	1.0	1686	0.02	1.1	1602	0.08	1.0	1574	0.06	1.0
97	Chr20a	2192	Opt	**2192**	**0.41**	1.0	2908	0.02	1.3	2592	0.09	1.2	2252	0.08	1.0
98	Chr20b	2298	Opt	**2298**	**0.92**	1.0	2960	0.02	1.3	2700	0.11	1.2	2778	0.08	1.2
99	Chr20c	14142	Opt	**14142**	**1.75**	1.0	20162	0.03	1.4	16762	0.09	1.2	17338	0.08	1.2
100	Chr22a	6165	Opt	**6156**	**0.47**	1.0	6862	0.03	1.1	6376	0.14	1.0	6494	0.11	1.1
101	Chr22b	6194	Opt	**6194**	**0.84**	1.0	6810	0.03	1.1	6630	0.14	1.1	6620	0.13	1.1
102	Chr25a	3796	Opt	**3796**	**66.08**	1.0	5098	0.05	1.3	5096	0.22	1.3	4412	0.16	1.2
103	Bur26a	5426670	Opt	**5503071**	1w	1.0	—	1w	N	N	N	N	N	N	N
104	Bur26b	3817852	Opt	**3916200**	1w	1.0	—	1w	N	N	N	N	N	N	N
105	Bur26c	5426795	Opt	**5569268**	1w	1.0	—	1w	N	N	N	N	N	N	N
106	Bur26d	3821225	Opt	**3962219**	1w	1.0	—	1w	N	N	N	N	N	N	N
107	Bur26e	5386879	Opt	**5473127**	1w	1.0	—	1w	N	N	N	N	N	N	N
108	Bur26f	3782044	Opt	**3864508**	1w	1.0	—	1w	N	N	N	N	N	N	N
109	Bur26g	10117172	Opt	**10221265**	1w	1.0	—	1w	N	N	N	N	N	N	N
110	Bur26h	7098658	Opt	**7195094**	1w	1.0	—	1w	N	N	N	N	N	N	N

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
111	Had12	1652	Opt	**1652**	0.53	1.0	1660	0.02	1.0	**1652**	**0.03**	1.0	**1652**	**0.05**	1.0
112	Had14	2724	Opt	**2724**	3.75	1.0	**2724**	**0.05**	1.0	**2724**	**0.05**	1.0	**2724**	**0.05**	1.0
113	Had16	3720	Opt	**3720**	216.80	1.0	**3720**	**0.03**	1.0	**3720**	0.06	1.0	**3720**	0.06	1.0
114	Had18	5358	Opt	**5358**	32373.22	1.0	**5376**	**0.03**	1.0	**5358**	0.09	1.0	**5358**	0.09	1.0
115	Had20	6922	Opt	6990	1w	1.0	**6922**	**0.05**	1.0	**6922**	0.11	1.0	**6922**	0.11	1.0
116	Tho30	149936	Opt	153110	1w	1.0	152362	0.05	1.0	152570	0.42	1.0	**151378**	**0.39**	1.0
117	Tho40	240516	6.69%	249054	1w	1.0	248438	0.09	1.0	244856	1.09	1.0	**244744**	**1.03**	1.0
118	Tho150	8133398	6.30%	8496800	1w	1.0	**8237302**	8.11	1.0	N	N	N	N	N	N
119	Lipa20a	3683	Opt	**3683**	**547.86**	1.0	3767	1w	1.0	N	N	N	N	N	N
120	Lipa20b	27076	Opt	**27076**	**0.03**	1.0	**27076**	**0.03**	1.0	N	N	N	N	N	N
121	Lipa30a	13178	Opt	**13379**	**1w**	1.0	13401	1w	1.0	N	N	N	N	N	N
122	Lipa30b	151426	Opt	**151426**	**0.06**	1.0	155022	0.05	1.0	N	N	N	N	N	N
123	Lipa40a	31538	Opt	**31901**	**1w**	1.0	31920	1w	1.0	N	N	N	N	N	N
124	Lipa40b	476581	Opt	**476581**	0.22	1.0	**476581**	**0.09**	1.0	N	N	N	N	N	N
125	Lipa50a	62093	Opt	62681	1w	1.0	62681	1w	1.0	N	N	N	N	N	N
126	Lipa50b	1210244	Opt	**1217844**	**3.63**	1.0	1228858	0.02	1.0	N	N	N	N	N	N
127	Lipa60a	107218	Opt	108050	1w	1.0	108130	1w	1.0	N	N	N	N	N	N
128	Lipa60b	2520135	Opt	**2546546**	**6.17**	1.0	2566565	0.20	1.0	N	N	N	N	N	N
129	Lipa70a	169755	Opt	170935	1w	1.0	N	N	N	N	N	N	N	N	N
130	Lipa70b	4603200	Opt	4665667	1w	1.0	4663286	0.34	1.0	N	N	N	N	N	N
131	Lipa80a	253195	Opt	254673	1w	1.0	N	N	N	N	N	N	N	N	N
132	Lipa80b	7763962	Opt	7813682	5.13	1.0	7847338	0.55	1.0	N	N	N	N	N	N

