AN ABSTRACT OF THE THESIS OF

Thanat Kriausakul for the degree of Master of Science in Industrial Engineering presented on November 16, 2001. Title: A Distance and Shape-Based Methodology for the Unequal Area Facility Layout Problem.

Abstract approv Redacted for Privacy

Rasaratnam Logendran

Significant improvements in production effectiveness have resulted from implementing cellular manufacturing systems (CMS). Following the cell formation, an important issue that needs to be addressed is the unequal cell (or department/facility) layout problem, which is the sub-issue in the CF problem. The work reported in this thesis illustrates the assignment of unequal cell locations in dealing with the known traffic movements on a shop floor. In addition, this research addresses the impact of the geometry or shape of the department as an important design factor in the unequal area facility layout problem, an issue that has not been addressed by the previous researchers.

The problem is formulated as a mixed-binary non-linear programming model and is proven to be NP-hard in the strong sense. Due to its computational complexity, a higher-level heuristic, based on a concept known as tabu-search, is proposed to efficiently solve the problem. Six different versions of the tabu search-based heuristic algorithm are tested on three different problem structures.

The results obtained from performing the experiment concluded that the tabu search-based heuristic using short-term memory and variable tabu-list sizes is preferred over other heuristics as the problem size increases. The performance comparison between the current and the previous research shows that the solution obtained for the well-known problems in this research are better than that obtained in the past.

A Distance and Shape-Based Methodology for the Unequal Area Facility Layout Problem

by

Thanat Kriausakul

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirement for the
degree of

Master of Science

Presented November 16, 2001
Commencement June 2002

Master of Science thesis of Thanat Kriausakul presented on November 16, 2001

APPROVED:

Redacted for Privacy

Major Professor, representing Industrial Engineering

Redacted for Privacy

Head or Chair of Department of Industrial and Manufacturing Engineering

Redacted for Privacy

Dean of Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for Privacy

Thanat Kriausakul, Author

ACKNOWLEDGMENT

I would like to express my sincerest gratitude to Dr. Rasaratnam Logendran, my major professor, for his guidance, advice, encouragement, and patience throughout the pursuit of my research. Without his help, this thesis would not have been possible.

My sincere thanks to all of my committee members: Dr. David Birkes, my minor professor from the Department of Statistics, Dr. Jeffrey Woldstad, my committee member from the Department of Industrial & Manufacturing Engineering, and Dr. David Rogge, my Graduate Council Representative from the Department of Civil, Construction, and Environmental Engineering.

My sincere thanks go to many of my friends who helped me out with this research. Wimol Lertwiwattrakul has always given me valuable advice and encouragement. Ploenpun Sakrittichai helped me with the programming language and given me encouragement. I would like to thank those individuals for their help whose names have not been mentioned here.

Finally, I would like to express my deepest gratefulness to my parents, my uncles, and all of my friends for their support and encouragement. This thesis has been an invaluable experience in my life.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

Figure Page

LIST OF TABLES

LIST OF TABLES (CONTINUED)

LIST OF TABLES (CONTINUED)

LIST OF APPENDICES

LIST OF APPENDIX FIGURES

LIST OF APPENDIX TABLES

DEDICATED

TO

MY PARENTS

A DISTANCE AND SHAPE-BASED METHODOLOGY FOR THE UNEQUAL AREA FACILITY LAYOUT PROBLEM

1. INTRODUCTION

One of the most important issues that must be resolved in manufacturing systems design is the assignment of facilities to locations. Plant layout and material handling affects the productivity and the profitability of a company more than almost any other major corporate decision. Tompkins and White (1984) and Sule (1988) emphasized this fact; the authors pointed out that 20-50% of the total operating expenses in manufacturing are attributed to material handling and layout related costs. Use of effective methods for facilities layout can reduce these costs by at least 30%.

Facility layout is the organization of the company's physical facilities to promote the efficient use of equipment, material, people, and energy. Material handling is defined simply as moving material. Material handling has affected working people more than any other area of work design (Fred 1993).

Facility layout problem (FLP) can also be regarded as a sub-issue of cellular manufacturing (CM), which is regarded as an application of Group Technology (GT). GT deals with the identification of part families, machine groups, and allocation of part families and machine groups to cells or vice versa. In this research, the facility layout problem focuses on physically locating cells in a floor plan. Several benefits have been attributed to implementing CM systems including reduced setup times, reduced queue times, reduced production lead-time and reduced work in progress. Previous studies have reported considerable improvement in production effectiveness achieved by the application of the methodology of facility layout in a factory (Burbidge and Dale 1984, Gallagher and Knight 1986). In a past survey of cellular manufacturing systems in the US (Wemmerlov and Hyer 1989), average reductions can be achieved in throughput by 45.6%, in WIP inventory by 41.4%, in materials handling by 39.3%, and in setup time by 32.0%. These results are consistent with those reported earlier in Burbidge (1979).

During the past two decades, a considerable amount of research devoted to this issue has been published. Most researchers included equal area and unequal area departments in their investigations. Since Koopmans and Beckman (1957) proposed the layout problem, which could be modeled as an Quadratic assignment problem (QAP) to solve the equal area facility layout, more than 50 different scientific papers have been published. However, when departments have unequal areas of physical space, the QAP cannot solve the unequal area facility layout problem. Armour and Buffa (1963) proposed the unequal area layout problem and applied the pairwise exchange method to solve it. After their work, few researchers focused on the unequal area layout problem. The recent work reported by Bazaraa (1975), van Camp *et al.* (1991), Kar Yan Tam and Shih Gong Li (1991), Kar Yan Tam (1992), and T. Hon-iden (1996) successfully solved the unequal area facility layout problem. Up to now only Hon-iden proposed a comprehensive framework for the design of facility layout, which is included the geometry or shape of the department. His work included the shape or geometric constraints in the evaluation of the initial solution.

This research paper attempts to address several objectives that are of major importance to the design of unequal area facility layout problem by employing the combinatorial search technique called “tabu search”. It is motivated by the lack of investigation from past facility layout researchers in unequal area facility layout problems (FLP). First of all, all past FLP research focused on assigning departments to locations based only on a distance measure, which is represented by the traffic flow. Therefore, this research proposes to include the geometric factor or shape factor along with the distance factor in the objective function. Second, previous FLP researchers had never considered an initial solution technique that is based on the shape factor. Only Hon-iden (1996) proposed the use of shape factor in his initial solution finding mechanism. Though the distance measure and shape measure are included in his initial solution finding mechanism, the shape measure is not included in the solution procedure for finding the objective function. In this research both distance and shape measures are simultaneously considered for finding a better initial solution, in the hope that they will lead to better final solution. Not only are both distance and shape measures included in the initial solution finding mechanism, but they are considered in the objective function also. Hon-iden (1996) was the first researcher who considered the shape measure in the step of finding the initial solution of the unequal area layout. However, in his work, the inflexibility in the size of each department restricts the possibility to find the better solution. In other words, the department sizes (widths and heights) are determined before the search begins. From the lack of flexibility in department sizes of Hon-iden's work, three categories of the department sizes are proposed and evaluated in both of the initial solution mechanism and the solution procedure for finding the objective function.

Besides the two major issues described in the previous paragraph, this research includes another important factor in the design of FLP, namely the aspect ratio. The aspect ratio roughly determines the shape of an individual department. Thus, a mathematical model in this research is developed to include this factor. Using the insight gained from the model, heuristic algorithms are developed to solve the FLP.

Including this chapter, this thesis contains eight chapters. The remaining chapters are organized as follows. Chapter 2 reviews several approaches used in solving the FLP. Chapter 3 presents major FLP issues considered in this research. Chapter 4 describes the mathematical formulation used to represent the FLP. In Chapter 5, the proposed heuristic algorithms and its application are presented in detail. In Chapter 6, the procedure used in assessing the quality of the heuristic algorithms is given. Chapter 7 presents the experimental design used in comparing the different heuristic algorithms developed in this research. Finally, conclusions and recommendations for future research are presented in Chapter 8.

2. LITERATURE REVIEW

Determining the physical organization of a production system is defined to be the facility layout problem. This well studied combinatorial optimization problem arises in a variety of production facilities, including service and communications settings. However, the focus of this research is on manufacturing facility layout.

The main functions of facility layout design are to allocate departments under the activity interrelationships and to optimize their space requirements. The objective is to design an efficient arrangement of space required by a department into an integrated whole, which is called area allocation. The dimensions and properties of a department are given to determine the interrelationship cost between all pairs of departments. A satisfactory layout is then usually selected under the constraint of minimizing interrelationship costs.

In general, the facility layout problem has been formulated as a quadratic assignment problem (QAP) (Koopmans and Beckman 1957, Lawler 1963, Peirce and Crowston 1971, Bazaara 1975, Burkard and Stratmann 1978, Kusiak and Heragu 1987, and Francis and White 1992). The quadratic assignment problem is modeled for equal area layout; thus the QAP can no longer consider solving the unequal area problems.

Over the years, several researchers have proposed classification of heuristic methods used in the facility layout problem (Kusiak and Heragu 1987). These classifications are useful to the researchers by providing them with an overall understanding of heuristic methods. In this classification, the work associated with the layout problem was divided into four groups: the constructive algorithm, the iterative improvement algorithm, the hybrid algorithm, and the graph theoretic algorithm. A short description for each category is presented below:

1. Constructive algorithm

   In a survey, Moore (1974) found that there were twice as many construction algorithms as improvement algorithms. In a constructive algorithm, the departments are assigned to a site, one at a time, until the complete layout is obtained. Many researchers have applied the constructive algorithm to solve the facility layout problem (Hillier and Connors 1966; Seehof and Evans 1967; Lee and Moore 1967; Zoller and Adendorff 1972; Neghabat 1974; Block 1978; Heragu and Kusiak 1986).

2. Iterative improvement algorithm

   In the improvement algorithm, there is always an initial solution, which is often randomly generated. Based on this initial solution, systematic exchanges between departments are made and the results are evaluated. The exchange which produces the best solution is retained and the procedure continues until the solution cannot be improved any further. Hence, the solution quality of improvement algorithms depends upon the initial layout evaluated. The improvement algorithm was used by Armour and Buffa 1963; Buffa 1964; Hillier 1963; Hillier and Connors 1966; Vollman 1968; Nugent 1968; Khalil 1973; Tompkins and Reed 1976; and Picone and Wilhelm 1984.

3. Hybrid algorithm

   Bazaraa and Kirca (1983) classified algorithms, which have the characteristics of optimal and suboptimal algorithms as hybrid algorithms.

For example, Burkard and Stratman (1983) proposed a heuristic algorithm, which uses concepts of the branch and bound algorithm (optimal algorithm), and an improvement algorithm. An initial solution is obtained using a branch and bound algorithm, and the solution is improved by using an improvement algorithm.

4. Graph theoretic algorithm

   Graph theoretic algorithms identify maximal planar subgraphs of a weighted graph, which show the relationship between the departments. A node in a graph represents each department. The area and shape of the departments are ignored at the beginning of the algorithm. Additional details about this algorithm can be found in Foulds 1991; and Hassan and Hogg 1987.

As mentioned in the paragraph before, almost all researchers who applied the above heuristic methods can only handle the equal area facility layout problems. The equal area problems are impractical for industry application. The area of each department is not always equal in real life. The unequal area facility layout problem is more relevant in industry applications and is a challenging topic that it is chosen to be studied in this research. The hybrid algorithm is chosen because a better initial solution leads to the better improvement of the final solution. The most recent development of search heuristics is in the area of the branch and bound approach, nonlinear optimization approach, simulated annealing, genetic algorithms, and the clustering approach. These approaches were used, respectively, by Bazaraa (1975), van Camp *et al.* (1991), Tam (1992), Tate and Smith (1995), and Hon-Iden (1996). Hon-Iden (1996) introduced the shape factor used in the clustering procedure to define the initial solution, and that initial solution led to identifying a better final solution compared with other researchers. With a strategic and intelligent application of the shape factor, this research can prove to be a very efficient and effective for solving the unequal area layout problem. The most recent development of search heuristics is in the area of simulated annealing, tabu search, and genetic algorithms. The tabu search is the only heuristic that has not been used in the area of the facility layout problem. Tabu search has been used to obtain optimal and near optimal solutions for a wide variety of applications. Some applications of tabu search have included scheduling, transportation network design, layout and circuit design problems, telecommunications, probabilistic logic and expert systems, neural network pattern recognition, and others (for a list of references and brief exposition of such application papers, see Glover and Laguna 1992). Although it is still in an early stage of development, tabu search has enjoyed a number of successes. In a variety of problem settings mentioned above, it has found solutions superior to the best previously obtained by alternative methods.

This research focuses on applying tabu search to solve the unequal area layout problem. The tabu search is a higher-level heuristic procedure for solving optimization problems, which is designed to guide other methods (or their component processes) to escape the trap of local optimality (Glover 1990). It was pioneered by Fred Glover (1986) and presented in detail in Glover (1989, 1990), and Glover and Laguna (1992). Tabu search has been used to obtain optimal and near optimal solutions for a wide variety of applications. The details about the tabu search-based heuristic algorithm developed in this research are provided in Chapter 5.

3. PROBLEM STATEMENT

The facility layout problem is concerned with the location and arrangement of departments, cells or machines on a plant or office floor. Because of the geometric and combinatorial aspects of the problem, the facility layout problem is a computationally difficult problem. For years, the research on the facility layout problem has progressed significantly. It started in 1957 when Koopmans and Beckman proposed the Quadratic assignment problem (QAP) to solve the equal area facility layout. Since this initial work, a variety of contributions have been published (Russell, 1996). Most of these published works did not consider either one or more of the important factors such as unequal area departments, shape of departments, and shape cost. Hence, these past researches fail to reflect the needs of real manufacturing systems that the departments would not always be equal areas. This research provides valuable insights to the unequal area facility layout problem. It has also laid the foundation for the future development of a comprehensive unequal area facility layout approach.

In the layout problem, the layout's efficiency is typically measured in terms of material handling cost. The material handling cost is defined as the distance between a pair of department locations multiplied by the flow matrix (interaction or traffic flow) and cost of transportation. Thus, the total material handling cost is the sum of each individual material handling cost required by every pair of department locations. The total material handling cost is used here because it approximately measures the effectiveness of grouping the departments that have the desirability of closeness. It is also directly used as a throughput measure of the manufacturing process. Clearly, the smaller the total material handling cost, the quicker is the transportation between department locations, and as a result the sooner the tasks get completed.

The choice of the criterion to be used in identifying the "best" solution from among several alternative solutions is not an easy one in the case of facility location and design problems. Perhaps the most popular criterion used is to minimize some function of distance traveled. Within an industrial setting, it is argued that minimizing distance will minimize material handling cost. In particular, it is often assumed that the material handling cost is the only significant factor and the material handling cost is linear functions of distance and flow volumes. But in this research, one of the design factors that is not included in the previous research is taken into consideration. This factor is called "shape factor" which is explained in detail in the next paragraph.

Almost all of the published works used minimizing distance as a matrix for minimizing the material handling cost of the layout. Only Hon-Iden's work (1996) showed the effect of the department shapes. He developed a new coefficient that represents the geometric flexibility of the department shapes. Although his work proposed the motivation for the shape factor in the layout problem, the shape factors were applied for the step that associates with the initial solution only. In this research, the shape factor is developed in order to add in the second term of the objective function. Not only is it included in the objective function, but it is used in the initial solution finding mechanism also. The shape factor is represented by a "normalized weight". It is the parameter that represents the contribution of the distance measurement and shape measurement. For the purpose of application, a normalized weight of 80% for distance and 20% for shape is used in this study. It means that 80% of the contribution to the objective function is made by the material handling cost or distance cost, and the remaining 20% of the contribution is made by the shape cost.

Numbers are used to represent the configuration of departments. The bay structure is introduced in this research in order to guide or direct the departments to configure in the floor plan. From the concept of bay structure, the floor plan can be divided in the one direction (vertical) into bays of varying width, and the departments of equal width but different height can be placed in each bay. In other words, the bay structure makes an effort to group the high interactions (distance measure and shape measure) of departments in the same bay. The higher the interactions, the closer the departments should be located within the same bay. The advantages for the bay structure are not only for grouping the departments that have the high interaction in the same bay, but also avoiding the intra traffic between bay-to-bay (might cause accidents, inconvenienced movements, complicated traffic flows, etc.). Thus, the traffic flows in the layout when applying the bay structure prefer travel within the bay rather than between the bays. The amount of traffic flows in the same bay is always much more than the bay-to-bay.

Before presenting the next chapters, there are three important issues that have to be presented: (3.1) Distance measurement methods (3.2) Aspect ratios and (3.3) Total area of the floor plan.

3.1 Distance Measurement Methods

In the layout problem area, the most common distance measurement has three methods, which are defined as follows:

1. Rectilinear distance (Manhattan distance/ City block distance)

$$d_{ij} = | x_i - x_j | + | y_i - y_j |$$

2. Euclidian distance

$$d_{ij} = [(x_i - x_j)^2 + (y_i - y_j)^2]^{1/2}$$

3. Squared Euclidean distance

$$d_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2$$

where (x_i , y_i) and (x_j , y_j) are the coordinates of the cross-sectional center of departments *i* and *j*, respectively.

The rectilinear distance measure is often used for factories and American cities, which are laid out in the form of a rectangular grid. For this reason it is sometimes called the Manhattan distance measure. The Euclidean distance measure is used where genuine straight-line travel is possible. The squared Euclidean distance measure is used if straight-line travel is possible, and when the designers wish to discourage excessive distances. Some researchers applied Euclidian distance in their works, but when main aisles are taken into consideration in the plant layout, rectilinear distance is more reasonable in the detailed layout. This research assumes the measurement distance is rectilinear.

3.2 Aspect Ratios

The aspect ratio (a_i) is another factor that restricts the shape and size of each department. The aspect ratio a_i of department i is defined as:

$$a_i = \frac{\textit{Height of rectangular area of department } i}{\textit{Width of rectangular area of department } i} = \frac{h_i}{w_i}$$

The orientation of the department can be classified into two categories: (1) free orientation and (2) fixed orientation. A free orientation department allows both vertical and horizontal orientation. Thus, the aspect ratio of a free oriented department can either be a_i or $1/\ a_i$. A fixed oriented department allows only vertical (or horizontal) orientation. Thus, its allowable aspect ratio range is simply $a_{i.}$ This research applies free orientation to make the problem more flexible.

3.3 Total Area of the Floor Plan

The space requirement in the floor plan would need to be determined in a very early step of the unequal area layout design problem. In fact, the standard floor plan is cheaper than custom-designed floor plan. It would be too expensive to build an exactly square footage floor (Fred E. Meyers, 1993). The standard floor plan could be 25’ × 50’, 40’ × 40’, 50’× 50’, and 100’× 100’. This refers to column spacing, so a 25’ × 50’ floor would come in multiples of 25’ in height and 50’ in width, thus resulting in a rectangular floor. A 1:2 height to width ratio is a very desirable shape for a floor plan because it provides ease of material flow and accessibility. Although any ratio of height to width is possible (even square), in this research a height to width ratio of 1:2 is used.

$$H: W = 1:2$$

The total space required is the sum of all departmental areas. In this research, it is assumed that 200 percent of the total space required is the total area in the floor plan. The 200 percent is considered because it allows the extra space for an aisle, work in progress, and a small amount of miscellaneous extra room is added in the floor plan.

Clearly, both distance measure and shape measure are two of the most significant factors that must be considered in the investigation of the unequal area facility layout problem. Consequently, the objectives of this research can be stated as:

(i) To develop a mathematical model that is capable of addressing the needs of an unequal area facility layout problem in the presence of design factors including unequal area departments, shape of departments, and shape cost.

(ii) To develop an efficient solution algorithm that can be used to solve the model specified in item (i). The algorithm should be capable of identifying a quality solution within a reasonable computational time, even on large industrial-size problems.

In the next chapter, the mathematical model for this problem is formulated as a mixed binary non-linear programming model. Its objective function focuses on minimizing the total cost, which is evaluated as the sum of the distance cost and shape cost.