QAPLIB				B&B			Simulated annealing			GRASP(dense)			GRASP(sparse)		
#	Name	Feas. solution	Gap	Feas. solution	Time	P.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r	Feas. solution	Time	p.r
133	Lipa90a	360630	Opt	362713	1w	1.0	N	N	N	N	N	N	N	N	N
134	Lipa90b	12490441	Opt	**12574357**	**168.31**	1.0	12636597	0.69	1.0	N	N	N	N	N	N
totall runtimes				891939.62			33.07			338.83			331.92		
average runtimes				15927.49			0.31			3.60			3.53		

This table's columns, contains five parts. The data from first part was taken form QAPLIB home page and the other parts are the results obtained from the algorithms. The first part contains 3 columns. First column is the name of instances, which is abbreviation of author and size of the instance. The second column shows the feasible solution for all the instances and the ones that reached to the optimal solution were marked with (Opt) in gap column and the rest are the best known solution. The gaps in between best known solution and the currently known best lower bound were shown in column number 3.

The other 4 parts of this table contain 3 columns each. The first one is the obtained feasible solution and the second column shows the runtime of the solution and the third column shows the Performance ratio. The best value among these four algorithms is bolded. The maximum runtime was set to 1 week and if during this 1 week the solution algorithm did not stop, it will drop the process and record the best know solution obtained by the algorithm(if there was not any solution, the square of the feasible solution marked by (—)) and the runtime of these processes are shown by (1w). The value of performance ratio is obtained by dividing the obtained value with mentioned method by the best known value in the QAPLIB web page. If this value is less than one, it means that the obtained value found is better than the value in the QAPLIB web page. Otherwise, the values which are closer to one are more reliable.

In some of the instances (for example all part b of the Taillard instances), the volume and amount of data matrices and the solution were overclocked and in these cases, the compiler was not able to solve the instance. This problem mostly happens in a few of

instances that were solved by GRASP algorithm. For example all of the Burkard instances, the Simulated Annealing method and both of GRASP algorithms can't find any solutions. The reason of this case is that the flow and distance matrices of Burkard instances are not symmetric and solving this kind of problem is difficult. The instances that faced this situation are marked with (N) and also for this kind of problem there is not any runtime and performance ratio.

According to the results obtained in this research, about 63% of the solutions in Branch and Bound Algorithm (B&B) were equal or closest to QAPLIB solutions which compared to the other algorithms is the highest percentage. It shows that the accuracy and efficiency of Branch and Bound algorithm is at least twice compared to the other algorithms. The instances which were solved by B&B method reached the objective function value. The reason for this is that unlike the other methods, B&B method is an exact algorithm and although it takes more time to solve the instances, it will give the objective function value.

Based on the average runtimes of each algorithm that led to a feasible solution without taking the instances that did not attain a feasible solution in one week and stopped into the consideration, although the best solutions were obtained by Branch and Bound algorithm, but this algorithm is dependent to a long runtime and has the longest runtime for solving the instances among the other algorithms. The average runtime for this algorithm is more than 265 minutes per instance. This amount of runtime can effect to decision for select the method for QAP.

Although the fastest solutions for instances are obtained by Simulated Annealing (with the average run time 0.3 second, but only 20% of the instances reach to best feasible solution by this algorithm. Both types of Grasp algorithms (Dense and Sparse) with the average runtime of 3.60 and 3.53 seconds and 35% and 33% of best feasible solutions, are of the most efficient methods for solving these problems. However, for all Bur and Lipa instances, because of the high volume of data, they do not have a good performance.

6.1 Analysis of Variance (ANOVA)

The analysis of variance (ANOVA) (Montgomery, 2001) of the objective function values is done to find out that is there any significant difference between the methods used to perform the computational experiments?