4. MODEL DEVELOPMENT

4.1 Background

The model developed in this research uniquely addresses the issues concerning the facility layout problem, which consists of two sub problems (1) laying out machines within a cell, and (2) laying out cells on the floor plan. While attention has been directed to the machine layout problem recently (Heragu and Kusiak 1988,1990), little has been done on the latter problem. The focus of this research is on the laying out cells on the floor plan. The model is formulated as a nonlinear programming model. The objective function focuses on minimizing the total material handling cost and the cost associated with the impact of shape on the layout. The constraints consist of equations or inequalities that deal with the major issues described in the previous chapter.

The material handling cost was explained in numerous research papers including textbooks (Meyers, 1993). However, the shape cost, associated with the impact of the unequal size or dimension of each department, is an additional cost that previous researchers have not considered. It is called a "pseudo cost" or a pleasing cost. The shape cost, in the second term of objective function, is totally different from the distance cost. The distance cost is evaluated from the distance between pairs of departments and the total number of unit loads, but the shape cost is evaluated from the unequal size of any pair of departments. In other words, the shape cost is not a real cost. For example, when a worker walks from one department to the other department with different sizes of departments both the distance cost and the pseudo cost must be determined, and the former can be generally calculated. The different sizes of the departments involved result in the zigzag aisles that the worker needs to walk or drive to the destination with many turns instead of straight aisles and less turns in the equal area departments. The unequal size of departments are much more complicated than the equal size of departments; therefore, the pseudo cost, must be added in the objective function.

The pseudo cost or shape cost is the cost that can be assessed from many issues, which are:

- The convenience of workers. The allocation of the different size of the departments will make the worker feel inconvenienced to walk or work in the zigzag routing.
- The safety of employees. Aisle of the different size of the departments can cause safety problems. For example: many turns or curves will be created when the different sizes of departments are introduced.
- Visual distractions. It will not be easy to find a tool or equipment in the unequal partitions or different size of departments. There are some obstructions or blind points that the workers cannot find a particular tool.
- Pleasing atmosphere. The impressions or opinions of the visitors or workers who have to be involved in the unequal size of departments.
- The productivity of the company. As a result of the above issues, the production time will be increased because of the slow traffic between departments.

There are many disadvantages of the different size of the department areas as described above; the issues associated with unequal areas cannot be neglected. Some designers might try to modify the unequal shapes to equal shapes of the departments. However, the modification of the unequal areas to equal area departments do not alleviate the disadvantages of unequal areas because the modification is going to increase the size of the smaller departments to be equal to the larger sizes, which it is going to create too many unusable areas. The investment for the extra area (unusable areas for modification) is much more expensive than the consideration of unequal area departments. Thus, the facility layout designers have to encounter the unavoidable unequal area problems.

The assumptions and notations used in the development of an appropriate mathematical model are stated below. Following this, a mathematical model, which includes the objective function and constraints, is presented. Finally, the description of the model as well as its computational complexity is also given.

4.2 Assumptions

(1) Facility (or department or cell) shape is rectangular.

(2) Rectilinear distance (or Manhattan or City block distance) is applied in this research.

(3) There is only one floor considered.

(4) Routing or flow matrix is known.

(5) Aspect ratio for each cell (department) is known.

4.3 Notation

Indices

N = Number of cells to be located on the floor

i and j = Cell or department index ($i, j \in 1,\ldots,N$, and $i \neq j$)

Parameters and coefficients

C_{ij} = Cost per unit distance per unit load (or unit traveling)

D_{ij} = Cost per unit length of unequal size between cell i and j in both x-direction and y-direction

A_i = Area of cell i

a_i = Aspect ratio of cell i

X = Width of the total area or floor

Y = Height of the total area or floor

a_{iL} and a_{iU} = Lower and upper bounds of a_i

□ = Normalized weight

M = A very large number

f_{ij} = Flow matrix (or From-to chart) between cell i and j

R = A real number

$$P = \begin{cases} 1 \text{ if } \sqrt{\dfrac{A_i}{a_{iL}}} \geq \sqrt{A_i a_{iU}} \\ 0 \quad \text{otherwise} \end{cases}$$

Variables

(x_i, y_i) = Coordinates of the cross-sectional center of cell i (department)

(w_i, v_i) = Coordinates of the lower-left corner of cell i

x_{ij} = Distance between cell i and cell j in x-direction

y_{ij} = Distance between cell i and cell j in y-direction

z_i = Width of the cell i

INT_{ij} = Binary variables

4.4 Mathematical Model

Let $\Pi = \{(i, j) \mid i = 1, \ldots, N\text{-}1;\ j = i+1, \ldots, N\}\}$

$\Pi_1 = \{(i \mid i = 1, \ldots, N\}$

$$\text{Minimize}\quad \alpha[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} f_{ij}\, C_{ij}\,(x_{ij} + y_{ij})] + (1\text{-}\alpha)[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} (|\frac{A_i}{z_i} - \frac{A_j}{z_j}| + |\, z_i\text{-}z_j\,|)^{*}D_{ij}] \quad ..(0)$$

Subject to:

$$x_{ij} \geq (w_i\text{-}w_j) + 0.5(z_i\text{-}z_j) \qquad (i,j) \in \Pi \qquad ....(1)$$

$$x_{ij} \geq (w_j\text{-}w_i) + 0.5(z_j\text{-}z_i) \qquad (i,j) \in \Pi \qquad ....(2)$$

$$y_{ij} \geq (v_i\text{-}v_j) + 0.5(\frac{A_i}{z_i} - \frac{A_j}{z_j}) \qquad (i,j) \in \Pi \qquad ....(3)$$

$$y_{ij} \geq (v_j\text{-}v_i) + 0.5(\frac{A_j}{z_j} - \frac{A_i}{z_i}) \qquad (i,j) \in \Pi \qquad ....(4)$$

$$x_{ij}\ \text{-}z_i/2\ \text{-}z_j/2 \geq \text{-}INT_{ij}M \qquad (i,j) \in \Pi \qquad ....(5)$$

$$y_{ij}\text{-}\ A_i/2z_i - A_j/2z_j \geq \text{-}(1\text{-}INT_{ij})M \qquad (i,j) \in \Pi \qquad ....(6)$$

$$z_i \leq \sqrt{\frac{A_i}{a_{iL}}} + M(1-P) \qquad i \in \Pi_1 \; ....(7)$$

$$z_i \geq \sqrt{\frac{A_i}{a_{iU}}} - M(P) \qquad i \in \Pi_1 \; ....(8)$$

$$z_i \leq \sqrt{A_i a_{iU}} + M(P) \qquad i \in \Pi_1 \; ....(9)$$

$$z_i \geq \sqrt{A_i a_{iL}} - M(1-P) \qquad i \in \Pi_1 \; ....(10)$$

$$0 \leq w_i \leq X - \max \{ \sqrt{\frac{A_i}{a_{iL}}}, \sqrt{A_i a_{iU}} \} \qquad i \in \Pi_1..........(11)$$

$$0 \leq v_i \leq Y - \max \{ \sqrt{A_i a_{iU}}, \sqrt{\frac{A_i}{a_{iL}}} \} \qquad i \in \Pi_1..........(12)$$

Case 1:

$$A_i^2 z_j^2 - A_j^2 z_i^2 \leq 0 \qquad .....(a)$$

Case 2:

$$z_j^2 \leq \frac{A_j^2}{a_{iL} A_i} \qquad .....(b)$$

$$R \leq a_{iU} A_i - a_{jL} A_j \qquad .....(c)$$

$$\frac{A_i^2}{z_i^2} - \frac{A_j^2}{z_j^2} - R \leq 0 \qquad .....(d)$$

$$R \geq 0 \qquad .....(e)$$

$$z_i^2 \geq \frac{A_i^2}{a_{jU} A_j} \qquad .....(f)$$

Case 3:

$$z_j^2 \leq \frac{A_j^2}{a_{iL} A_i} \qquad \text{.....(g)}$$

$$R \leq a_{iU} A_i - a_{jL} A_j \qquad \text{.....(h)}$$

$$\frac{A_i^2}{z_i^2} - \frac{A_j^2}{z_j^2} - R \leq 0 \qquad \text{.....(i)}$$

$$R \geq 0 \qquad \text{.....(j)}$$

$$z_j^2 \geq \frac{A_j^2}{a_{iU} A_i} \qquad \text{.....(k)}$$

$$z_i, w_i, v_{i,} R \geq 0$$

$$INT_{ij} = 0 \text{ or } 1$$

4.5 Model Description

The facility layout problem in this research is formulated as a mixed-binary non-linear programming model. The objective function in the above mathematical model focuses on minimizing the total material handling cost and the shape cost. The equation for the objective function consists of two terms. The first term evaluates the total material handling cost, which is evaluated from the traffic flow and distance between any pairs of departments that have been described in the previous chapter. The second term is introduced to account for the fact that the area and shape of each department are different from each other. The motivations that the absolute values of widths (or heights) of each department pair are represented in the objective function have been discussed in Section 4.1. In other words, a penalty for the different area and shape of departments is evaluated to be the second term of objective function. Thus, when all department areas are equal, this penalty or second term will be equal to zero.

The distance and geometric (or shape) measurement has the important role for locating the departments, so both terms in the objective function have to be weighed. In this research, the normalized weight parameter (α) is assumed to be 0.8, which means 80% of distance measure and 20% of shape measure are added up to be the total cost.

For the convenience of this research, the cost per unit distance per unit load (C_{ij}) and the cost per unit of inequality of the shape (D_{ij}) are assumed to be 1. That means the traveling distance from one department to the other will cost 1 unit (or dollar) per unit length, and the inequality of these department shapes (x-direction and y-direction) will cost 1 unit (or dollar) per unit length as well.

Before proceeding with the description of each constraint, an important insight into the structure of the model is stated. Typically, the model for the facility layout problem would include two categories of constraints as follows:

(1) Non-overlapping constraint

Example: All departments must not overlap each other.

(2) Locating departments in the floor constraint

Example: All departments must be located inside the shop floor.

However, during the course of the model development, the impact of department shapes has to be considered along with the two former constraints. Thus, the shape constraints are added in the model. Those constraints are indicated with the letters instead of number behind the constraint. The reason why shape constraints are indicated with letters is explained in the following paragraphs.

The constraints of the model can be described as follows:

(i) Constraint equations (1)-(4) ensure that the distance between any pair of departments is equal to the lower-left corner point and either half of width or height between department *i* and *j*. Equations 1, 2 and 3, 4 are the transformations of the absolute value of the distance along the X-axis and Y-axis, respectively.

(ii) Constraint equations (5) and (6) are non-overlapping constraints. These constraints state that the distance between the centers of two (*i* and *j*) departments (i.e., x_{ij} along the x-axis, and y_{ij} along the y-axis) is always greater or equal to the sum of their width (or height) divided by two. The binary variables (INT_{ij}) are incorporated in these constraints to ensure that one of the two constraints is always active.

(iii) Constraint equations (7)-(10) determine the width and height of each department to obtain its shape. They are the dimensions of each department, which must be within acceptable limits as defined in the aspect ratio. The area and the shape of the departments are different from each other. Relocating the machines in the department might change the shape of the department. Therefore, a feasible range of aspect ratio should be considered. For instance, the width of the department i (z_i) should be greater or equal to the square root of the area i (A_i) divided by the upper limit of the aspect ratio (constraint equation 8). Two of these constraints ((7)-(10)) will be active when the binary variable P is 0 and the other two will be active when P is equal to1.

(iv) Constraint equations (11) and (12) ensure that all departments are placed inside the floor plan. The maximum value in the parenthesis is the predetermination of the limitation of width (or height) of department i. Because the free orientation department or the flexibility of the department shapes is applied in this research, the values in the parenthesis of the constraint equations (11) and (12) are identical. The free orientation department has been previously explained in Chapter 3. The lower-left corners of department i must not be greater than the width (or height) of floor minus the maximum value of width (or height) of department i, and must be greater or equal to 0.

(v) Constraint equations (a)-(k) perform the function for common height characteristics, which could be divided into 3 cases. The common height (h_i^2) is the height which any two of departments can share their one side of rectangular to each other. The product of area and aspect ratio is equal to the common height (h_i^2). The common height of each department can be expressed in two values because of the boundary of the aspect ratio. The relationship of a pair of departments could be divided in 3 cases, depending upon the common height ranges (see Figure 4.1). All 3 cases of common height relationships can be represented by the following equations:

$$a_{iL} * A_i \leq h_i^2 \leq a_{iU} * A_i$$

Case 1: $a_{iU} * A_i \leq a_{jL} * A_j$

Case 2: $a_{jL} * A_j \leq a_{iU} * A_i \leq a_{jU} * A_j$

Case 3: $a_{jL} * A_j \leq a_{iU} * A_i$ and

$a_{jU} * A_j \leq a_{iU} * A_i$

Here, it is assumed that $a_{iL} * A_i \leq a_{jL} * A_j$ and $h_i = \text{Height} = \frac{A_i}{z_i}$

The assumption above ensures that the common height of department *i* is always to the left of department *j*.

In Figure 4.1, let

$a_{iL} * A_i = \text{Q}$, $a_{iU} * A_i = \text{R}$,

$a_{jL} * A_j = \text{S}$, and $a_{jU} * A_j = \text{T}$

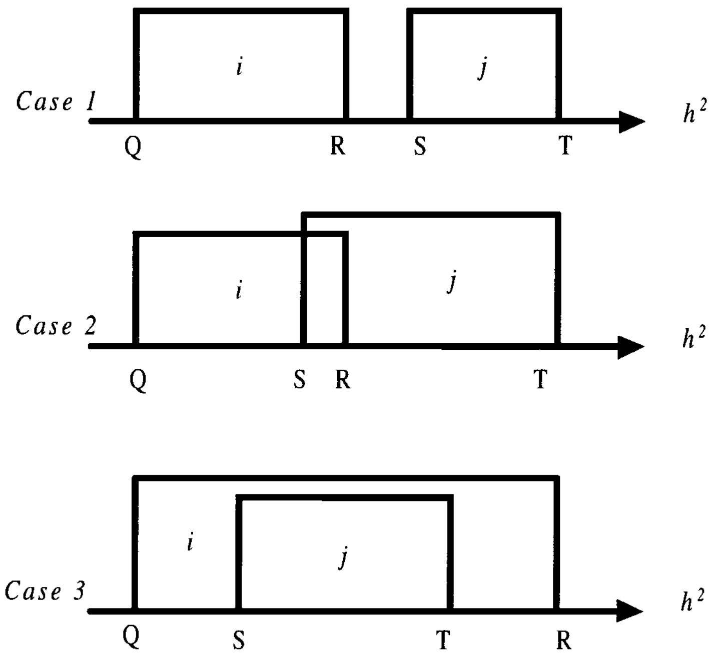


Figure 4.1 Range of h_i^2 and h_j^2 for Three Cases

Thus, three cases above can be evaluated as:

Case 1: $R \leq S$

Case 2: $Q \leq S \leq R \leq T$

Case 3: $Q \leq S \leq T \leq R$

All of the equations above can now be developed as the equivalent constraint equations (a)-(k).

Case 1: There is no common height range between department *i* and *j* (no intersection of the common ranges) In other words, no specific value of height that departments *i* and *j* can use as the same height in order to place departments *i* and *j* next to each other.

Case 2 and 3: There are the common ranges of the height, but the dissimilarity of case 2 and 3 is the intersection between each other. Case 2 is partial intersection and case 3 is full-intersection (see Figure 4.1).

For example, let the total number of departments in a floor plan be equal to 5. Then, the total number of combinations (or pairs) of each department pairs will be equal to 10, which is given by $N!/(2!*(N-2)\ !)$. The given data will also provide the aspect ratios and areas of each department, and the range of common heights can be calculated easily. So, each combination must belong to only one case as presented above. Thus, it can be concluded that one of three cases is always active for each of department pair.

The common height range between departments *i* and *j* represents many possible values of the height. A value of height in the common range will be decided in order to locate these departments next to each other with that height. In addition, all pairs of the equal area departments have only one value of the common height. The common height is considered because from the second term in the objective function the minimization problem can take an advantage when any of two departments have the same height or same width. The $|\frac{A_i}{z_i} - \frac{A_j}{z_j}|$ or $|z_i\text{-}z_j|$ term in the objective function will be equal to 0. In this research the application of the bay orientation, which stacks the departments in the bays and each bay has the same width, is used so the $|z_i\text{-}z_j|$ is always equal to 0 when *i* and *j* are located in the same bay. In other words, if the equal area problem is considered in this research, the second term of objective function will be equal to zero.

4.6 Computational Complexity of the Research Problem

The mathematical model developed above is a binary non-linear programming model. In general, a non-linear programming problem belongs to a class of NP-complete problems (Gary and Johnson 1979). This claim alone is not sufficient to conclude that the research problem is an NP-complete problem. Van Camp *et al* (1991). have investigated a special case of this problem, and proven NP-hard in the strong sense. Their investigation focused on an unequal area layout problem that did not include the impact of shape on the layout. As the special case of this research problem was proven NP-hard in the strong sense, this research problem must be strongly NP-hard as well.

5. HEURISTIC ALGORITHM

5.1 Tabu Search Introduction

Tabu search (TS) is a metaheuristic approach for solving combinatorial optimization problems. It is an adaptive procedure that can be superimposed on many other methods to prevent them from being trapped at locally optimal solutions. The method was pioneered by Glover (1986) and presented in detail in Glover (1990a), (1990b), (1991), and Glover and Laguna (1992). The applications of tabu search have included scheduling, transportation network design, layout and circuit design problems, telecommunications, probabilistic logic and expert systems, neural network pattern recognition, and others.

The Tabu search method starts with an initial solution. Using some local exchange heuristics the method generates from the current solution a list of candidate solutions. If the exchange results in a large number of candidate solutions, the user might decide to restrict the search only to a subset of them. Then, the solutions in the candidate list have to be evaluated. This research deals with a minimization problem and "Cost" is our objective function. The method selects the best solution (configuration of the layout) from the candidate list of solutions as the one having minimum cost. If this selection is forbidden, the method proceeds to select the next best solution in the candidate list. The forbidden status is specified using a set of rules explained in the next section. The selected solution from the candidate list becomes the new current solution.

This process will continue until it reaches the stopping criteria explained below.

The motivation for developing a tabu-search based heuristic algorithm for solving the problem addressed in this research is its computational complexity, which is shown to be NP-hard in the strong sense. Tabu search has been proven to find the optimal or near optimal solution within a reasonable computation time.

5.2 Tabu Search Mechanism

The Tabu search method could be called the hill-climbing heuristic, which progresses unidirectionally from an initial feasible solution to a local optimum. The limitation of a hill-climbing procedure in a combinatorial problem setting is that the local optimum obtained at its stopping point, when no improving moves are possible, may not be a global optimum. In contrast to the hill climbing, TS provides a guide to continue the exploration without becoming confounded by an absence of improving moves and without falling back into a local optimum from which it previously emerges. Tabu search is built based on three primary features (Glover, 1990b).

(1) The use of flexible memory structures to collect information during the search process. It allows the evaluation criteria and historical search information to be exploited more thoroughly than by rigid memory structures (as in branch-and-bound) or by memoryless systems (as in simulated annealing and other randomized approaches).

(2) An associated mechanism that is based on the interaction between imposing and freeing the constraints on the search process (embodied in the tabu restrictions and the aspiration criteria).

(3) The combination of memory functions of different time spans, from short term to long term, to implement strategies for intensifying and diversifying the search.