The methods used in this thesis to solve QAPs are considered as treatments (4 treatments) and the QAPs are considered as observations (134 observations) of the ANOVA test. Then the performance ratio is considered as the observed value of each observation for each treatment. The aim of the ANOVA test is to prove one of the following hypothesizes at the significant level of $\alpha = 0.1$:

$\boldsymbol{H_0}$: There is no difference in treatment means (the solution methods have the same performance).

$\boldsymbol{H_1}$: There is a significant difference in treatment means (the solution methods have different performances).

The table 2 contains the output of ANOVA test using SPSS software.

Table 2: The results of ANOVA

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	.042(a)	3	.014	2.569	.054
Intercept	451.586	1	451.586	82051.942	.000
Method	.042	3	.014	2.569	.054
Error	2.345	426	.006		
Total	463.565	430			
Corrected Total	2.387	429			

As the results of ANOVA shows at $\alpha = 0.1$ H_0 is rejected ($Sign. \cong 0.05$) and there is significant difference between the performance of the four used methods for QAP.

The Fisher Least Significant Difference (LSD) test also is performed for pairwise comparison of the methods. The summary of the LSD test ($\alpha = 0.1$) is reported in table 3.

Table 3: The results of LSD test.

(I) Method	(J) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
B & B	GRASP D	.009310	.0100122	.353	-.010369	.028990
	GRASP S	.012651	.0100122	.207	-.007029	.032330
	SA	-.013485	.0095775	.160	-.032310	.005340
GRASP D	B & B	-.009310	.0100122	.353	-.028990	.010369
	GRASP S	.003340	.0108212	.758	-.017929	.024610
	SA	-.022795(*)	.0104203	.029	-.043277	-.002313
GRASP S	B & B	-.012651	.0100122	.207	-.032330	.007029
	GRASP D	-.003340	.0108212	.758	-.024610	.017929
	SA	-.026135(*)	.0104203	.013	-.046617	-.005654
SA	B & B	.013485	.0095775	.160	-.005340	.032310
	GRASP D	.022795(*)	.0104203	.029	.002313	.043277
	GRASP S	.026135(*)	.0104203	.013	.005654	.046617

Because of the nature of the QAPs, the less obtained performance ratio is favored. Therefore based on table 3, significant negative mean difference is favored. The results prove the followings,

- Dense GRASP has significantly better performance than Simulated Annealing method.
- Sparse GRASP has significantly better performance than Simulated Annealing method.
- Both GRASP methods have better performance than Simulated Annealing and Branch and Bound but not significantly at $\alpha = 0.1$.
- Branch and Bound method performs better than Simulated Annealing but not significantly at $\alpha = 0.1$.
- Sparse GRASP has better performance than Dense GRASP but not significantly at $\alpha = 0.1$.

Chapter 7

CONCLUSION

The purpose of this research is to illustrate which of these four algorithms mentioned in the previous parts are more suitable for analyzing QABLIB's instances with respect to runtime.

According to the feasible solution that recorded in table 1, it can be concluded that, if the runtime is not a priority, the Branch and Bound algorithm is still one of the best algorithms for solving these kinds of optimization problems, but there is no guarantee when is the optimal solution achieve. And if the classification of these algorithms is made with respect to runtimes, the best method is Simulated Annealing with the average runtime of 0.3 seconds, but because this method is a heuristic method, there is no guarantee to reach the objective value.

The solution time can be extremely long for large problem instances. Therefore it is possible that within a week the optimal solution is not achieved. Thus the B&B algorithm can be understood not only exact but heuristic method as well.

Obviously if a more powerful processor and greater volume of Ram is used for this research, it will lead to a better runtimes.

REFERENCES

Ahuja, R., Orlin, J., & Tiwari, A. (2000). A greedy genetic algorithm for the quadratic assignment problem. *Computers and Operations Research, 27(10),* 917-934.

Arkin, E., Hassin, R., & Sviridenko, M. (2001). Approximating the maximum quadratic assignment problem. *Information Processing, 77,* 13-16.

Baykasoglu, A. (2004). A meta-heuristic algorithm to solve quadratic assignment formulations of cell formation problems without presetting number of cells. *Journal of Intelligent Manufacturing, 15(6),* 753-759.

Bazaraa, M., & Sherali, H. (1979). New approaches for solving the quadratic assignment problem. *Operation Research Verfahren, 32,* 29-46.

Blanchard, A., Elloumi, S., Faye, A., & Wicker, N. (2003). A cutting algorithmfor the quadratic assignment problem. *INFOR, 41,* 35-49.