Tabu search starts with an initial solution, which can be a feasible or an infeasible solution. This initial solution can be randomly or systematically generated. Nevertheless, starting the search with a "good" feasible solution may speed up the process to get to an optimal/near-optimal solution. This is because the solution space is wider if the search process starts from an inferior initial solution. In fact the wider the solution space is, the longer it takes to get to an optimal/near-optimal solution. Since a good initial solution is significant for tabu search, the method for generating initial solutions is developed. This method is explained in detail in the next section. In contrast to hill climbing, where the immediate improved solution is used as the next move, tabu search generates a list of candidates moves from an initial solution by applying simple perturbation methods to the initial solution. This step is usually called a neighborhood search. After that, each candidate move in the list is evaluated and the best one (minimum or maximum solution) is selected as the next move, subject to certain constraints. These constraints, built in the form of tabu restrictions, are designed to prevent the reversal and the repetition of certain moves by rendering selected attributes of these moves forbidden or tabu. The primary goal of the tabu restrictions is to permit the method to go beyond points of local optimality while still making high quality moves at each step. It also records recent moves in the order in which they are made. The length of time a tabu move is enforced depends on the size of the tabu list. The rule FIFO is applied in the tenurity of an attribute. Research in the past has reported that tabu-list size depends on the size of the problems being investigated. Thus, prior experimentation is required to determine a good size for the tabu list.

Tabu search allows the forbidden or tabu moves to be performed in the search process when the aspiration criterion is satisfied. A simple, but widely used, type of aspiration criterion is the removal of tabu status of a move if a candidate move yields the best solution found so far. This means that the tabu restriction can be overridden if an aspiration criterion is satisfied. After all neighborhood solutions are tested against tabu status and aspiration criteria, the move that yields the best solution is selected for future perturbation. Once the best move is selected, it will be admitted into a list called the candidate list (CL). Every chosen best solution has to be checked against the CL. The check is necessary to assure that a solution is not considered more than once for perturbation.

There are different methods to terminate the search process. It could be the maximum number of moves that has been admitted into the index list (IL) or the imposed number of moves without improving the best solution has been performed. In the latter case, if there is no improvement in the objective function value after a specific number of iterations has been performed the entire search would be terminated. Yet another method is to impose a limit on the computation time used in the search process.

In many of tabu search applications, two types of memory functions are applied. Up to this point, the short-term memory is completed. The effect of short-term memory can be amplified by applying the long-term memory function. The long-term memory can be applied to direct the search to focus in the region that is historically found good (intensification process) or in the region that is barely visited (diversification process).

The long-term memory is embodied in a frequency matrix that keeps track of the essential information of all previous moves. After that, a new starting solution can be identified using the information from long-term memory. The search process will use this restarting solution as a new initial solution to do a restart.

5.3 Initial Solution Finding Mechanisms

Before describing the detailed steps of the heuristic algorithm developed in this research, an initial solution finding mechanism is presented. Two major steps for finding a good initial solution are proposed as follows:

5.3.1 Adaptive Slicing Tree Construction

A slicing tree diagram is a binary tree model of slicing structure with *n* leaves and *n-1* nodes, where each node represents a level of the relationship (or a closeness rating of each department pair) between a pair of leaves, and each leaf represents a cell or a department. There are a number of ways to construct a slicing tree diagram. The hierarchical clustering technique is applied to construct the slicing tree. The clustering technique is a group of multivariate techniques used to group objects (subjects, respondents, products, etc.) based on the characteristics they possess. Each object within the cluster will be similar to every other object, and different from objects in other clusters. In other words, homogeneity is maximized within clusters and heterogeneity is minimized between them. In this research, the characteristics that are considered to group the departments together are distance and shape measures. The higher the number of traffic flow, the closer the departments should be placed next to each other. The use of clustering techniques requires a distance measure (v_{ij}) between departments. A dissimilarity coefficient (or closeness rating) denoted as A_{ij} between every pair of departments can be constructed using the traffic information. First, the dissimilarity coefficient between departments is defined by:

$$A_{ij} \quad = \quad 1/(1+v_{ij})$$

where v_{ij} is a distance measure (traffic volume or number of parts moved) between department i and j ($v_{ij} = v_{ji}$). This dissimilarity coefficient based on traffic volume has two properties: (1) it is normalized between 0 and 1; (2) it inverses the order of the traffic volume. Given any two traffic volumes v_{ij} and v_{kl} with $v_{ij} < v_{kl}$, it is clear that $A_{ij} > A_{kl}$. The higher the traffic volume, the lower the level should the slicing tree node be determined.

The dissimilarity coefficients between every pair of departments are calculated to create a symmetrical matrix. This matrix is then input into a numerical clustering procedure in order to construct a slicing tree diagram.

K.Y. Tam (1992) used the number of traffic flow (v_{ij}) for finding the dissimilarity coefficient. For finding the symmetrical matrix in this research, both distance and shape measures are considered simultaneously. The distance measure is the number of traffic flow, while the shape measure is the new coefficient that associated the impact of department shapes, and it can be explained as follows.

The concept of bay configuration used in this research assumes that each bay consists of a limited number of departments. The width of each department in a bay is adjacent to each other. The departments in the same bay are arranged from the bottom to the top of the layout with a specified dimension of width. This arrangement might create an empty space or an unoccupied area at the top of each bay (Figure 5.1 shows the empty spaces of the layout).

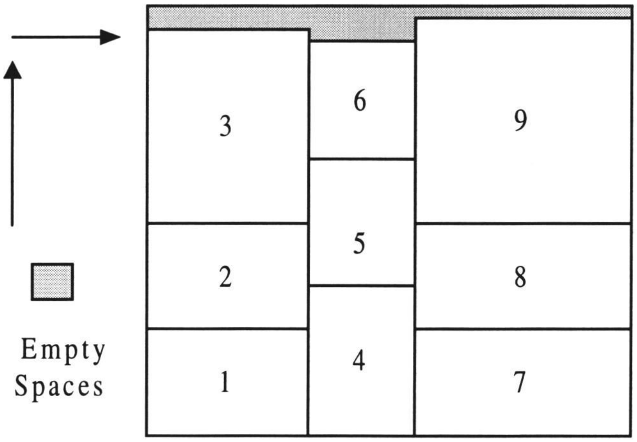


Figure 5.1 Drawing of the Layout

However, the mathematical model presented later will attempt to reduce the amount of empty space. The number of bays must be determined in the early step of bay configuration. In fact, the good looking layout should have the rectangular shape and tends to have the number of bays and number of departments as close as possible. The other reasons are that the shape or geometric constraints will be violated if the number of bays is too small or too large. Thus, the number of bays is equal to the square of the number of departments represented as follows:

$$\textit{Number of bays } (C) = INT\,(N)^{1/2}$$

$$N = \textit{Number of departments}$$

In addition, the number of departments assigned in a bay should be approximated the same as number of bays. For example, given the number of departments is 10, so the number of bays is equal to 3 and number of departments in each bay should be 3. However, there is one more department that remains in a 10-department problem. The department formation is created by assigning the remaining department to the first bay, and to the second bay in the case of one more extra department if it remains (11-department problem). Given the number of departments is 22, the number of bays is equal 4 and number of departments in each bay should be 6, 6, 5 and 5 respectively. However, this assignment is a rough evaluation of the number of departments in each bay (temporary formation) of the layout. The result from a mathematic model, explained in the next section, determines the valid formation of the layout. The formation for the number of departments in each bay can be presented as a "number" and is separated by "- ". For example, the formation 5-3-2 means the first, second and third bay has 5, 3 and 2 departments, respectively.

The total area of the departments in a bay is the sum of the areas of the departments that are assigned to that bay. Thus, each bay has limited area for assigning a few departments. In other words, the total area of the departments in a bay must be less than or equal to that bay area. Once the number of bays is determined, each bay area can be approximately evaluated by dividing the sum of the areas of the departments by the number of bays.

As mentioned before, the distance measure represents the relationship between the traffic flow and the distance between a pair of departments, and the shape measure must represent the relationship between the shapes of a pair of departments. The department area (A_i) is evaluated in order to create the relationship of shape (or shape matrix). The mathematic model, minimizing the difference between the sum of the department areas that are assigned in each bay and each bay area, should be created. Not only can the model create the shape matrix, but it can also reduce the amount of empty space or unoccupied area in each bay. The binary integer programming problem can be presented as:

Minimize

$$Z = \sum_{c=1}^{C} \left| B_c - \sum_{i=1}^{N} A_i / C \right|$$

subject to

$$B_c = \sum_{i=1}^{N} A_i * e(c, i) \qquad ; c = 1, \ldots, C \qquad \ldots\ldots(1)$$

$$\sum_{c=1}^{C} e(c, i) = 1 \qquad ; i = 1, \ldots, N \qquad \ldots\ldots(2)$$

$$e(c, i) = \begin{cases} 1 & \textit{if department i is assigned in bay c} \\ 0 & \textit{otherwise} \end{cases}$$

Notations

Indices

N = Number of cells to be located on the floor

i = Department index (i = 1,..., N)

c = Bay index (c = 1,..., C)

Parameters and Coefficients

C = Maximum number of bays in the layout

A_i = Area of cell i

Variables

B_c = Sum of the department areas that are assigned in bay *c* *e (c, i)* = Binary variables

Constraint equation (1) ensures that sum of the department areas assigned to bay *c* is equal to the variable (B_c) which is used in the formulation of the model.

Constraint equation (2) ensures that each department is assigned to a bay.

In case the number of departments in each bay is not equal, the average bay area is introduced. The average bay area is calculated by the sum of the total area divided by the total number of departments and multiplied by the number of departments that are assigned in each bay. In other words, each bay area is proportional to the number of departments in that bay. The greater the number of departments, the larger the bay area assigned. For example, given there are 10 departments in the layout and the number of bays is equal to 3 (INT $(10)^{1/2}$). One of 3 bays has 4 departments, which is not equal to the number of the departments in the other 2 bays (3 departments). Thus, the average bay area is applied.

As the above model takes on the form of a binary integer linear programming problem, it can be easily solved using the commercial integer LP solver (LINDO, 1998). A feasible solution always exists. It shows the assignment of each department to a bay and also the formation of the departments in each bay. But the placement sequence of departments in a particular bay is still unknown. The formation from the result of the mathematical model is used in the next steps in this research, and the temporary formation is ignored. In addition, the feasible solution also shows the limited number of departments in each bay. At this point, the shape matrix is evaluated in order to create the new symmetrical matrix, which includes both distance and shape measures. The shape matrix is evaluated by rating (0 or 1; 1 for "yes", and 0 for "no") the relationship of the assigned departments within a particular bay. The rating 1 means there is a relationship between the pair of departments due to the shape measure, otherwise the rating 0 is applied.

For example, given the LP solution decides to group departments 1,3 and 7 in the first bay that means departments 1, 3 and 7 have a relationship between each other. The combinations of the department pairs will be 1 vs 3, 1 vs 7 and 3 vs 7. Thus, three department pairs (1 vs 3, 1 vs 7 and 3 vs 7) are put in the shape matrix with rating 1 each, and the other departments that department 1, 3 and 7 do not have a relationship with (Ex. 2, 4, 5, 6, 8, 9 and 10) will rate with 0. After the relationships of department 1, 3 and 7 in the first bay are inputted into the shape matrix, the remaining departments in other bays will repeat this procedure until the shape matrix is completely created. The new dissimilarity coefficients (A_{ij} *) between every pair of departments is represented as follows:

$$A_{ij} * \quad = \quad 1/(1+ v_{ij})$$

The parameter v_{ij} above is a sum of the distance measure and the shape measure. As the data from the traffic matrix and the shape matrix use different units (traffic matrix = number of unit travels, and shape matrix = the rating of the relationships), the normalization is introduced in order to combine two matrixes. Both distance (flow matrix) and shape measures (shape matrix) have to be normalized using a number between 0 and 1. The normalized symmetrical matrix is then inputted to a numerical clustering procedure to construct a slicing tree diagram.

A number of clustering procedures, such as single linkage, complete linkage (or furthest neighbor linkage), and density linkage, have been developed (Anderberg, 1973) and are available in a number of statistical packages (e.g. SAS, SPSS). In his paper, Tam (1992), reported the single linkage method is less attractive than the average linkage method in constructing a slicing tree diagram. This research uses this insight advantageously and applies the average linkage method to create the slicing tree diagram. The average linkage method is included in the Matlab (2000) software.

5.3.2 The Strategy for Interpreting the Slicing Tree to the Initial Solution

After the slicing tree diagram is created, the next step is to interpret this diagram to obtain an initial solution. The smallest value of average linkage (result from the clustering technique) represents the most interaction between two departments. It means that the pair of departments should be located as close as possible. The steps associated with choosing and assigning the departments are presented later. In this research, the configuration of the departments can be read from bottom to top and from left to right and each bay is divided by "/". For example, the configuration given by 1,2,3/4,5,6/7,8,9 is a representation of the layout previously shown in Figure 5.1.

A reference starting point is proposed in order to convert the configuration of the layout to the drawing layout. The drawing layout is assumed to be a grid layout, and the reference starting point is the origin (coordinate (0,0)). The first department in the configuration is always located at (0,0) and the rest of the departments are placed next to it in the order of bottom to top (same bay) and left to right (next bay). For example, the first department of the configuration [1,2,3/4,5,6/7,8,9] is department 1, so it has to be located at (0,0) (see Figure 5.1).

The single linkage clustering (minimum or nearest-neighbor method) uses the nearest distance between two departments to evaluate the dissimilarity coefficient. The single linkage always creates the unbalanced or thin tree structure (Figure 5.2). While the average linkage clustering considers the average distance between two departments, and generates a more balanced tree structure (Figure 5.3). The unbalanced tree structure has an opportunity in violating the geometric or shape constraints. Thus, this research ignores the application of single linkage clustering. More information about the slicing tree is available in Tam's paper (1992).

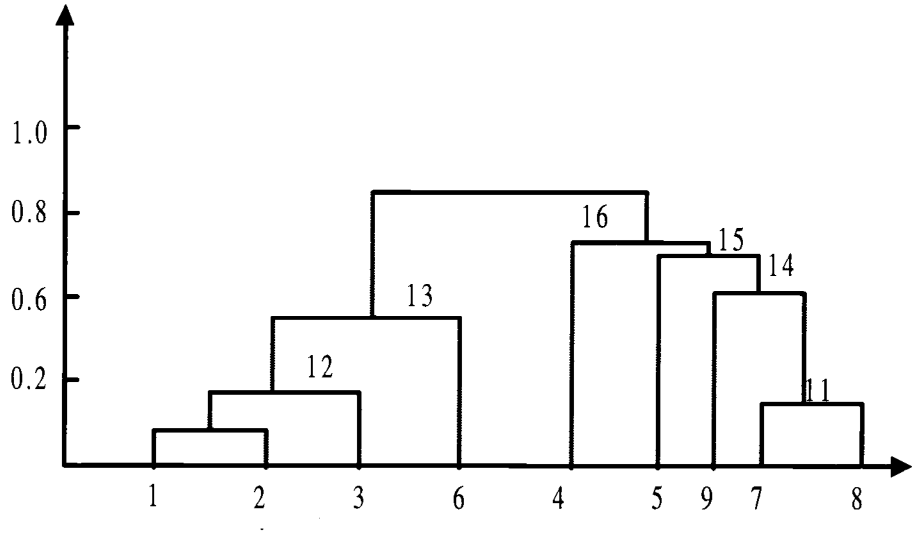


Figure 5.2 The Slicing Tree Diagram for the 9-department Problem (Single Linkage)

Steps in interpreting the slicing tree diagram to obtained an initial configuration are:

(i) Choose the department pair that has the smallest value of average linkage and assign that pair of departments in the lower-left of the floor plan, which is the first bay. In the event of a tie the department that has the smallest department number is selected (for example, pair 1 vs 3 is selected because it has a smaller department number (1) than pair 2 vs 6 (2 and 6)).

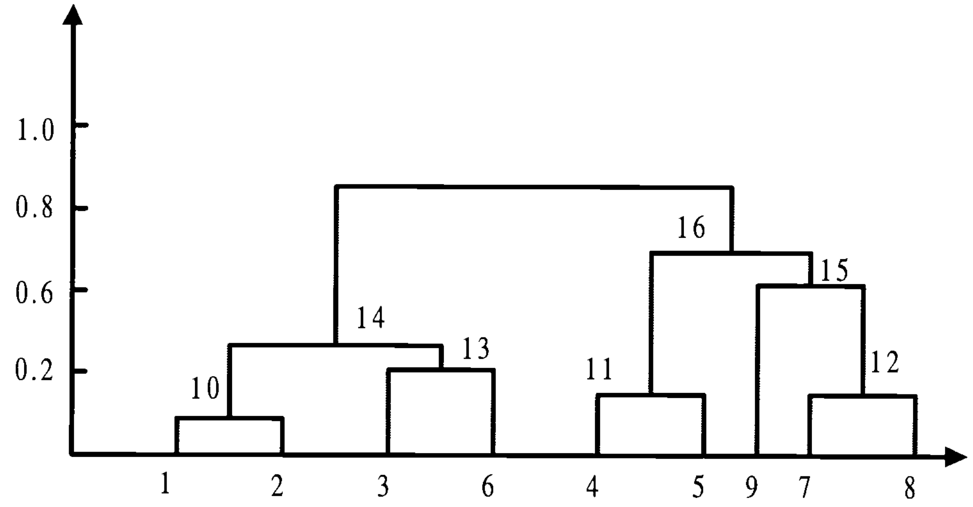


Figure 5.3 The Slicing Tree Diagram for the 9-department Problem (Average Linkage)

(ii) After the first pair is assigned to the first bay, the second best pair (the second smallest value of average linkage) is assigned to the next bay (2^{nd}). The assignment of the department pairs will proceed until each bay has a pair of departments.

(iii) From the slicing tree, a node merges two branches (links or groups) together. The value of each node is calculated from the clustering procedure, which is the average linkage. The node represents the average linkage value between two departments or two groups of departments. The next remaining pair uses the node to merge with the pair that has been assigned (Step (i) and Step (ii)).

(iv) A decision should be made about the single department, which does not pair with any other department. The branch or link of the single department will be the decision criterion in order to assign the location of that single department. (for example, if department 4 has a link with pair 1 vs.3, department 4 will be assigned adjacent to pair 1 vs.3).

Finally, the initial layout configuration is evaluated.

For example, for a 9-department problem the slicing tree diagram (Average linkage) is shown in Figure 5.3 and the average linkage result is shown in Table 5.1. (Note: As this is a 9-department problem, the indices that are greater than 9 in Table 5.1 (10-16) indicate the rank of the department pairs.)

Table 5.1 The Average Linkage Result for the 9-department Problem

Node	Dept.	Dept.	Avg. Linkage
1	1	2	0.15
2	4	5	0.18
3	7	8	0.19
4	3	6	0.31
5	10	13	0.42
6	9	12	0.62
7	11	15	0.73
8	14	16	0.91

From the steps of interpreting the slicing tree diagram to initial configuration, the first three smallest values of average linkage are department pairs, 1 vs.2, 4 vs.5, and 7 vs.8 (Table 5.1). So, they are assigned to the first, second, and third bay, respectively. The next pair, which is 3 vs.6, will be located on the top of the pair that merges with the same node in the tree diagram. In this example, 1 vs.2 has a branch with 3 vs.6, but the first bay can have only 3 departments. So pair 3 vs.6 has to be split to the next closest bay (2nd). The remaining departments will be assigned to the bays in a similar fashion until all of them are located. From this example problem, the initial configuration is [1,2,3/4,5,6/7,8,9], and the layout of this initial solution is previously shown in Figure 5.1.

At this point, the heuristic algorithm operates the tabu search and perturbation methods at two levels (inside and outside tabu searches). The first level deals with the department location identification and the second level deals with the bay assignment. The number of permutations for identifying the locations of departments is many more in comparison to the number permutations for assigning the location of bays. Therefore, the department-location identification has a significantly higher impact on the design of facility layout than the bay assignment. Consequently, in the development of the heuristic algorithm, the inside tabu search will serve as the major search while the outside search will serve as the minor search. The final solution for the problem is composed of the solution corresponding to optimal/near-optimal department-location identification together with the solution corresponding to optimal/near-optimal bay assignment. The flow chart shown in Figure 5.4 illustrates the heuristic mechanism incorporated in the tabu search-based procedure. The pseudo code for the heuristic is also provided in Appendix F.

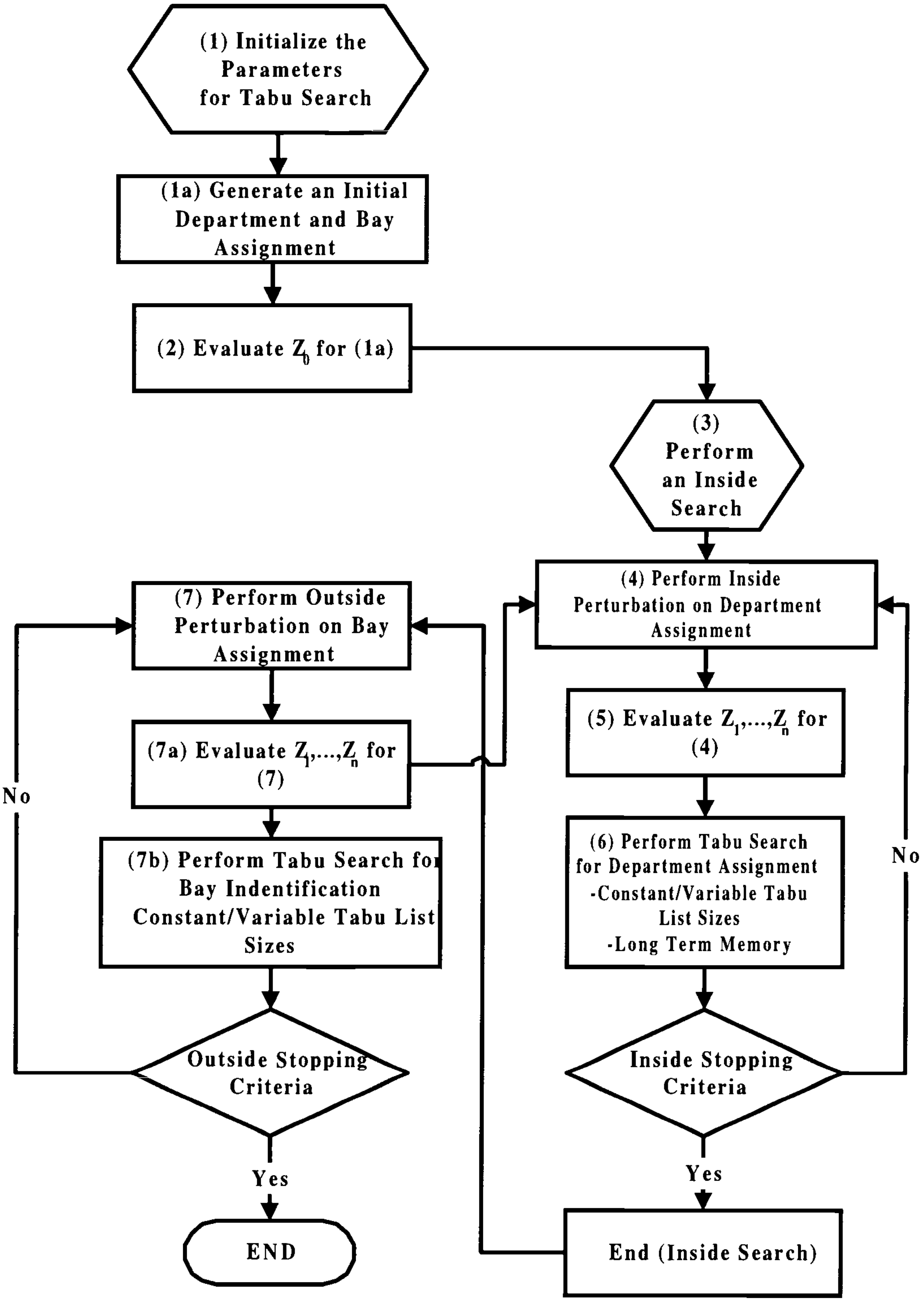

(1) Initialize the Parameters for Tabu Search
(1a) Generate an Initial Department and Bay Assignment
(2) Evaluate Z_0 for (1a)
(3) Perform an Inside Search
(4) Perform Inside Perturbation on Department Assignment
(5) Evaluate $Z_1,...,Z_n$ for (4)
(6) Perform Tabu Search for Department Assignment -Constant/Variable Tabu List Sizes -Long Term Memory
Inside Stopping Criteria
No
Yes
End (Inside Search)
(7) Perform Outside Perturbation on Bay Assignment
(7a) Evaluate $Z_1,...,Z_n$ for (7)
(7b) Perform Tabu Search fo Bay Indentification Constant/Variable Tabu List Sizes
Outside Stopping Criteria
No
Yes
END

Figure 5.4 Flowchart of Tabu Search-Based Heuristic Algorithm.

5.4 Step Associated with Heuristic Algorithm

Notation:

A feasible solution (FS) for the problem considered here consists of a sequence of department-location identifications called FSd and a sequence of bay assignments called FSb. For each sequence, applying a specific neighborhood function to its current sequence could generate a set of seeds.

The application of tabu search begins with the initial solution as the seed. There are two methods developed to generate a set of neighborhood solutions from a seed. The total material handling cost is evaluated for each of the solutions generated by applying these methods. The best solution is then selected as the new seed to generate a new set of neighborhood solutions. This process is repeated in every iteration of tabu search until the search is terminated. The performance criteria and the steps related to tabu search application are explained in the next section.

In order to generate a set of neighborhood solutions from a chosen seed, two methods of moves are applied to the seed: (1) Swap move and (2) Insert move. A swap move is a move that interchanges the position of two departments that are assigned to the same bay or different bay. An insert move is a move that inserts a department to any bay except the one that it currently occupies. The reason for that is if the insert move considered the insertion in the same bay it will repeat the configuration as does the swap move. A swap move allows two departments from the same or different bays to exchange positions. An insert move allows a department move from one bay to another. The structure of solutions produced by swap moves is always the same as the structure of its parent solution (seed). In other words, swap moves do not change the total number of departments that are assigned to each bay. On the contrary, insert moves always produce solutions that change the total number of departments assigned in a bay. However, after the experiment is performed there is good evidence that those insert moves create many infeasible solutions in the facility layout problem. This will be explained later. The swap move and insert move are described separately in the following two subsections.

To illustrate the details of a swap move and an insert move the data used by van Camp *et al.* (1991) is used. The initial solution that needs to start the move is evaluated by constructing the slicing tree diagram. The clustering technique built in the Mathematical software (MATLAB, 2000) is used to create the slicing tree diagram. The slicing tree diagram (or dendogram) for this problem is shown below:

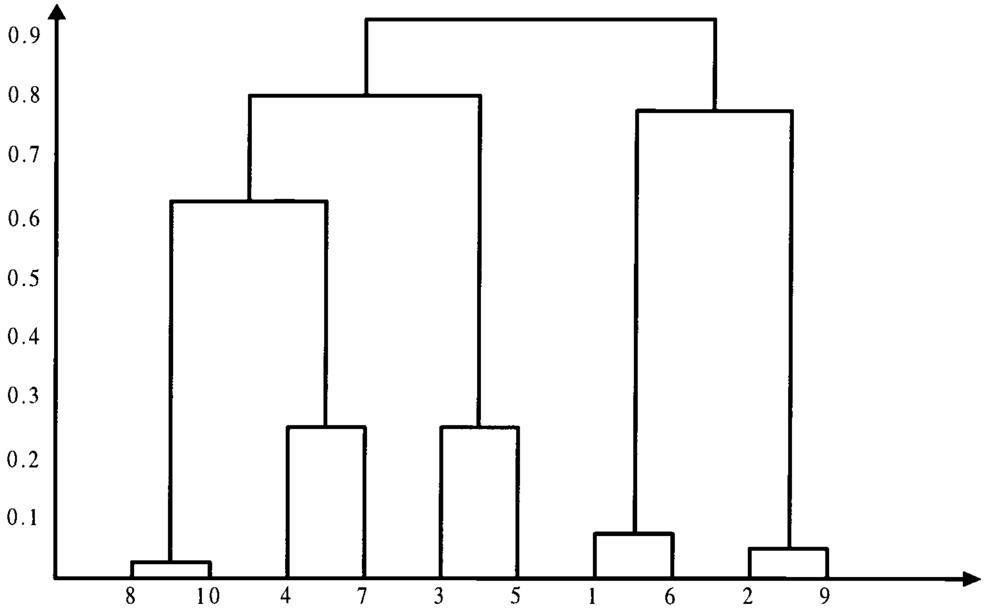


Figure 5.5 The Dendogram for the Ten-Department Problem

From the dendogram, the department configuration is evaluated by following the steps mentioned in Section 5.3.1 and 5.3.2. Thus, the initial solution would be [8,10,4,7/2,9,3/1,6,5]. Up to now, this initial solution configuration is assumed to be valid for explaining the swap moves and insert moves, and the moves are applied in the inside search, which is department-location identification. The details for evaluating this initial solution are explained in Step 1 of the heuristic algorithm presented later.

The sequence of department-location identification can be identified as:

$$[B_1: \{d_1, d_2, d_3, \ldots, d_x\} / B_2: \{d_1, d_2, d_3, \ldots, d_x\} / \ldots. / B_C: \{d_1, d_2, d_3, \ldots, d_x\}],$$

where B_i denotes bay i, d_j denotes department j at location i, x is the maximum number of departments that can be assigned to a bay, and C is the total number of bays in the floor plan.

5.4.1 Swap Move

From the initial configuration, let department 8 be the first department considered for swap. Then, department 8 swaps with department 10, 4, 7, 2, 9,3, 1, 6 and 5, respectively. The configuration after the first swap process will be [10,8,4,7/2,9,3/1,6,5]. The next swap move will consider the next department, which is department 4. If the process is continued, the swap move will move to swap with departments in the second and third bays (2, 9, 3, 1, 6 and 5). Finally, the last department (5) of the starting initial solution will swap with department 8. The swap move is repeated by starting with department 10, and swapping it with department 4. All combinations of any two departments are swapped and the move must not repeat the move that has been considered. In other words, the swap move is a combination of any 2 departments chosen from all departments in the floor plan. Thus, the total number of swap moves will be equal to:

$$\textit{Number of swap moves } (m) = N! / (N-2)! * 2$$

In this example the total number of swap moves is 45.

Table 5.2 The Swap Moves

Swap Moves					
Swap departments	**Cost**	**Swap departments**	**Cost**	**Swap departments**	**Cost**
8 and 10	23570	4 and 7	25590	2 and 5	22830
8 and 4	25430	4 and 2	27670	9 and 3	27180
8 and 7	28290	4 and 9	Infeasible	9 and 1	35000
8 and 2	33410	4 and 3	Infeasible	9 and 6	28280
8 and 9	Infeasible	4 and 1	Infeasible	9 and 5	26990
8 and 3	26340	4 and 6	34490	3 and 1	26500
8 and 1	Infeasible	4 and 5	25000	3 and 6	26560
8 and 6	45290	7 and 2	Infeasible	3 and 5	22350
8 and 5	47660	7 and 9	25260	1 and 6	24620
10 and 4	29310	7 and 3	24020	1 and 5	24100
10 and 7	39090	7 and 1	29610	6 and 5	26660
10 and 2	36090	7 and 6	30670		
10 and 9	Infeasible	7 and 5	22820		
10 and 3	35770	2 and 9	24670		
10 and 1	Infeasible	2 and 3	25130		
10 and 6	53370	2 and 1	31570		
10 and 5	43790	2 and 6	27890		

From Table 5.2, 8 out of the 45 solutions are infeasible solutions. The infeasible solutions show that the swap moves violate the total floor plan restriction (constraint equations (11) and (12) in Chapter 4).

5.4.2 Insert Move

Let department 8 be the first department considered to be inserted into the departments that currently occupy each bay. From the starting initial solution configuration [8,10,4,7/2,9,3/1,6,5], department 8 is inserted in the second bay. It skips its own bay because it will repeat the configuration as that obtained by performing the swap move. There are 3 departments in the second bay, so there will be 4 insert moves in this bay. See configuration items 1 to 4 in Table 5.3.

The insert move will continue until department 8 is inserted next to department 5 in the third bay (item 8). Repeat this move by starting with department 10. The total number of insert moves will be equal to 86 in this example problem. From Table 5.3, 54 out of 86 solutions are infeasible solutions. A comparison shows that 82% of solutions with swap move are feasible solutions but only 37% of solutions are feasible solutions with insert move. Thus, in the unequal area facility layout problem the swap move is more attractive than the insert move. This research applies the swap move and ignores the insert move in the perturbation of the tabu steps.

Table 5.3 The Insert Moves

Insert move								
	Insert departments	Cost		Insert departments	Cost		Insert departments	Cost
1	10,4,7/8,2,9,3/1,6,5	27044	33	2,8,10,4,7/9,3/1,6,5	Infeasible	60	1,8,10,4,7/2,9,3/6,5	Infeasible
2	10,4,7/2,8,9,3/1,6,5	27688	34	8,2,10,4,7/9,3/1,6,5	Infeasible	61	8,1,10,4,7/2,9,3/6,5	Infeasible
3	10,4,7/2,9,8,3/1,6,5	30900	35	8,10,2,4,7/9,3/1,6,5	Infeasible	62	8,10,1,4,7/2,9,3/6,5	Infeasible
4	10,4,7/2,9,3,8/1,6,5	37310	36	8,10,4,2,7/9,3/1,6,5	Infeasible	63	8,10,4,1,7/2,9,3/6,5	Infeasible
5	10,4,7/2,9,3/8,1,6,5	45399	37	8,10,4,7,2/9,3/1,6,5	Infeasible	64	8,10,4,7,1/2,9,3/6,5	Infeasible
6	10,4,7/2,9,3/1,8,6,5	45133	38	8,10,4,7/9,3/2,1,6,5	32013	65	8,10,4,7/1,2,9,3/6,5	26369
7	10,4,7/2,9,3/1,6,8,5	44381	39	8,10,4,7/9,3/1,2,6,5	32481	66	8,10,4,7/2,1,9,3/6,5	28791
8	10,4,7/2,9,3/1,6,5,8	47236	40	8,10,4,7/9,3/1,6,2,5	31953	67	8,10,4,7/2,9,1,6/6,5	27123
9	8,4,7/10,2,9,3/1,6,5	29905	41	8,10,4,7/9,3/1,6,5,2	33317	68	8,10,4,7/2,9,3,1/6,5	27639
10	8,4,7/2,10,9,3/1,6,5	31530	42	9,8,10,4,7/2,3/1,6,5	Infeasible	69	6,8,10,4,7/2,9,3/6,5	Infeasible
11	8,4,7/2,9,10,3/1,6,5	33410	43	8,9,10,4,7/2,3/1,6,5	Infeasible	70	8,6,10,4,7/2,9,3/1,5	Infeasible
12	8,4,7/2,9,3,10/1,6,5	37373	44	8,10,9,4,7/2,3/1,6,5	Infeasible	71	8,10,6,4,7/2,9,3/1,5	Infeasible
13	8,4,7/2,9,3/10,1,6,5	46078	45	8,10,4,9,7/2,3/1,6,5	Infeasible	72	8,10,4,6,7/2,9,3/1,5	Infeasible
14	8,4,7/2,9,3/1,10,6,5	49495	46	8,10,4,7,9/2,3/1,6,5	Infeasible	73	8,10,4,7,6/2,9,3/1,5	Infeasible
15	8,4,7/2,9,3/1,6,10,5	49216	47	8,10,4,7/2,3/9,1,6,5	Infeasible	74	8,10,4,7/6,2,9,3/1,5	Infeasible
16	8,4,7/2,9,3/1,6,5,10	51555	48	8,10,4,7/2,3/1,9,6,5	Infeasible	75	8,10,4,7/2,6,9,3/1,5	Infeasible
17	8,10,7/4,2,9,3/1,6,5	Infeasible	49	8,10,4,7/2,3/1,6,9,5	Infeasible	76	8,10,4,7/2,9,6,3/1,5	Infeasible
18	8,10,7/2,4,9,3/1,6,5	Infeasible	50	8,10,4,7/2,3/1,6,5,9	Infeasible	77	8,10,4,7/2,9,3,5/1,5	Infeasible
19	8,10,7/2,9,4,3/1,6,5	Infeasible	51	3,8,10,4,7/2,9/1,6,5	Infeasible	78	5,8,10,4,7/2,9,3/1,6	Infeasible
20	8,10,7/2,9,3,4/1,6,5	Infeasible	52	8,3,10,4,7/2,9/1,6,5	Infeasible	79	8,5,10,4,7/2,9,3/1,6	Infeasible
21	8,10,7/2,9,3/4,1,6,5	29842	53	8,10,3,4,7/2,9/1,6,5	Infeasible	80	8,10,5,4,7/2,9,3/1,6	Infeasible
22	8,10,7/2,9,3/1,4,6,5	30328	54	8,10,4,3,7/2,9/1,6,5	Infeasible	81	8,10,4,5,7/2,9,3/1,6	Infeasible
23	8,10,7/2,9,3/1,6,4,5	29299	55	8,10,4,7,3/2,9/1,6,5	Infeasible	82	8,10,4,7,5/2,9,3/1,6	Infeasible
24	8,10,7/2,9,3/1,6,5,4	29500	56	8,10,4,7/2,9/3,1,6,5	Infeasible	83	8,10,4,7/5,2,9,3/1,6	20789
25	8,10,4/7,2,9,3/1,6,5	Infeasible	57	8,10,4,7/2,9/1,3,6,5	Infeasible	84	8,10,4,7/2,5,9,3/1,6	22159
26	8,10,4/2,7,9,3/1,6,5	Infeasible	58	8,10,4,7/2,9/1,6,3,5	Infeasible	85	8,10,4,7/2,9,5,3/1,6	21120
27	8,10,4/2,9,7,3/1,6,5	Infeasible	59	8,10,4,7/2,9/1,6,5,3	Infeasible	86	8,10,4,7/2,9,3,5/1,6	21967
28	8,10,4/2,9,3,7/1,6,5	Infeasible						
29	8,10,4/2,9,3/7,1,6,5	Infeasible						
30	8,10,4/2,9,3/1,7,6,5	Infeasible						
31	8,10,4/2,9,3/1,6,7,5	Infeasible						
32	8,10,4/2,9,3/1,6,5,7	Infeasible						

There are two levels of search: (1) Inside search and (2) Outside search. The departments within the bay (FSd) are denoted as inside search and the bay configurations (FSb) are denoted as outside search. Two different sets of seeds considered for such a feasible solution are defined as follows:

- Sd(FSd) = [FSd': FSd' is a sequence of department locations obtained from FSd by perturbing on each location, but one location at a time].

  Sd, which is also called the inside perturbation, starts by generating a set of seeds from FSd. Then, it evaluates each seed in the set and returns the one with the minimum value FSd'.

The procedure used for the inside perturbation is as follows:

(1) Perturb on each department occupying a location, but one department at a time.

(2) Start the swapping process by exchanging a pair of department locations, while the other departments remain at their original locations. Basically, the two departments are swapped, while all the other departments remain at their original locations.

(3) Perform the inside perturbation on every unique combination of two different departments in the same bay and different bay.

- Sb(FSb) = [FSb': FSb' is a sequence of bay assignment obtained from FSb by excecuting the neighborhood function (Sb)].
- Sb, which is also called the outside perturbation, starts by generating a set of seeds from FSb. Then, it evaluates each seed in the set and returns the one with the minimum value as FSb'.

The procedure used for the outside perturbation is as follows:

(1) Perturb on each bay that has a set of departments in each bay.

(2) The perturbation starts by swapping a bay with the other bay, which is next to it, while the other bays, which are not selected to move, remain at their original assignment.

(3) Perform the outside perturbation on every unique combination of two different bays.

Once, the notations are given, the steps associated with the tabu search-based heuristic algorithm can be summarized as follows:

Step 1: Generate the first initial solution (FS_0). FS_0 consists of the first outside initial solution (FSb_0) and the first inside initial solution (FSd_0).