Burkard, R., & Bonniger, T. (1983). A heuristic for quadratic Boolean programs with applications to quadratic assignment problems. *European Journal of Operation Research, 13,* 374-386.

Christofides, N., & Benavent, E. (1989). An exact algorithm for the quadratic assignment problem. *37*, 760-768.

Clausen, J. (1999, March 12). Branch and Bound Algorithms Principles and Examples. *Department of Computer Science, University of Copenhagen, Universitetsparken 1, DK-2100*.

Cung, V.-D., Mautor, T., Michelon, P., & Tavares, A. (1997). A scatter search based approach for the quadratic assignment problem. *Proceedings of IEEE International Conference on Evolutionary Computation*, 165-169.

Deinko, V., & Woeginger, G. (2000). A study of exponential neighborhoods for the traveling salesman problem and for the quadratic assignment problem. *Mathematical Programming, 78*, 519-542.

Dickey, J., & Hopkins, J. (1972). Campus building arrangement using Topaz. *Transportation Research, 6*, 59-68.

Drezner, Z., & Marcoulides, G. (2003). A distance-based selection of parents in genetic algorithms.Kluwer Academic Publishers. 257-258.

Elshafei, A. (1977). Hospital layout as a quadratic assignment problem. *Operations Research Quarterly, 28(1)*, 167-179.

Eschermann, A., & Wunderlich, H. (1990, june 26-28th). Optimized synthesis of self-testable finite state machines. *In 20th International Symposium on Fault-Tolerant Computing (FFTCS 20).*

Feo, T., & Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization, 6*, 109-133.

Forsberg, J., Delaney, R., Zhao, Q., Harakas, G., & Chandran, R. (1994). Analyzing lanthanide-included shifts in the NMR spectra of lanthanide (III) complexes derived from 1,4,7,10-tetrakis (N,N-diethylacetamido)-1,4,7,10-tetraazacyclododecane. *Inorganic Chemistry, 34*, 3705-3715.

Francis, R., & White, J. (1974). Facility Layout and Location: An Analytical Approach. *Prentice-Hall, Englewood Cliffs, NJ*.

Gavett, J., & Plyter, M. (1994). The optimal assignment of facilities to locations by Branch and Bound. *Operation Research, 42*, 860-878.

Gendronu, B., & Grainic, T. (1994). Parallel branch and bound algorithms: Survey and synthesis. *Technical report, Center for Research on Transportation*.

Gilmore, P. (1962). Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM Journal on Applied, 10*, 305-313.

Gilmore, P. (1962). Optimal and suboptimal algorithms for the quadratic assignment problem. *J. SIAM, 10*, 305-313.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Sience, 8*, 156-166.

Gutin, G., & Yeo, A. (2002). Polynomial approximation algorithms for TSP and QAP with a factorial domination number. *Discrete Applied Mathematics, 119*, 107–116.

Hadley, S., Rendl, F., & Wolkowicz, H. (1992). A new lower bound via projection for the quadratic assignment problem. *Mathematics of Operations Research, 17*, 727-739.

Hubert, L., & Schulz, J. (1976). Quadratic assignment as a general data analysis strategy. *British Journal of Mathematical Psychology, 29*, 190-241.

Jingwei, Z., Ting, R., Husheng, F., Jinlin, Z., & Ming, L. (2012). Simulated annealing ant colony algorithm for Quadratic Assignment Problem. *International Conference on Natural Computation*, 789-793.

Junger, M., & Kaibel, V. (2000). On the SQAP-polytope. *SIAM journal on Optimization, 11*(2), 444-463.

Junger, M., & Kaibel, V. (2001). The QAP-polytope and the star transformation. *Discrete Applied Mathematics, 111*(3), 283-306.

Karisch, S., & Rendl, F. (1995). Lower bounds for the quadratic assignment problem via triangle decompositions. *Mathematical Programming, 71(2)*, 137-152.

Kaufman, L., & Broeckx, F. (1978). An algorithm for quadratic assignment problems using Bender's decomposition. *European Journal Operation Research, 2*, 204-211.

Kirkpatrick, S., Gelatt Jr, C., & Vecchi, M. (1983). Optimization by Simulated Annealing. *Science, 220*, 671-680.

Koopmans, T., & Beckmann, M. (1957). Assignment problems and the location of economic activities. *Econometrica, 25*, 53-76.

Krarup, J., & Pruzan, P. (1978). Computer-aided layout design. *Mathematical Programming Study, 9*, 75-94.