- FSb_0 is the sequence of bay-location identification set by the outside search. It is described as:

  $[B_1; B_{2;} \ldots B_C]$,

  where C is the maximum number of bays that can be in the floor plan. Normally, this configuration is started with $[B_1; B_2; \ldots B_C]$ or [1; 2; 3...C].

- FSd_0 is the sequence of department-location identification. It is associated with FSb_0. FSd_0 is described as:

  $[B_1\text{: } \{d_1, d_2, d_3, \ldots, d_X\}; B_2\text{: } \{d_1, d_2, d_3, \ldots, d_X\}; \ldots .B_C\text{: } \{d_1, d_2, d_3, \ldots, d_X\}]$,

  where B_i denotes bay i, d_j denotes department j at location i, X is the maximum number of departments that can be assigned to a bay, and C is the total number of bays in the floor plan.

Then, follow the strategies for finding the initial configuration

1. Normalize the traffic flow (material flow). Find the total number of traffic flows in the from-to chart or flow matrix, and find out the normalized traffic flow. It will create a new traffic flow matrix.

2. Evaluate the number of bays in the floor plan by

   $$Number\ of\ bays\ (C) = INT\ (N)^{1/2}$$

   $$N = Number\ of\ departments$$

   After the number of bays is evaluated (refer to Section 5.3), the number of departments in each bay is automatically evaluated, which is approximately equal to *C*. The reason is, no bay would have too many (or too few) departments compared with each other. Otherwise, the overall final layout might have an unbalanced shape. Thus, the number of bays would be close to the maximum number of departments in each bay. The number of departments in each bay is estimated in order to create the temporary formation.

3. Calculate the shape matrix by using the solution from the binary integer programming model. Some departments would be assigned in a specific bay. For example, if departments 1 and 3 are assigned in the same bay, the department 1 to 3 in the shape matrix will be set equal to one. It means that departments 1 and 3 have a relationship that contributes to the shape measure. After the collection of all department pairs, the shape matrix would be created. At this point, the temporary formation changes to the formation from the result of binary integer programming.

4. Normalize the shape matrix.

5. Sum the flow matrix and shape matrix.

6. From the result of previous step, find the dissimilarity coefficient (A_{ij}*) by

$$A_{ij}\,* \quad = \quad 1/(1+ \textit{the matrix from 5.})$$

7. Cluster the departments to construct the slicing tree diagram. The slicing tree will be interpreted to find the initial solution configuration.

8. From the slicing tree diagram created in step 7, assign the pairs that have the lowest similarity coefficient in the left-most bay to right-most bay. In other words, the first *C* pairs of departments that have a small similarity coefficient will be assigned to the first *C* bays. There might be a tie among the similarity coefficients. Ties are broken in favor of the department pair that has the smallest index. For example, if the similarity coefficient of department 1 vs 3 is equal to department 2 vs 7, department 1 vs 3 is chosen first (1 is smaller index than 2).

9. The department pairs that are not assigned will be selected to be placed next to the departments that have been assigned in step 8. If a department or a pair of departments has a branch (link) to the assigned pair, it will be placed next to that assigned pair (see the tree diagram). The number of departments must not exceed the maximum number of departments in a bay (*C*).

10. If there are some departments that are not assigned, the smaller index will be selected and assigned in the available bay.

11. Finally, the initial solution will be identified.

Step 2: Evaluate the cost associated with the distance measure (material handling) and shape measure by using the initial solution (FS_0), which are the FSd_0 and FSb_0. As mentioned in the previous chapter, the unequal area facility layout problem is originally developed as a mixed-binary non-linear programming model and is a NP-hard problem in the strong sense. A transformation technique is proposed in order to transform the mixed-binary non-linear programming into the controllable problem. The transformation technique and how the unequal area problem is controlled are explained next.

5.4.3 Transformation Technique

Refer to the original objective function and constraint equations in Chapter 4; one of the most important variables is width (or height) of each department. From the relationship between the aspect ratio and the area of each department, the range of width (or height) of each department can be evaluated. In other words, the dimension of width or height can be evaluated as the values of upper and lower bounds, which have been established in the constraint equations (7) - (10). The width or height variable becomes a known value, when a value of the department width or height in the range is selected. The selection of a value must also correspond to the shortest distance relationship with other departments.

In fact, two departments have the shortest distance (centroid to centroid) when they are adjacent to each other. When two departments have a long common width (adjacent side of a department pair), they always reduce the distance between them. Figure 5.6 shows the distance comparison between two pairs of departments that have different common width. In this research, between the common width and height, the common width is used in order to correspond to the bay configuration orientation.

From Figure 5.6, Pair B has the longer common width than Pair A, so Pair B has the shorter distance between them than Pair A. This fact corresponds to the concept of bay configuration (which groups the departments that have the high interactions together in the same bay).

The total distance will be substantially reduced when both of bay configuration concept and the maximum common width are simultaneously applied. The common width (or height) ranges of each pair are easily evaluated from the given data (Areas and Aspect ratios).

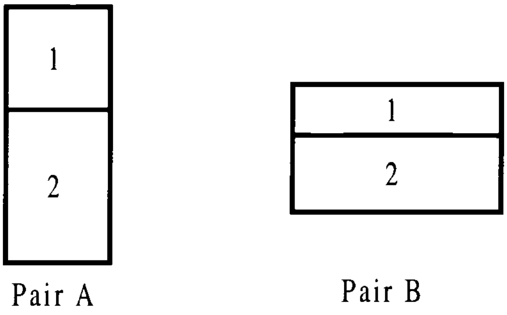


Figure 5.6 The Comparison Between Two Pairs of Departments

From the bay configuration concept, each department in the same bay has the same width but different height. Thus, a value in the intersection of the common width ranges that are grouped in the same bay is selected to identify the width in each bay.

From the previous paragraphs, the widths (variables) in the original model can be evaluated by selecting a minimum of maximum common width in each bay. All the known widths and heights (variables in the original model) in the mathematical model (Section 4.4) are substituted next. As mentioned in Section 5.3.2, the first department in the configuration must be placed in the coordinate (0,0), and the rest of the departments are placed in the order of bottom to top (same bay) and left to right (next bay). Now, the mixed-binary non-linear programming model is transformed into the controllable problem. The objective function value (Z_0) is calculated by using a Matlab 6 (Mathworks, 2000).

As the model is originally in the form of a mixed-binary non-linear program, it can be solved using the commercial LP solver (Lingo, 1998). Notice that this model is only capable of handling some feasible solutions. In many circumstances, infeasible solutions could also exist if the dimensions of the total area of the floor plan are too restricted. In addition, in the case of infeasible solutions, the model would be useless because the solutions obtained from the LP solvers cannot be interpreted. So the application of tabu search is introduced, and the program is written in Matlab 6 (Mathworks, 2000). The restriction of the total area of the floor plan in the original model is observed introducing the penalty procedure strategy. The penalty procedure is proposed in many published researches in order to differentiate the infeasible solutions from the feasible solutions. This research integrates the strategy into the model as follows:

- Constraint equations (11) and (12) ensure that all departments are placed inside the floor plan. For any department that exceeds the boundary of the floor plan, the penalty procedure adds a constant value (M1) in the objective function value that is sufficiently large to differentiate the infeasible solutions from the feasible solutions.

The only constraint equations that restrict the feasibility of the layout are constraint equations (11) and (12). If the solution reports an infeasible solution that means the layout exceeds the boundary of the floor plan.

Step 3: Given the initial objective function value (Z_0) of the initial solution (FS_O), perform an inside search to explore for a new and better solution. The inside search will only focus on the assignment of departments in a bay. It will take a sequence of department assignment associated with the current bay-location identification set by the outside search and attempt to improve it. At this point, the inside search will take the FSd_O and use it as the initial parent node to start the search.

Step 4: Using the inside initial solution (FSd_O), generate a set of seeds by perturbing on each department, but one department at a time. The perturbation procedure is given by the inside perturbation described at the beginning of this section.

Step 5: Evaluate the objective function value (Z) of each seed using the same procedure outlined in Step 2. From the seed evaluation, select the seed that has the minimum value and use it as the parent node for the subsequent move of the inside tabu search. The inside search will move from one configuration of department assignment to another and thus, at each move, the parameters that need to be updated are as follows:

(1) Inside Tabu List (in_TL)

The in_TL is a parameter because it is used as a list to prevent performing the search by perturbing on a department configuration that was most recently perturbed.

Whenever an inside move is performed, the in_TL is updated by admitting the perturbing attributes into the list. The perturbing attributes contain the information on departments and bays that are involved. The perturbing attributes that appear in the in_TL indicate that they have been considered at some previous iterations and thus, they receive tabu status. They would not be considered in the next several iterations, unless their tabu status has expired or an aspiration criterion, which allows the tabu status to be overridden, is satisfied. The perturbing attributes will remain tabu for only a certain number of iterations determined by the inside tabu list size. The in_TL is updated circularly according to its size. It means that if the in_TL was stored up to its size, the oldest entry must be removed before the next entry is stored (First-in-first-out, FIFO). There are two types of tabu list size in this research (1) the fixed tabu list size, (2) the variable tabu list size. In determining the formula used for each parameter of the inside tabu search, it is observed that they are closely related to the number of departments. This relationship of tabu list size and number of departments corresponds to the evaluation of tabu list size in the Chiang and Kouvelis's paper (1996), which they studied in the equal area department problem. Therefore, estimation for the number of perturbations performed during the inside neighborhood search is given as follows:

- For fixed in_TL $= \lfloor (N/\text{factor})^{1/2} \rfloor$, if $(N/\text{factor})^{1/2}$ is a real number with a decimal value < 0.5

  $= \lceil (N/\text{factor})^{1/2} \rceil$, if $(N/\text{factor})^{1/2}$ is a real number with a decimal value ≥ 0.5

- For variable in_TL, there will be three sizes,

- The initial size = $\lfloor (N/factor)^{1/2} \rfloor$, if $(N/factor)^{1/2}$ is a real number with a decimal value < 0.5

  = $\lceil (N/factor)^{1/2} \rceil$, if $(N/factor)^{1/2}$ is a real number with a decimal value ≥ 0.5

- The decreased size = $\lfloor (N/(factor*2))^{1/2} \rfloor$, if $(N/(factor*2))^{1/2}$ is a real number with a decimal value < 0.5

  = $\lceil (N/(factor*2))^{1/2} \rceil$, if $(N/(factor*2))^{1/2}$ is a real number with a decimal value ≥ 0.5

- The increased size = $\lfloor (N/(factor*0.5))^{1/2} \rfloor$, if $(N/(factor*0.5))^{1/2}$ is a real number with a decimal value < 0.5

  = $\lceil (N/(factor*0.5))^{1/2} \rceil$, if $(N/(factor*0.5))^{1/2}$ is a real number with a decimal value ≥ 0.5

where N is the total number of departments in the floor plan, and the factor increases when the problem size increases:

$$\mathrm{INT}(x) = \begin{cases} \lfloor x \rfloor, & \text{if x is a real number with a decimal value} < 0.5 \\ \lceil x \rceil, & \text{if x is a real number with a decimal value} \geq 0.5 \end{cases}$$

(1) Small size problem	5 to 10 departments, factor	=	1.4
(2) Medium size problem	11 to 20 departments, factor	=	2.6
(3) Large size problem	21 to 30 departments, factor	=	2.8

(2) Inside Aspiration Level (in_AL)

The aspiration criterion is the condition a tabu search has to satisfy in order to be released from its tabu restriction. At the beginning of the search process, Aspiration

Level (AL) is set to be equal to the total cost of the initial solution. At every iteration, if the total cost of the selected best solution is less than AL, it is updated to be equal to the total cost of the selected best solution.

(3) Inside Candidate List (ICL) and Inside Index List (IIL)

The ICL collects the best configuration of department assignment selected at each iteration that would be applied for future perturbations while the IIL collects the configurations that are the local optima of the inside search. The functions and operations of the two lists are described below.

At the start of the search, the initial solution (FSd_0) is considered as the first local optimum, therefore it is admitted to the IIL as well as CL. When all seeds of an initial node have been evaluated, the configuration that contributes to the lowest objective (in minimization problem) function value (Z) is selected and admitted into the ICL and used as the new node for the next perturbation. The new configuration in ICL that has its objective function value (Z_1) smaller than the initial objective function value (Z_0) would receive a star. The star indicates that it has the potential for becoming the next local optimum.

Now, the new configuration FSd_1 is then perturbed in a similar fashion. The next configuration, which would be admitted into the ICL, is selected as that having the best objective function value (Z_2) from among the seeds perturbed from FSd_1. Suppose that $Z_2 \geq Z_1$, then the configuration corresponding to Z_1 would receive double stars, and would be admitted into the IIL as the first local optimum obtained for the inside search. Otherwise, Z_2 would receive a star. A configuration receiving a star has the potential for becoming the next local optimum while a configuration with double stars is the next local optimum and, therefore, admitted into the IIL. Before a configuration is admitted to the CL, it has to be checked against all entries in the CL. If the configuration already exists in the CL, another best configuration has to be chosen instead.

(4) Stopping Criteria

There are two stopping criteria considered in this research: The number of iterations without improvement (IWI) and the number of entries into the Inside Index List (IIIL). These two criteria are applied together in monitoring the inside tabu search. The search will be terminated, if one of the criteria is met.

When the solution obtained from the current inside move does not show any improvement over the solution of the previous inside move, the IWI is increased by one. On the other hand, it is reinitialized back to zero whenever an improvement over the previous inside move is found. The IIIL is increased by one every time that an inside move is admitted into the Inside Index List (IIIL). The number of entries into the IIIL represents the number of local optima found so far during the inside search.

Based on the preliminary experimentation, the IWI and IIIL are assumed proportional to the total number of departments in the floor plan. Thus, the stopping criteria are evaluated as follows:

- For the fixed tabu list size, the inside stopping criteria are determined by the formula:

IWI $= \lfloor(N*\text{factor})^{1/2})\rfloor$, if it is a real number with a decimal value < 0.5

$= \lceil(N*\text{factor})^{1/2})\rceil$, if it is a real number with a decimal value ≥ 0.5

IIIL $= \lfloor((N*\text{factor}*1.3)^{1/2}\rfloor$, if it is a real number with a decimal value < 0.5

$= \lceil (N*factor*1.3)^{1/2} \rceil$, if it is a real number with a decimal value ≥ 0.5

- For the variable tabu list size, the inside stopping criteria are determined by the formula:

IWI $= \lfloor 0.56*(N*factor)^{1/2} \rfloor$, if it is a real number with a decimal value < 0.5

$= \lceil 0.56*(N*factor)^{1/2} \rceil$, if it is a real number with a decimal value ≥ 0.5

IIIL $= \lfloor N*factor*1.3)^{1/2} \rfloor$, if it is a real number with a decimal value < 0.5

$= \lceil N*factor*1.3)^{1/2} \rceil$, if it is a real number with a decimal value ≥ 0.5

The guideline for using IWI with variable in_TL is as follows:

- If there is no improvement within the last IWI iteration with the initial in_TL, then decrease the in_TL to the decreased size evaluated in step 5.
- If there is no improvement within the last IWI iteration with the decrease in_TL, then increase the in_TL to the increased size evaluated in step 5.
- If there is no improvement within the last IWI iteration with the increase in_TL, then terminate the inside search.

Step 6: To intensify and diversify the search performed in step 5, the advance mechanism of tabu search called the long-term memory, is also employed. The long-term memory for inside search (in_LTM) is used to direct the search into a new region that has greater potential of getting superior results. The LTM can be directed to explore into the area that has provided good solutions previously, for the intensification process or into the area that has received the least attention from previous searches, for the diversification process. The LTM utilizes a matrix that keeps track the frequency of inside moves attribute. The attribute of interest is the placement of departments at their locations. So, the LTM matrix keeps a record on the number of times that each department has been assigned to a specific location according to the history of moves obtained by the inside search. The matrix is updated regularly as the inside search progresses. Every time an outside move is performed, the entry in the matrix, which corresponds to the department-location identification at that point, is increased by one. By keeping track of the frequency of department-location identification, the LTM matrix provides the information about which locations have been occupied the most or least frequently by specific departments.

From the information obtained from the LTM frequency matrix, a restart configuration is generated. The restarts generate new initial configurations, which are intended to intensify or diversify the search into new regions. The new initial configuration is determined by applying the LTM frequency matrix to the initial department-location configuration that was found in step 1. There are two types of LTM in this research: the LTM based on maximal frequency (in_LTM_MAX) and the LTM based on minimal frequency (in_LTM_MIN). The LTM_MAX is intended to intensify the search by focusing on the area that has been searched frequently in previous searches, while the LTM_MIN is aimed at diversifying the search by directing the search to the area that has received the least attention in previous searches. The LTM_MAX generates a restart configuration by fixing a department to a respective location according to the maximal entry of the LTM frequency matrix. When a department is fixed to its respective location, the inside perturbation of tabu search would not perturb on them. This binding of department to location will remain throughout the duration of the search for that restart until a new restart is generated again and a new binding other than the previous one will become effective. The LTM_MIN is implemented in the same way as the LTM_MAX, except it generates its restart according to the minimal entry of the LTM frequency matrix. The number of departments that would be fixed to their locations and the number of restarts is equal to 1 and 2, respectively, based on preliminary experimentation. At the end of each restart, the LTM frequency matrix has to be reinitialized to zero.

When the required number of restarts for the inside search has been reached, the entire search would be terminated. Then, the final solution will give the lowest total cost for the entire search process (minimization problem).

Step 7: When the inside search is terminated, the optimal/near optimal department assignment would be obtained as the one that contributes to the lowest cost found throughout the inside search. The direction of the search would be returned to the outside search. Perform the outside search, in the same fashion as the inside search, for the bay locations level (outside search). The out_move is identified by the move that transforms a bay location configuration into another bay locations configuration considered among the seeds. By using the minimization of total cost from the inside search for each bay locations configuration in the seeds, the out_move is performed in the same manner as the in_move. Evaluate each bay location configuration ($Z_1, \ldots, Z_n$), for the initial department perturbation (step 4). The value of the move and the aspiration criterion would also be investigated in a similar fashion to those for the inside search. From the preliminary experiment, the parameters of the outside search are corresponding to the number of bays (C) in the problem. The following parameters for the outside tabu search are updated as the search progresses.

(1) Outside-tabu list (out_TL)

Every time an out_move is performed, the bay that moved to the next adjacent location would be admitted into the out-tabu list along with its original location. The out_tabu list is updated circularly as the in_tabu list is updated in the inside search. Two types of out_tabu list are considered.

The fixed tabu-list size for the outside search is determined by the following formula.

For fixed out_TL $= \lfloor (C-1)/2 \rfloor$, if (C-1)/2 is a real number with a decimal value < 0.5

$= \lceil (C-1)/2 \rceil$, if (C-1)/2 is a real number with a decimal value ≥ 0.5

For variable out_TL, there will be three sizes,

- The initial size of out_TL $= \lfloor (C-1)/2*0.95 \rfloor$, if (C-1)/2*0.95 is a real number with a decimal value < 0.5

$= \lceil (C-1)/2*0.95 \rceil$, if (C-1)/2*0.95 is a real number with a decimal value ≥ 0.5

- The decreased size of out_TL $= \lfloor (C-1)/2.1 \rfloor$, if (C-1)/2.1 is a real number with a decimal value < 0.5

$= \lceil (C-1)/2.1 \rceil$, if (C-1)/2.1 is a real number with a decimal value ≥ 0.5

- The increased size of out_TL $= \lfloor (C-1)/1.1 \rfloor$, if (C-1)/1.1 is a real number with a decimal value < 0.5

$= \lceil (C-1)/1.1 \rceil$, if (C-1)/1.1 is a real number with a decimal value ≥ 0.5

where C is the total number of bays.