Land, A. (1963). A problem of assignment with interrelated costs. *Operation Research Quarterly, 14*, 185-198.

Lawler, E. (1963). The quadratic assignment problem. *Management Science, 9*, 586-599.

Li, Y., Pardalos, P., & Resende, M. (1994). Fortran subroutines for approximate solution of dense quadratic assignment problems using GRASP. *Technical report, AT&T Bell Labratories*.

Mills, P., Tsang, E., & Ford, J. (2003). Applying an extended guided local search to the quadratic assignment problem. *Annals of Operation Research, 118,* 121-135.

Miranda, G., Luna, H., Mateus, G., & Ferreira, R. (2005). A performance guarantee heuristic for electronic components placement problems including thermal effects. *Computers and Operations Research, 32*, 2937-2957.

Mirchandani, P., & Obata, T. (1979, May). Locational decisions with interactions between facilities: the quadratic assignment review. *Working paper Ps-79-1,Rensselaer Polytechnic Institute*.

Misevicius, A. (2005). A tabu search algorithm for the quadratic assignment problem. *Computational Optimization and Applications, 30(1),* 95-111.

Montgomery, D. C. (2001). *Design and analysis of experiments* (5th ed.). John Wiley.

Nissen, V., & Paul, H. (1995). A modification of threshold accepting and its application to the quadratic assignment problem. *OR, 17*, 205-210.

Nugent, C., Vollmann, T., & Ruml, J. (1969). An experimental comparison of techniques for the assignment of facilities to locations. *Journal of Operation Research, 16,* 150-173.

Oliveira, C., Pardalos, M., & Resende, M. (2004). GRASP with path relinking for the quadratic assignment problem. *In: Experimental and Efficient Algorithms, Third International Workshop (WEA 2004),* 356-368.

Padberg, M., & Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman. *SIAM Review, 33,* 60-100.

Pardalos, P., & Crouse, J. (1989). A parallel algorithm for the quadratic assignment problem. *In Proceeding of the Supercomputing 1989 Conference,* 351-369.

Peng, T., Huanchen, W., & Dongme, Z. (1996). Simulated annealing for the quadratic assignment problem: A further study. *Computers and Industrial Engineering, 31(3-4),* 925-928.

Pollatschek, M., Gershoni, N., & Radday, Y. (1976). Optimization of the typewriter keyboard by simulation. *Angewandte Informatik, 17,* 438-439.

Rabak, C., & Sichman, J. (2003). Using A-Teams to optimize automatic insertion of electronic components. *Advanced Engineering Informatics, 17 (2),* 95-106.

Resende, M., Ramakrishman, K., & Drezner, Z. (1995). Computing lower bounds for the quadratic assignment problem with an interior point algorithm for linear programming. *Operations Research, 43*, 481-791.

Roucairol, C. (1987). A parallel branch and bound algorithm for the quadratic assignment problem. *Discrete Applied Mathematcs, 18*, 211-225.

Singh, S., & Sharma, P. (2006). Two-level modified simulated annealing based approach for solving facility layout problem. *International Journal of Production Research. Department of Industrial and Management Engineering, Indian Institute of Technology Kanpur*.

Skoin-Kapov, J. (1990). Tabu search applied to the quadratic assignment problem. *ORSA Journal on Computing, 2(1)*, 33-45.

Steinberg, L. (1961). The backboard wiring problem: A placement algorithm. *SIAM Review, 3*, 37-50.

Thonemann, U., & Bolte, A. (1994). An improved simulated annealing algorithm for the quadratic. *Working paper, School of Business, Department of Production and Operations Research*.

Urban, T. (1998, 01). Solution procedures for daynamic facility layout problem. *Annals of operation research, 76*, 323-342.

Wang, S., & Sarker, B. (2005). Solving facility layout problem using an improved genetic algorithm. *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences E88A (2),* 606-610.

Wess, B., & Zeithofer, T. (2004). On the phase coupling problem between data memory layout generation and address pointer assignment. *Lecture Notes in Computer Science 3199,* 152-166.

Wilhelm, M., & Ward, T. (1987). Solving quadratic assignment problems by simulated annealing. *IIE Transaction, 19/1*, 107-119.

Zhoa, Q., Karisch, S., Rendl, F., & Wolkowicz, H. (1996). Semidefinite programming relaxations for the quadratic assignment problem. . *Technical Report DIKU-TR-96/32, Department of Computer Science, University of Copenhagen*.