For the perturbation of bay locations, the maximum number of seeds that can be generated is equal to (C-1) which means the out_move is limited to (C-1) alternatives. Realistically, therefore, the sizes of out_tabu list are proportional to (C-1) which is the number of seeds for each out_move.

(2) Outside Aspiration Level (out_AL)

Similar to the inside search, the aspiration criterion, out_AL, is created and initially set equal to the total cost for the initial bay location configuration. The out_tabu status can be overwritten only when the corresponding bay locations configuration contributes to a total cost less than the aspiration level at the current iteration.

(3) Outside Candidate List (OCL) and Outside Index List (OIL)

In the same fashion as the inside search, OCL and OIL are created for the outside search. OCL contains the potential bay locations configurations selected to perform future perturbation, while OIL consists of the local optima evaluated as the outside search progresses. The approaches used for admitting the bay locations configuration into the OCL and OIL are comparable to those for the ICL and IIL. Thus, the OCL and OIL are analogous to the ICL and IIL, respectively. The final solution, indicating which locations each bay should take, is selected as the entry into the OIL which contributes the lowest total cost.

(4) Stopping Criteria

In order to terminate the outside search, a stopping criterion is considered: the number of iterations without improvement (OWI). This criterion is used in monitoring the outside tabu search. If OWI is satisfied, the search is terminated.

The OWI is increased by one if a non-improvement solution is found after an outside move is performed and on the other hand, it is reinitialized back to zero whenever an improvement over the previous outside move is found. The number of entries into the Outside index list (OIIL) are not used in this research due to the different bay locations configuration are not many as the department location configuration.

Based on the preliminary experimentation, the stopping criteria are evaluated as follows:

- For the fixed tabu list size, the inside stopping criteria are determined by the formula:

$$\begin{aligned} \text{OWI} &= \lfloor (C-1)/2*1.2 \rfloor, \text{ if it is a real number with a decimal value} < 0.5 \\ &= \lceil (C-1)/2*1.2 \rceil, \text{ if it is a real number with a decimal value} \geq 0.5 \end{aligned}$$

- For the variable tabu list size, the inside stopping criteria are determined by the formula:

$$\begin{aligned} \text{OWI} &= \lfloor (C-1)/2*1.2*.95 \rfloor, \text{ if it is a real number with a decimal value} < 0.5 \\ &= \lceil (C-1)/2*1.2*.95 \rceil, \text{ if it is a real number with a decimal value} \geq 0.5 \end{aligned}$$

The guideline for using OWI with variable out_TL is as follows:

- If there is no improvement within the last OWI iteration with the initial out_TL, then decrease the out_TL to the decreased size evaluated in step 7.

- If there is no improvement within the last OWI iteration with the decrease out_TL, then increase the out_TL to the increased size evaluated in step 7.

- If there is no improvement within the last OWI iteration with the increase out_TL, then terminate the outside search.

The entire search would be terminated when the required number of restarts for the inside search and the number of iterations without improvement (OWI) have been reached in the inside and outside search, respectively. The number of restarts for the inside search is assumed equal to 2. Finally, it would return the optimal/near-optimal bay-location configuration together with optimal/near-optimal department-location configuration, which is the configuration that gives the lowest total cost for the entire search process.

5.5 Application of the Heuristic Algorithm to an Example Problem

An example problem is presented to illustrate the application of the heuristic. The example problem involves ten departments or cells on a shop floor. The data and assumptions for this example were carefully chosen from the paper by van Camp *et al.* (1991). He presented a heuristic algorithm to be used in the development of minimal-cost facility layouts, which is the same objective function as that in this research. The heuristic algorithm used was based on nonlinear programming (NLP) techniques. He claimed that the data for his problem came from a real production plant that produced electronic components (Bhatnagar, 1989). The problem consists of ten departments of unequal areas. These areas A_i, in square meters, are given in Table 5.4. The layout in his work was to be developed for an existing facility, and the overall shape of the shop floor was constrained to being rectangular. The dimensions of the shop floor are 25 ×51 m. Thus, the total area is equal to 1275 m^2. It was assumed that for a valid layout, no department could be narrower than $\underline{5}$ m. (minimum width or height must be greater or equal to 5 meters), and there was no restriction on the maximum width of a department (they can be a square). The cost per unit distance, from his work, already multiplied by the material flow, is given in Table 5.5. However, in the development of this research, the total area is to be calculated by height: width ratio of 1:2, which is mentioned in Chapter 3.

Table 5.4 Departmental Areas for Ten-department Problem

Department	1	2	3	4	5	6	7	8	9	10
Area (A_i)	238	112	160	80	120	80	60	85	221	119

Table 5.5 Cost of Material Flow: Ten-department Problem

Dept.	1	2	3	4	5	6	7	8	9	10
1	-	0	0	0	0	218	0	0	0	0
2		-	0	0	0	148	0	0	296	0
3			-	28	70	0	0	0	0	0
4				-	0	28	70	140	0	0
5					-	0	0	210	0	0
6						-	0	0	0	0
7							-	0	0	28
8								-	0	888
9									-	59.2
10										-

From the total area of the floor plan mentioned in Chapter 3, the total area required in the shop floor is the sum of all department areas multiplied by 200% allowance for the extra areas. So the total area required for this problem is:

$$\sum_{i=1}^{10} A_i = 238+112+160+\ldots+119$$

$$= 1275 \quad m^2$$

Based on a 200% allowance, the area = 2550 m^2

The dimensions are assumed to be = 2:1 (W: H)

Thus, the height = 35.7 m.

≈ 36 m., and

width = 72 m.

Now, the dimension and area of the shop floor is recalculated. Based on van Camp *et al.*'s paper (1991), it is assumed that for a valid layout, no department dimension could be narrower than 5 meters, and there is no restriction on the maximum width or height of a department. To maintain flexibility in assigning the height and width for a department the aspect ratio is introduced in this research. It should be noted that van Camp *et al.* (1991) did not incorporate the aspect ratio for guiding the dimension of any departments. To compensate for the lack of aspect ratio in their problem, the 5-meter restriction can be applied to calculate the upper bound of height (or width) of each department. For example, a pair of departments 1 and 2 have areas of 238 m^2 and 112 m^2 respectively. With the 5-meter restriction, the upper bound on the height (or width) of departments 1 and 2 can be computed as shown below:

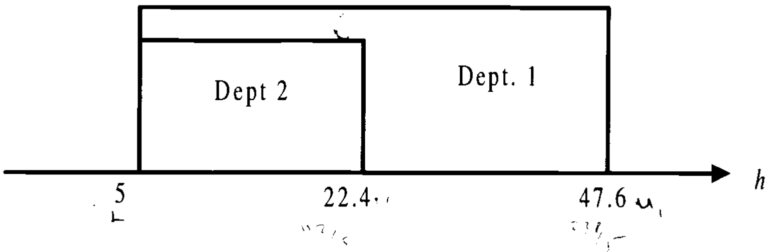


Figure 5.7 Illustration of the Heights (or Widths) of Departments 1 and 2 with the 5-meter Restriction.

From Figure 5.7, the maximum height (or width) of departments 1 and 2 is 22.4 and 47.6, respectively. In the proposed model, this is case 3 explained in the previous chapter. After performing the computation for all pairs, case 3 was found to dominate in all pairs of departments in this example problem.

Step 1: The strategies for finding the initial configuration

1. Normalize the traffic flow (material flow) in Table 5.5.

Table 5.6 Normalized Table 5.5

Dept.	1	2	3	4	5	6	7	8	9	10
1	-	0	0	0	0	0.1	0	0	0	0
2		-	0	0	0	0.07	0	0	0.14	0
3			-	0.01	0.03	0	0	0	0	0
4				-	0	0.01	0.03	0.06	0	0
5					-	0	0	0.1	0	0
6						-	0	0	0	0
7							-	0	0	0.01
8								-	0	0.41
9									-	0.03
10										-

2. Calculate the number of bays in this problem.

Number of bays (C) = INT ($\sqrt{N}$), N = Number of departments

$= \sqrt{10}$ (Round down)

$\approx$ 3 bays

Thus, one of the three bays has 4 departments and the other two bays have 3 departments each. Thus, the temporary formation for the number of the departments in the bays will be 4-3-3. However, the formation will be reevaluated in the following step.

3. Calculate the shape matrix by using Lindo software:

In the concept of bay configuration, each bay consists of limited number of departments, and the total department area in a bay will be the sum of any departments that are filled in that bay. For finding the total available areas in each bay, the greater the number of departments in the bay, the larger the areas that must be assigned. Thus, the size of each bay should approximately be proportional with the number of department in each bay. For example, say the summation of all departments is equal to 100 ft^2, and the first, second and third bay are assigned to have 4, 3 and 3 departments, respectively. ($\lfloor (10)^{1/2} \rfloor \approx 3$). The number of bays is close to the number of departments in each bay, so the temporary formation 4-3-3 is applied. Thus, the first bay will have 100/4 ft^2, second and third bay will have 100/3 ft^2. In order to minimize the difference between the sum of the department areas that are assigned in each bay and each bay area, the binary programming in Lingo software (see appendix A.1) is used. The data needed for finding the interaction between departments in order to create the shape matrix is area A_i.

For illustration of this example problem, the final result obtained from the Lingo program shows that the department 2, 3, 5 and 10 are in the first bay, department 1, 7 and 8 are in the second bay, and department 4, 6 and 9 are in the last bay (see Appendix A.2). Now, the interaction matrix due to the geometry of each department can be generated, and the formation of this problem is 4-3-3. If the temporary formation is not the same as the formation from the mathematical result, the formation from the mathematical result will indeed be the valid one to apply in the following steps. Coincidentally, the formation obtained from the Lingo program is the same as the temporary formation.

4. In the shape matrix, the 1/12 is the normalization of the geometric relationship rating. In this example problem, the total number of geometric relationship frequency (shape measure) between all department pairs is 12. For example, shape matrix 1 and 7 is 1/12 because departments 1 and 7 have the geometric relationship and are rated as 1.

5. Sum Table 5.6 and 5.7

6. Find the dissimilarity matrix by

$$A_{ij} \quad = \quad 1/(1 + \textit{the matrix from 5.})$$

7. Input the A_{ij} to a numerical clustering procedure to construct a dendogram.

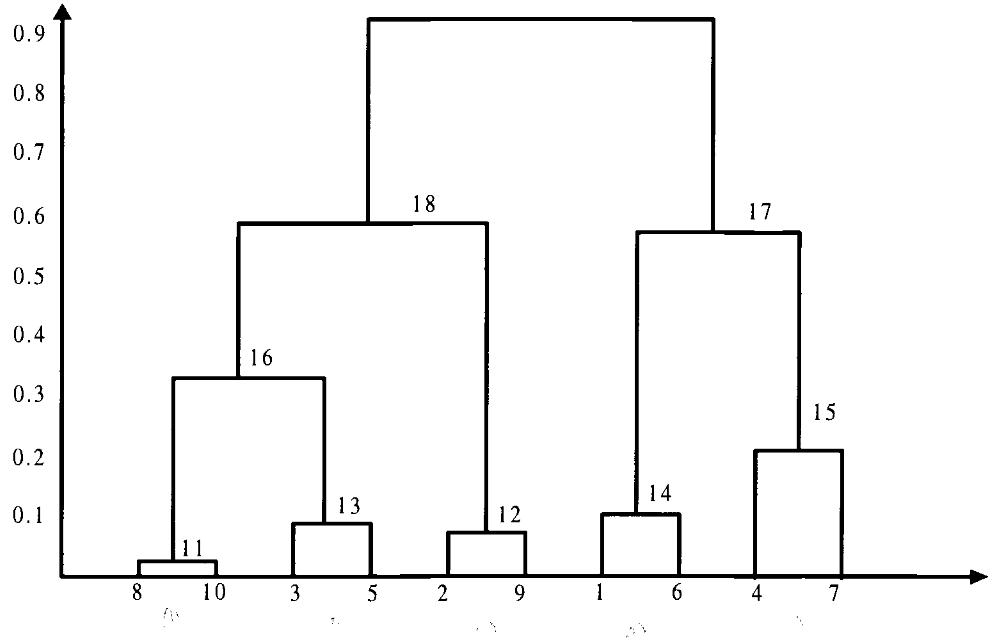
Table 5.7 Shape Matrix

Dept.	1	2	3	4	5	6	7	8	9	10
1	-	0	0	0	0	0	1/12	1/12	0	0
2		-	1/12	0	1/12	0	0	0	0	1/12
3			-	0	1/12	0	0	0	0	1/12
4				-	0	1/12	0	0	1/12	0
5					-	0	0	0	0	1/12
6						-	0	0	1/12	0
7							-	1/12	0	0
8								-	0	0
9									-	0
10										-

8. Assign the first three pairs that have the smallest average linkage (which means the highest interaction between departments, see appendix A.3). From Figure 5.8, department pairs 8 vs10, 2 vs 9, and 3 vs 5 are assigned in the first, second and third bays, respectively. The departments 1, 6, 4 and 7 are the rest.

9. After three pairs of departments are assigned, there are two available departments that can be assigned in the first bay and one department each in the second and third bay. The next pair that is filled in the initial layout should be 1 vs6. However, from the bay format 4-3-3, the first bay has 2 more unoccupied departments and the next pair of departments that will link with 8 vs 10 is 3 vs 5. So 3 vs 5 can be moved on the top of 8 vs 10, and the third bay can be filled with 1 vs 6.

Figure 5.8 Dendogram for Initial Layout



10. Now, the initial configuration can be writen as 8,10,3,5/2,9, _/1,6, _. Two departments (4 vs 7) are assigned next. The department that has the smaller index is chosen. The second bay has one more vacant space. Department 4 is smaller in index than 7, so department 4 is placed in the second bay and the last one (7) is placed in the last bay.

11. Finally, the initial configuration is represented as 8,10,3,5/2,9,4/1,6,7. Now, the initial solution (FS_0) can be explained by FSd_0 and FSb_0 as follows:

a. FSd_0 or the initial inside solution will have the following configuration: [8,10,3,5/2,9,4/1,6,7], which is a sequence of department-location identification. This sequence describes that first bay has departments 8, 10, 3 and 5, second bay has departments 2, 9 and 4, and finally the last bay has departments 1, 6 and 7.

b. FSb_O or the initial outside solution will have the following configuration: [1, 2, 3], which is a sequence of bay-location identification. This sequence describes that the first bay consists of departments 8, 10, 3 and 5 is located at the left most point of the floor plan. The second bay is placed next to first bay that is on the right of the first bay and the last third bay is placed next to second bay.

Step 2: Given the two configurations of the initial solution (FS_O), which are the FSd_O and FSb_O, apply the transformation technique in Section 5.4 in order to transform from a mixed-binary non-linear programming model to the controllable problem and solve it using Matlab 6 (Mathworks, 2000). Evaluate the objective function value (Z_0), where

$$Z_0 = \alpha[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} f_{ij}\, C_{ij}\,(x_{ij} + y_{ij})] + (1-\alpha)[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} (|\frac{A_i}{z_i} - \frac{A_j}{z_j}| + |z_i - z_j|)*D_{ij}]$$

$\alpha = 0.8$, $C_{ij} = 1$ and $D_{ij} = 1$ (assumed in the previous chapter);

$N = 10$;

A_i or A_j from Table 5.4;

f_{ij} from Table 5.5;

z_i or z_j is chosen from maximum value of the common width ranges.

x_{ij} or y_{ij} can be evaluated from a command in the MATLAB 6 (2000).

From the assumption that minimum width or height of each department must not be less than 5 meters (in this example problem), the maximum width of each department can be evaluated as:

Bay 1: Department 8, 10, 3 and 5: $Width_1$ = {32, 24, 17, 23.8}, respectively

Bay 2: Department 2, 9 and 4: $Width_2$ = {22.4, 44.2, 16}, respectively

Bay 3: Department 1, 6 and 7: $Width_3$ = {47.6, 16, 12}, respectively

Note: $Width_c$ = the set of widths in bay c, and c = 1, ..., C

The width of each bay can be evaluated by selecting the minimum of $Width_c$, From the above sets, the width of bay 1, bay 2 and bay 3 are equal to 17, 16 and 12 meters, respectively. At this point, all department sizes are known and substituted in the objective function to evaluate the objective function value.

All constraints in Chapter 4 are valid, except the constraint equations (11) and (12) which ensure that all departments are placed inside the floor plan are still present in the model. The objective function value of the initial solution can be evaluated. The program yields an objective function value (Z_0) of 22370. The value of Z_0 indicates that the solution is feasible. If the solution were infeasible, the value of Z_0 would be much greater than that. The infeasible solution indicates one or more of the total height in the bay exceed(s) the limitation on total height. For an infeasible solution, the penalty procedure will add a constant value (M1) that is sufficiently large in order to distinguish the infeasible solution from the feasible solution. In this example problem, M1 is assigned a value of 25000. For example, if the result detects that a solution is infeasible, it will immediately add 25,000 points to the objective function of the infeasible solution (i.e. 22370) and the initial Z_0 will become 47370.

Step 3: Given Z_0 of 22370, an inside search is performed to explore for a better solution. The outside search will pass the initial department configuration (FSd_0) to the inside search. The outside initial solution FSb_0 = [1,2,3] is the bay location configuration. This FSd_0 derives its initial configuration from the information supplied by step1. The inside search will use the FSd$_0$ as an initial node to perform inside perturbations. From step1, the configuration of FSd$_0$ for this example problem would be

[8,10,3,5/2,9,4/1,6,7].

Step 4: Using the inside initial solution (FSd$_0$) as a node, generate a set of seeds Sp (FSd$_0$) by using the inside perturbation. The procedure for inside perturbation is described earlier in Section 5.4. For this example problem, all possible interchanges (swaps) of two departments are considered. The configuration of the FSd$_o$ is converted to a layout by assigning the departments from bottom to top for each department in a bay and from left to right for each bay (see Figure 5.9). For starting the perturbation, a temporary fixed department, located at the bottom-left of the layout, is defined. So in this example problem department 8 is the fixed department.

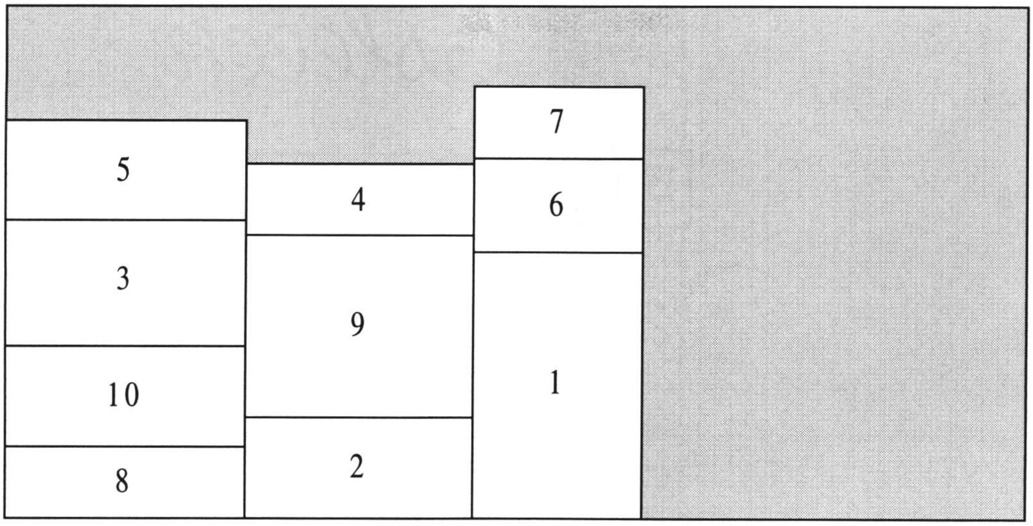


Figure 5.9 Layout for Initial Solution of Example Problem

The first swap move will be the interchange between the temporary fixed department and the department next to it, that is department 10. Then department 8 interchanges with department 3 and so on. After all departments have been interchanged with department 8, the next temporary fixed department will be reassigned to department 10, the swap moves will be continued until all departments are assigned to the temporary fixed department (except the last department in the layout which is department 7). The total number of swap moves will be formulated as:

$$Number\ of\ swap\ moves\ (m) = N!\ /\ (N\text{-}2)!\ {}^*2$$

So, there are 45 swap moves in this example problem.

Table 5.8 The Neighborhood Solutions of Initial Solution [8,10,3,5/2,9,4/1,6,7] as a Result of Applying Swap Moves

Swap Moves					
Swap departments	**Cost**	**Swap departments**	**Cost**	**Swap departments**	**Cost**
8 and 10	20800	3 and 5	20760	2 and 7	21350
8 and 3	19290	3 and 2	25460	9 and 4	23320
8 and 5	28280	3 and 9	25850	9 and 1	24470
8 and 2	36540	3 and 4	22820	9 and 6	Infeasible
8 and 9	32300	3 and 1	27240	9 and 7	23970
8 and 4	34000	3 and 6	Infeasible	4 and 1	23670
8 and 1	48480	3 and 7	23530	4 and 6	23860
8 and 6	44090	5 and 2	2504	4 and 7	22790
8 and 7	Infeasible	5 and 9	26500	1 and 6	21400
10 and 3	29420	5 and 4	24430	1 and 7	21840
10 and 5	31300	5 and 1	Infeasible	6 and 7	23630
10 and 2	32190	5 and 6	30370		
10 and 9	33430	5 and 7	Infeasible		
10 and 4	33490	2 and 9	21270		
10 and 1	Infeasible	2 and 4	21190		
10 and 6	46070	2 and 1	24840		
10 and 7	Infeasible	2 and 6	25110		

Step 5: Evaluate the objective function value (Z) of each seed using the same procedure outlined in step 2. Seed 1 of FSd_0 (Sd_1) has Z of 20800; seed 2 of FSd_0 (Sd_2) has Z of 19,290 and so on (See Table 5.8). Since Sd_2 has a smallest Z, Sd_2 is selected as the next parent node of the tabu search. The minimum cost is 19290. The move that results in this value is obtained by swapping departments 8 and 3. The layout generated by swapping department 8 and 3 would be used as the new seed for next iteration. Thus, before the inside search continues on perturbing the new parent node, the following parameters need to be updated:

(1) Inside Tabu List (in_TL)

The application of tabu list is to prevent the inside search from revisiting previous solutions or repeating its previous moves. Whenever an inside move is executed, in_TL is updated by admitting certain attributes into the list. In this example problem, the first move of inside search has just been performed. The move was performed as the result of swapping departments 8 and 3. Therefore, departments 8 and 3 are the first entry in the tabu list. The presence of departments 8 and 3 in the tabu list implies that these two departments are not allowed to swap positions for the number of iterations indicted by the size of the tabu list unless an aspiration criterion is satisfied. Two types of tabu list size are used: fixed in_TL and the variable in_TL. As mentioned earlier in Section 5.4, the parameters used for the inside tabu search will be closely related to the number of departments in the layout.

Proceeding with the list size of in_TL, for each type of in_TL, it is evaluated as follows:

$$\text{INT}(x) = \begin{cases} \lfloor x \rfloor, \text{ if x is a real number with a decimal value} < 0.5 \\ \lceil x \rceil, \text{ if x is a real number with a decimal value} \geq 0.5 \end{cases}$$

- For fixed in_TL = INT $(N/1.4)^{1/2}$ =INT $(10/1.4)^{1/2}$ = INT (2.67) or 3 as it is rounded up to its closest integer.
- For variable in_TL, there will be three sizes,
  - The initial size =INT $(N/1.4)^{1/2}$ =INT $(10/1.4)^{1/2}$ = INT (2.67) or 3 as it is rounded up to its closest integer.
  - The decreased size = INT $(N/(1.4*2))^{1/2}$ =INT $(10/(1.4*2))^{1/2}$ = INT (1.88) or 2 as it is rounded up to its closest integer.
  - The increased size =INT $(N/(1.4*0.5))^{1/2}$ = INT $(10/(1.4*0.5))^{1/2}$ =INT (3.77) or 4 as it is rounded up to its closest integer.

(2) Inside Aspiration Level (in_AL)

The aspiration level of the inside search is initially set equal to the objective function value of the inside initial solution (Z_0), which is 22370. So, in_AL is set to 22370 and it is updated when a smaller total cost is found during the inside search.

(3) Inside Candidate List (ICL) and Inside Index List (IIL)

As mentioned before, the initial department location configuration is admitted into the ICL. The new configuration obtained for this example is also admitted into both ICL and IIL, as it will be selected to perform future perturbations. Furthermore, the new configuration has a smaller total cost (19290) than the total cost of the initial configuration (22370). Thus, it is given a star, to indicate that it has the potential of becoming the next local optimal.

- ICL = {[8,10,3,5/2,9,4/1,6,7], [3,10,8,5/2,9,4/1,6,7]*}
- IIL = {[8,10,3,5/2,9,4/1,6,7]}

(4) Stopping Criteria

In order to terminate the inside search, the number of iterations without improvement (IWI) and the number of entries into the Inside index list (IIIL) are used simultaneously. The IWI is increased by one every time a non-improvement move is made. On the other hand, if for any inside iteration there is an improvement in total costs, the number of iterations without improvement for the inside search will be reset to zero.

For this example, evidently there is an improvement according to the first iteration. Therefore, the number of iterations without improvement for the inside search (IWI) is reset to zero.

The IIIL is increased by one every time an inside move is admitted into the IIL. Based on the preliminary experimentation, the IWI and IIIL are assumed proportional to the total number of departments in the floor plan. The formula used in the application of IWI and IIIL, as being applied to the example problem, is as follows:

- For fixed ITL,

  IWI = INT $(N*1.4)^{1/2}$ = INT $(10*1.4)^{1/2}$ = INT (3.74) or 4 as it is rounded up to its closest integer.

  IIIL = INT $(N*1.4*1.3)^{1/2}$ = INT $(10*1.4*1.3)^{1/2}$ = INT (4.26) or 4 as it is rounded down to its closest integer.

- For variable ITL,

  IWI = INT (0.56* (N*1.4) $^{1/2)}$ = INT (0.56* (10*1.4) $^{1/2}$) = INT (2.09) or 2 as it is rounded down to its closest integer.

  IIIL = INT (N*1.4*1.3) $^{1/2}$ = INT (10*1.4*1.3) $^{1/2}$ = INT (4.26) or 4 as it is rounded down to its closest integer.

The guideline for using IWI with variable ITL is as follows:

- If there is no improvement within the last IWI (2) iteration with the initial ITL (3), then decrease the ITL to the decreased size of ITL (2).
- If there is no improvement within the last IWI (2) iteration with the decrease ITL (2), then increase the ITL to the increased size of ITL (4).
- If there is no improvement within the last IWI (2) iteration with the increase ITL (4), then terminate the inside search.

At this point in the example problem, both stopping criteria are not satisfied yet because so far there is no non-improvement move (IWI=0), and one entry into the IIL (IIIL=1).

The results for the inside search with fixed tabu-list size for FSb_0 = [1,2,3] using FSd_0 = [8,10,3,5/2,9,4/1,6,7] as an initial layout solution configuration are shown in Table 5.9.

Table 5.9 Results Obtained for the Inside Search of FSd$_0$ = [8,10,3,5/2,9,4/1,6,7] as an Initial Configuration.

Iteration No.	Entries into ICL	Total Cost (Z)	Entries into IIL
0	8,10,3,5/2,9,4/1,6,7**	22370	Yes
1	3,10,8,5/2,9,4/1,6,7*	19292	
2	3,10,8,5/7,9,4/1,6,2*	18141	
3	3,10,8,5/7,4,9/1,6,2*	16512	
4	3,5,8,10/7,4,9/1,6,2**	15973	Yes
5	3,5,8,10/4,7,9/1,6,2	16291	
6	3,5,8,10/9,7,4/1,6,2	17450	
7	3,5,8,10/9,7,4/2,6,1*	16434	
8	3,5,8,10/9,4,7/2,6,1**	16067	Yes
9	3,10,8,5/9,4,7/2,6,1	16567	
10	5,10,8,3/9,4,7/2,6,1	17659	
11	5,8,10,3/9,4,7/2,6,1*	16861	
12	10,8,5,3/9,4,7/2,6,1**	16389	Yes
13	10,8,5,3/9,7,4/2,6,1	16750	

The inside search, starting with [8,10,3,5/2,9,4/1,6,7], is terminated after 13 iterations have been performed because one of the two stopping criteria has been reached. The number of entries into the Inside index list (IIIL) for the fixed tabu-list size is equal to 4 (IIL = 4). While, the number of iterations without improvement (IWI) has not been reached (IWI=4). The CL has 14 entries and IL has 4 entries. The best solution obtained by employing short-term memory function is found at the fourth iteration with a total cost of 15973. The best solution is pointing to the following layout: [3,5,8,10/7,4,9/1,6,2].

Step 6: To diversify the inside search performed in step 5, the inside long-term memory is implemented. The inside long-term memory (IN_LTM) is the frequency matrix that keeps track of the tenure of an option for each department throughout the inside search. Every time a new department configuration is constructed the entries in IN_LTM matrix corresponding to the departments and their respective options in the configuration are increased by one.

Originally, the entries in IN_LTM are all initialized to zero. After the first in_move is performed, from initial department configuration [8,10,3,5/2,9,4/1,6,7] to the next configuration [3,10,8,5/2,9,4/1,6,7], the IN_LTM would be updated as shown in Table 5.10.

Table 5.10 Entries into the IN_LTM matrix as FSd_0 has just been identified.

Department	1st bay	2nd bay	3rd bay
1	0	0	1
2	0	1	0
3	1	0	0
4	0	1	0
5	1	0	0
6	0	0	1
7	0	0	1
8	1	0	0
9	0	1	0
10	1	0	0

As the inside search progresses the IN_LTM frequency matrix is updated regularly. The corresponding IN_LTM frequency matrix for the inside search after the number of entries into the Inside index list (IIIL) has been reached in Step 5 is represented in Table 5.11.

Table 5.11 Entries into the IN_LTM matrix at the time of termination.

Department	1st bay	2nd bay	3rd bay
1	0	0	14
2	0	2	12
3	14	0	0
4	0	14	0
5	14	0	0
6	0	0	14
7	0	12	2
8	14	0	0
9	0	14	0
10	14	0	0

Using the information obtained from the IN_LTM frequency matrix, a restart configuration is generated. There are two types of restarts considered in this research: the restart that is based on maximal frequency (IN_LTM_MAX) and the restart that is based on minimal frequency (IN_LTM_MIN). The IN_LTM_MAX, which is intended to intensify the search, fixes the department to its respective location according to the maximal entry in the frequency matrix. On the other hand, the IN_LTM_MIN, which is intended to diversify the search, fixes the department to its respective location according to the minimal entry in the frequency matrix.

For example, the maximal entry in the IN_LTM frequency matrix is equal to 14, and it corresponds to the first bay of department 3, 5, 8 and 10, the second bay of department 4 and 9, and the third bay of department 1 and 6. The row-wise first best strategy is used to break ties. Therefore, the maximal entry of 14 according to the third bay of department 1 is used for generating the first new restart. From Table 5.11, out of

14 entries in third bay, department 1 has 7, 0 and 7 entries in the first, second and third position, respectively.

Table 5.12 Entries of Department 1 in the Third Bay

Department 1 in the third bay		
1 st position	2 nd position	3 rd position
7	0	7

The first best strategy is also used to break tie. Thus, the department 1 is fixed in the first position of the third bay in order to construct the new initial configuration for the next restart. The other departments are still assigned to the same location as they were in the initial configuration. As a result, the new initial configuration for the next restart is [8,10,3,5/2,9,4/1,6,7]. The underline indicates that department 1 in the third bay is now fixed throughout the next restarted search. The search for the next restart would be performed in a similar fashion according to the procedure described in step 5. The results obtained with the first long-term memory restart and the resulting IN_LTM are shown in Table 5.13 and 5.14, respectively.

Table 5.13 Results obtained for the inside search starting with the inside first restart configuration.

Iteration No.	Entries into ICL	Total Cost (Z)	Entries into IIL
0	8,10,3,5/2,9,4/1,6,7**	22370	Yes
1	3,10,8,5/2,9,4/1,6,7*	19292	
2	3,10,8,5/7,9,4/1,6,2*	18141	
3	3,10,8,5/7,4,9/1,6,2*	16512	
4	3,5,8,10/7,4,9/1,6,2**	15973	Yes
5	3,5,8,10/4,7,9/1,6,2	16291	
6	3,5,8,10/9,7,4/1,6,2	17450	
7	3,5,8,10/9,4,7/1,6,2**	17115	Yes
8	3,5,8,10/9,4,7/1,2,6	17573	
9	3,5,8,10/6,4,7/1,2,9**	17556	Yes
10	3,10,8,5/6,4,7/1,2,9	18200	

Table 5.14 Entries into the IN_LTM matrix at the time of termination (first restart)

Department	1st bay	2nd bay	3rd bay
1	0	0	11
2	0	2	9
3	11	0	0
4	0	11	0
5	11	0	0
6	0	2	9
7	0	9	2
8	11	0	0
9	0	9	2
10	11	0	0

From Table 5.14, the maximal entry into the frequency matrix has to be identified. In this case, it is found to be 11, which corresponds to department 3 in the first bay. Department 1 is skipped because it has been considered. Therefore, the maximal entry of

11 according to the first bay of department 3 is used for generating the second new restart. From Table 5.14, out of 11 entries in first bay, department 3 has 10, 0, 1 and 0 entries in the first, second, third and fourth position, respectively.

Table 5.15 Entries of Department 3 in the First Bay

Department 3 in the first bay			
1 st position	2 nd position	3 rd position	4 th position
10	0	1	0

The location of department 3 is swapped with department 8, due to the most frequency of department 3 is located in department 8's location (the first position). The next restart for the LTM_MAX would be [3,10,8,5/2,9,4/1,6,7]. Using the same approach, the results obtained with the second long-term memory restart are presented in Table 5.16.

Table 5.16 Results obtained for the inside search starting with the inside second restart configuration.

Iteration No.	Entries into ICL	Total Cost (Z)	Entries into IIL
0	3,10,8,5/2,9,4/1,6,7**	19292	Yes
1	3,10,8,5/7,9,4/1,6,2*	18141	
2	3,10,8,5/7,4,9/1,6,2*	16512	
3	3,5,8,10/7,4,9/1,6,2**	15973	Yes
4	3,5,8,10/4,7,9/1,6,2	16291	
5	3,5,8,10/9,7,4/1,6,2	17450	
6	3,5,8,10/9,7,4/2,6,1*	16434	
7	3,5,8,10/9,4,7/2,6,1**	16067	Yes
8	3,10,8,5/9,4,7/2,6,1	16567	
9	3,8,10,5/9,4,7/2,6,1	18071	
10	3,8,10,5/9,4,7/1,6,2	19119	
11	3,8,10,5/7,4,9/1,6,2**	17342	Yes
12	3,8,10,5/4,7,9/1,6,2	17524	

Table 5.17 Summary of final solutions obtained from the inside search with two long-term memory restarts based on LTM_MAX

Number of Restart	The Best solution in the IIL	Total Cost
Initial	3,5,8,10/7,4,9/1,6,2	15973
First Restart	3,5,8,10/7,4,9/1,6,2	15973
Second Restart	3,5,8,10/7,4,9/1,6,2	15973

For this problem instance, the LTM_MAX is not very effective in directing the search to a new region, which is truly the case here. Even though the long-term memory based on maximal frequency was not able to identify a better solution for this example problem, there is still another strategy based on minimal frequency that could be used.

If the LTM_MIN is applied in this example problem, the minimal entry in the frequency matrix has to be identified. In this case, it is found to be zero, which corresponds to all departments in this problem. There should be strategies to break the ties. First, from the frequency matrix, the department that has never been swapped or located in other bays (unique bay location) must not be considered to be the fixed department for the next starting solution. Skip the department that is assigned in only one bay. From the preliminary experiment, when the unique bay location department is swapped and fixed to the bay that has the minimum entry (0), it always leads to the infeasible solution. For the example, department 1 is located in the third bay only, so the department 1 should be skipped (refer to Table 5.11). After the first strategy is used, only departments 2 and 7 are left. Department 2 has 2 and 12 entries in the second and third bay, respectively. While department 7 has 12 and 2 entries in the second and third bay, respectively. Second, the row wise first best strategy is used to break the ties, department 2 is selected.

If the initial layout configuration (FSd_0) is [8,10,3,5/2,9,4/1,6,7], then the new restart configuration would be [8,10,$\underline{2}$,5/3,9,4/1,6,7], where department 2 has been placed in first bay. Department 2 has the least frequency (0) in the first bay and department 3 is the first least frequency (0) in the second bay. So department 2 has to swap with department 3. Here, department 2 is underlined to indicate that it is now fixed at first bay and throughout the rest of the search with the first restart. Notice that the least frequency is used here for LTM_MIN in contrary to the most frequency usage for LTM_MAX. Performing the search in a similar fashion as the OLTM_MAX, the results for the OLTM_MIN are presented in Table 5.18.

Table 5.18 Inside search results for the first restart based on minimal frequency.

Iteration No.	Entries into ICL	Total Cost (Z)	Entries into IIL
0	8,10,2,5/3,9,4/1,6,7**	25464	Yes
1	8,10,2,9/3,5,4/1,6,7*	22424	
2	8,10,2,9/7,5,4/1,6,3*	20731	
3	8,10,2,9/5,7,4/1,6,3*	20103	
4	8,10,2,9/5,7,4/3,6,1*	19646	
5	8,10,2,9/5,4,7/3,6,1**	19241	Yes
6	10,8,2,9/5,4,7/3,6,1	19284	
7	10,8,2,9/7,4,5/3,6,1	19454	
8	10,8,2,9/7,4,5/1,6,3	19545	
9	10,8,2,9/7,4,5/6,1,3	19826	

From Table 5.19, the minimal entry into the frequency matrix has to be identified. In this case, it is found to be 0, which corresponds to department 3 in the first bay. Thus, department 3 is swapped to one of the departments in the first bay (2, 5, 8 and 10). The least frequency (0) in the second bay is department 8 (department 2 is ignored because it has been considered, and department 8 is the first best). The location of department 3 is swapped with department 8, due to the least frequency of department 8 is located in the department 3's location (second bay). The next restart for the LTM_MIN would be [3,10,2,5/8,9,4/1,6,7]. Using the same approach, the results obtained with the second long-term memory restart are presented in Table 5.20, and the summary of final solutions obtained from the inside search with two long-term memory restarts based on LTM_MIN are presented in Table 5.21.

Table 5.19 Entries into the IN_LTM_MIN matrix at the time of termination (first restart).

Department	1st bay	2nd bay	3rd bay
1	0	0	10
2	10	0	0
3	0	2	8
4	0	10	0
5	1	9	0
6	0	0	10
7	0	8	2
8	10	0	0
9	9	1	0
10	10	0	0

Table 5.20 Inside search results for the second restart based on minimal frequency.

Iteration No.	Entries into ICL	Total Cost (Z)	Entries into IIL
0	3,10,2,5/8,9,4/1,6,7**	36477	Yes
1	3,10,8,5/2,9,4/1,6,7*	19292	
2	3,10,8,5/7,9,4/1,6,2*	18141	
3	3,10,8,5/7,4,9/1,6,2*	16512	
4	3,5,8,10/7,4,9/1,6,2**	15973	Yes
5	3,5,8,10/4,7,9/1,6,2	16291	
6	3,5,8,10/9,7,4/1,6,2	17450	
7	3,5,8,10/9,7,4/2,6,1*	16434	
8	3,5,8,10/9,4,7/2,6,1**	16067	Yes
9	3,10,8,5/9,4,7/2,6,1	16567	
10	3,8,10,5/9,4,7/2,6,1	18071	
11	3,8,10,5/9,4,7/1,6,2	19119	
12	3,8,10,5/7,4,9/1,6,2**	17342	Yes
13	3,8,10,5/4,7,9/1,6,2	17524	

Table 5.21 Summary of final solutions obtained from the inside search with two long-term memory restarts based on LTM_MIN.

Number of Restart	The Best solution in the IIL	Total Cost
Initial	3,5,8,10/7,4,9/1,6,2	15973
First Restart	3,10,2,5/3,9,4/1,6,7	19241
Second Restart	3,10,2,5/8,9,4/1,6,7	15973

Step 7: Now, the out_move is performed, similar to step 4 of the inside search. The out_move transforms a sequence of bay location configuration to another sequence of bay location in its seeds. The value of out_move and the aspiration criterion would also be investigated in the same fashion as those for the inside search.

From the previous steps in this example, the initial feasible bay location configuration is obtained which is [1,2,3]. This configuration transforms to a new bay location configuration [2,1,3] and [1,3,2] since they do not contribute to the lower total cost in its seeds (see Table 5.22). The perturbation of the bay configuration does not consider the inverse configuration of itself, because it would obtain the same solution. For example, the bay configuration [1,2,3] obtains the same solution as [3,2,1]. The department configurations for the bay location configuration [1,2,3], [2,1,3] and [1,3,2] are [8,10,3,5/2,9,4/1,6,7], [2,9,4/8,10,3,5/1,6,7] and [8,10,3,5/1,6,7/2,9,4], respectively. The results obtained for the outside search of each bay locations configuration are presented in Table 5.22.

Table 5.22 Results obtained for the outside search of each bay locations configuration in Sb (FSb_0).

The Bay Location Configuration in the Seeds of [1,2,3]	**The Department configuration Obtained for the Outside search**	**Total Cost**
[2,1,3]	[6,2,9/1,5,8,10/3,4,7]	16910
[1,3,2]	[3,5,8,10/7,4,9/1,6,2]	15973

Similar to the inside search (step 5), the following parameters for the outside search are also updated during the search process.

(1) Outside-tabu list (out_TL)

Consider the out_move in this example, which moves the initial feasible bay locations configuration [1,2,3] to the next configuration [2,1,3] and [1,3,2]. The bay, which is moved to the next adjacent location one at a time, would be admitted into the out_TL along with its original location. In this example problem, there is no improvement when the location of the bay is moved. Thus, there is no entry in the out_TL. For example, say [1,3,2] has the potential to entry into the out_TL, bay 2 along with its location (2) would be moved into the out_TL as the first entry.

out_TL = [$pos_2(2)$]

The interpretation of this entry in the out_TL is that bay 2 occupied location 2 in the most recent iteration and it has been moved to the next adjacent location (location3). The out_TL is updated regularly as the in_TL for the inside search. There are two types of out_TL are considered as well. The fixed tabu-list size and the variable tabu-list size are determined by the formulae stated previously. Nevertheless, it is not appropriate to consider the variable tabu-sizes because the number of bays in this problem is too small.

The fixed tabu-list size for the outside search is determined by the following formula.

- For fixed out_TL = (C-1)/2 = (3-1)/2 = 1
- For variable out_TL, there will be three sizes,
  - The initial size = $\lceil (C-1)/2*0.95 \rceil = \lceil (3-1)/2*0.95 \rceil$= 0.95 or 1 as it is rounded up to its closest integer.
  - The decreased size = $\lceil (C-1)/2.1 \rceil = \lceil (3-1)/2.1 \rceil$ = 0.95 or 1 as it is rounded up to its closest integer.
  - The increased size = $\lceil (C-1)/1.1 \rceil = \lceil (3-1)/1.1 \rceil$ = 1.81 or 2 as it is rounded up to its closest integer.

(2) Outside Aspiration Level (out_AL)

As for the inside search, the outside aspiration level (out_AL) is initially set equal to the objective function value of the inside initial solution (Z_0), which is 22370. So, in_AL is set to 22370. This value is obtained for the initial bay locations configuration [1,2,3]. As the outside search progresses the out_AL is updated if the total cost evaluated for the current configuration is found to be better than the best configuration found so far. Thus, out_AL is not updated in this problem.

(3) Outside Candidate List (OCL) and Outside Index List (OIL)

Similar to the inside search, the initial feasible bay locations configuration is admitted into both OCL and OIL. The next configuration is also moved into the OCL as it will be considered to perform future perturbations. As this configuration contributes to a lower total cost compared to the initial configuration, it is also given a star because it has the potential of becoming the next local optimum. For this example problem, the configuration of the first move will be admitted into the OCL. Since the total cost (Z) of the first move is not better than the previous cost (Z_0), it would not receive a star. So, the entries into the OCL and OIL are as follows:

- OCL = {[1,2,3], [1,3,2]}
- OIL = {[1,2,3]}

(4) Stopping Criteria

The number of iterations without improvement for the outside search is similar to the inside search procedure. The number of iterations without improvement for the outside search (OWI) is increased by one, if there is no improvement in the total cost relative to the recent out_move. However, if in any iteration there is an improvement in total cost, the number of iterations without improvement will be reinitialized to zero. In this example, the first out_move does not show an improvement in total cost (15973). Thus, the number of iterations without improvement (OWI) is equal to one.

The number of iterations without improvement is used as a stopping criterion to stop the outside search. The number of iterations without improvement for the outside search is determined by:

- For the fixed out_TL (notice that only the fixed tabu-list size is considered in this example), the outside search stopping criterion is determined by the number of iterations without improvement (OWI):

$OWI = \lfloor (C-1)/2*1.2 \rfloor = \lfloor (3-1)/2*1.2 \rfloor = \lfloor 1.2 \rfloor$ or 1 as it is rounded down to its closest integer.

The number of entries into the Outside index list (OIIL) is not used in this sample problem due to the different bay locations configuration (only 3 configurations) are not many as the department location configuration. The results obtained from performing the outside search are presented in Table 5.23.

Table 5.23 Results obtained for the outside search starting with FSb_0 = [1,2,3] as the initial bay location configuration.

Iteration No.	**Entries into OCL**	**Total Cost (Z)**	**Entries into OIL**
0	[1,2,3]**	15973	Yes
1	[2,1,3]	16910	
2	[1,3,2]*	15973	

The effect of bay locations in this example problem can be seen from the results presented in Table 5.23. Different bay location configurations can have a significant impact on evaluating different minimum total cost. Therefore, taking bay location into consideration can be beneficial in determining the best solution for the original problem. However, this example has only 3 bays. The three different bay location shows in Table 5.23 are the only distinguishable bay location configurations. As a result the outside search in this problem has been shortened. The application of long-term memory for the outside search is not implemented in this research because of the small number of bays. The number of bays in the large size of problem is still small compared with the number of departments. Thus, the direction of the search for the facility layout problem emphasizes the inside search rather than the out side. In other words, the long-term memory for the outside search does not enhance the potential of identifying new starting point, and the fundamental elements of intensification and diversification strategies of the outside search are already present in the short-term memory component of TS. The long-term memory for the outside search should be ignored in the layout problem.

6. RESULTS AND DISCUSSIONS

This chapter focuses on evaluating the comparative performance of six different algorithms of the tabu-search based heuristics (Table 6.1). The data generation, number of test problems, design of experiment, experimental results and discussion are included in this chapter. The number of the test problems is presented in Tables 6.2 - 6.4. The experimental results for each test problem structure obtained from applying each heuristic algorithm along with the CPU time are illustrated in Tables D.1-D.3 (Appendix D), for the small, medium and large problem structures, respectively. The results from the analysis of variance for each problem structure are presented in Table 6.5. Furthermore, the interpretations of the results, which compare the different means of the six algorithms, are evaluated by the pairwise comparisons. The pairwise comparison is a widely used procedure for comparing all pairs of treatment means that are the average total costs in this research.

In comparison to the small and medium problem, the complexity of the large problem is estimated to be many folds higher with regard to the computation time determined from the experiment. For example, most of the small problems are solved in less than two minutes, but the large problems have taken up to 5 hours of computation time to solve. In the real facility layout problem, the number of departments in a floor plan is normally not greater than 20 departments. Therefore, in this research, the number of test problems used for the small, medium and large problem structures will vary slightly. The operating characteristic curve in the statistical method is applied in order to determine the sample size (number of test problems of a specific size). The details of this application are explained later. Regardless the number of test problems used, the experiment for each problem structure will strictly follow the guidelines given by "Design of Experiment" (Montgomery, 1997). Accordingly, the objectives of this chapter can be stated as follows:

1. To analyze the performance of the six different tabu search-heuristics on each problem structure.
2. To analyze the impact of tabu search features, particularly the tabu list size and the long-term memory, on each problem structure.

Based on the features that have significant impact on the performance of tabu search, the tabu search-based heuristic can be implemented in six different algorithms. As mentioned in Chapter 5, the features considered in this research are the tabu list size and the application of long-term memory. Two types of the tabu list size can be applied: the fixed and the variable tabu list size. Also, two strategies can be used for the long-term memory, one based on maximum frequency and the other based on minimum frequency. In addition, each heuristic algorithm employs two levels of search, which are executed as the inside tabu search and the outside tabu search. Thus, the six different algorithms of tabu search-based heuristic are organized in Table 6.1.

6.1 Data Generation

To compare the performance of the six different tabu search-based heuristics, a single-factor experiment is constructed. In this case, the factor is characterized by each of

Table 6.1 The Six Different Algorithms of the Tabu Search-Based Heuristic.

Heuristic Type No.	**Inside search**		**Outside search**	
	Tabu List	**Memory**	**Tabu List**	**Memory**
TS1	Constant	Short	Constant	Short
TS2	Constant	Long-Min	Constant	Short
TS3	Constant	Long-Max	Constant	Short
TS4	Variable	Short	Variable	Short
TS5	Variable	Long-Min	Variable	Short
TS6	Variable	Long-Max	Variable	Short

the different tabu search-based heuristic and measured by the total cost evaluated. As the test problems used with each heuristic can be different, the experiment is conducted as a randomized complete block design using the test problems as blocks and the different tabu search-based heuristic as treatments. Otherwise, the influence of differences in structure of the test problems can contribute to identifying a difference in the performance of the heuristics. Using the randomized complete block design the difference can be wholly attributed to the difference in performance of each heuristic itself, and not the difference between test problems. In this research the size of test problem is divided into 3 sizes (problem structures) as follows:

(i) Small size problem is 5 to 10 departments

(ii) Medium size problem is 11 to 20 departments

(iii) Large size problem is 21 to 26 departments

The number of the test problems for each problem size used for the experiment will be illustrated later. The data needed in the experiment are generated using a randomization process. The procedure used in the randomization is outlined below:

(i) Set all the randomization processes to uniformly distributed random numbers. The random numbers will always take integer values.

(ii) Randomize the areas for each test problem between 20 and 80 square feet. In the medical facility (Fred Meyers, 1993), the area requirement for each facility in a first aid room varies approximately from 20 to 80 square feet.

(iii) Randomize the assignment of traffic flow between 0 and 10. Ten is the maximum number of travels from department *i* to department *j*. Zero means there is no travels or part movement in a specific pair of departments. After all traffic flows are assigned, the flow matrix is created automatically.

(iv) Assume the aspect ratio of all departments to be 0.5 for the lower bound and 2 for the upper bound. From Fred Meyers', (Plant Layout and Material Handling, 1993), the reasonable department shapes should have one side of department two times as long as the other side. The reason is that it would be impractical in real industry practice to have a department that is too wide or too narrow (aspect ratio is less than 0.5 or greater than 2).

The data generated by the randomization process for the unequal area facility layout problems are presented in Appendix B1, B2 and B3, for small, medium, and large problem structures, respectively.

6.2 Number of Test Problems

In any experimental design problem, a critical decision is the choice of sample size that determines the number of replicates to run. Obviously, if the experimenter is interested in detecting small effects, more replicates are required than if the experimenter is interested in detecting large effects. In this section, the operating characteristic curve is applied to determine the sample size (number of blocks). The operating characteristic curve is a plot of the type II error probability of a statistical test for a particular sample size versus a parameter that reflects the extent to which the null hypothesis is false. These curves can be used to guide the experimenter in selecting the number of replicates so that the design will be sensitive to important potential differences in the treatments. For more details on operating characteristic curve, the reader is advised to refer to the text by Montgomery (1997).

The probability of type II error of equal sample sizes per treatment (say),

$$\beta = 1 - P\,\{\text{Reject } H_0 | \, H_0 \text{ is false}\}$$

$$= 1 - P\,\{F_0 > F_{\alpha,\, a-1,\, N-a} | \, H_0 \text{ is false}\}$$

Operating characteristic curves given in the text by Montgomery are used to evaluate the probability statement in the equation above. These curves plot the probability of type II error (β) against a parameter Φ, where

$$\Phi^2 = \frac{b\sum_{i=1}^{a}\tau_i^2}{a\sigma^2}$$

Φ = Parameter from the operating characteristic curves

σ = Standard deviation

b = Number of test problems or blocks

μ_i = Treatment means i

$\bar{\mu}$ = $(1/a)\sum_{i=1}^{a}\mu_i$ = Average of the individual treatment means

τ_i = $\mu_i - \bar{\mu}$

a = Number of treatments

Curves in the text are available for α = 0.05 and 0.01 and a range of degrees of freedom for numerator and denominator. The parameter Φ must be specified for using the operating characteristic curves. Determining the parameter Φ is always difficult to do in practice. The use of the operating characteristic curves in this approach is not easy as it is usually difficult to select a set of treatment means on which the sample size decision should be based. The total costs in this research will increase when the number of departments increases. As a result, the standard deviation of the problem instances will also increase. To alleviate this difficulty an alternate approach for the calculation of the parameter Φ is introduced as follows:

$$\Phi = \sqrt{(1+0.01P)^2 - 1}(\sqrt{b})$$

P = Percentage for the increase in the standard deviation of an observation beyond which the model wish to reject the hypothesis that all treatment means are equal

The above equation can be found in the text by Montgomery (1997). The percentage for the increase in the standard deviation in this research is assumed to be acceptable in the range of 10%-40%. The larger the unequal area facility layout problem the higher is the standard deviation evaluated. For preliminary experimentations in this research, the acceptable percentages for small, medium and large problems are assumed to be no greater than 15%, 20% and 40%, respectively.

The procedure for determining the number of test problems is described next. Given the small problem structure has six heuristics (treatments, a) and a type II error probability of at least 0.095 (β), and $\alpha = 0.05$. With $a - 1 = 5$ and the number of blocks assumed equal to 11 ($b = 11$), the degrees of freedom can be evaluated as $(a\text{-}1)$ *$(b\text{-}1)$ = 5 *(10) = 50. From the operating characteristic curve with $\alpha = 0.05$, the parameter Φ is equal to 1.95. Finally, from the above equation, the percentage increase in standard deviation can be evaluated as 16.0%. The number of test problems (blocks) for each structure are shown in the table below:

Table 6.2. The Number of the Small Test Problems with the Power of 0.95

b	***(a-1)(b-1)***	**Φ**	**Percentage**
10	45	1.98	17.9%
11	50	1.95	16.0%
12	55	1.90	14.0%

Table 6.3 The Number of the Medium Test Problems with the Power of 0.93

b	***(a-1)(b-1)***	**Φ**	**Percentage**
5	20	2	34.10%
6	25	1.93	27.00%
7	30	1.81	21.30%
8	35	1.79	18.30%

Table 6.4 The Number of the Large Test Problems with the Power of 0.90

b	*(a-1)(b-1)*	**Φ**	**Percentage**
2	5	2.61	109%
3	10	2.18	60.70%
4	15	1.91	37.90%

For the large problem structure, the number of test problems has been reduced due to its extensive computation time. Most small problems are solved in less than five minutes, while some large problems have taken up to 8 hours to solve. This large variation in computation time between the two problem structures is mainly attributed to the differences in size of problem structures. The difference in the size of the problem structure has increased the search space of the large problem. This increase in search space has caused the search to consider more configurations before making a move and also, more moves are required before the search can be terminated. In fact, the larger the number of departments in a problem, the longer the computational time needed to identify the best solution. Thus, the percentage increase in standard deviation has been increased in order to decrease the number of the large test problems used in the experiment.

From the previous paragraph, the computational time and the standard deviation increase for the large problem structure are considerably higher than the small and medium problem structures. Thus, this research tends to reduce the number of test problems for the large problem structure, which is equal to 4 test problem instances. The normal probability plot for each problem structure, presented in Figures C.1-C.3

(Appendix C), is used to detect any departure from the assumption of normal distribution. The plots of the residuals show the severe indication of non-normality in all problem structures. Thus, the nonparametric method is introduced to analyze the experimental results.

The analysis of variance for nonparametric method will work accurately when the approximation of $a*b \geq 30$ (a = number of treatments, and b = number of test problems) is applied. In order to meet this requirement, this research assumes to increase the number of test problems for the large size by 1. Thus, the number of test problem for the large size is equal to 5, and $a*b$ = 72, 42 and 30, for the small medium and large size, respectively. Furthermore, the increment of the 4-test problem to 5-test problems for the large size problem leads to the power of test increase from 0.9 to 0.93. With the power at least 0.93 for all problem structures, they are quite adequate for analyzing the results from the experiment.

Finally, the number of test problems selected are 12, 7 and 5 for the small, medium and large size, respectively.

6.3 Design of Experiment

The procedure specified in the design of experiment, known as the single factor experiment, is employed in order to compare the performance of the heuristic algorithms. The factor in this case is characterized by each of the six heuristic algorithms and measured by the minimum total cost evaluated. The single factor experiment can be performed either as a completely randomized design or randomized block design. In a completely randomized design, it is normally assumed that the variability of results comes from a single source only. If two or more sources would affect the variability of results, they have to be blocked. Blocking these undesirable sources will increase the accuracy of the results as well as improve the sensitivity of the comparison. This blocking capability is provided by the randomized block design. Since the experiment performed here can also be affected by the structure of the test problems, the randomized block design is employed instead of the completely randomized design. Recall that each problem structure will be experimented with several test problems. Therefore, if a difference in the performance of the heuristics is identified, it can be totally attributed to the difference in the heuristics and not the difference between test problems. For further details on completely randomized design and randomized block designs, refer to the text by Montgomery (1997). In this research, the analysis of variance is preformed to find a significant difference among the total costs obtained for the test problems with the six heuristics. The significance level α, also referred to as type I error, is assumed equal to 5%. Due to the non-normality of the data distribution, parametric methods such as F-test and t-test are not appropriate for analyzing the experimental results.

The alternative to F-test and t-test are non-parametric methods known as Friedman test and Wilcoxon signed-rank test. Friedman test is useful to check if there is any significant difference between the treatment levels (TS1-TS6). If there is an evidence of significant difference between the heuristics, Wilcoxon signed-rank test will be applied to identify which heuristic performed distinguishably better than the rest. In the Friedman test, there is an ordering of the treatments, one tending to produce the lowest responses, another the next lowest, and so on. An indication of the position of the *a*th treatment in this ordering is provided by the average rank:

$$R_{a.} = (R_{a1} + \ldots.. + R_{ab})/b$$

where, R_a = Ranks in the *i*th treatment

b = Number of test problems or blocks

The $R_{i.}$ is substituted in the Friedman statistic, which is provided in the text by Lehmann (1975). Moreover, there are ties among the observation within the block. The application of midrank method must be used. A detailed description on the application of midrank method can be found in the text by Lehmann (1975) also.

6.4 Experimental Results and Discussion

The experimentation in this research is performed on a Pentium II 300 MHz with 192 MB RAM. The experimental results for each test problem obtained with each heuristic are presented in Table D.1-D.3 (Appendix D). The analysis of variance for each problem structure is evaluated by the application of Friedman test. The comparisons of the results for the average total costs along with the Wilcoxon signed-rank test for each problem structure is shown in Table E.1 (Appendix E). Moreover, to suggest the user for choosing the best heuristic among six versions of tabu search based-heuristic, the pairwise comparisons of the different memory functions and the different tabu list sizes are applied.

For each size of problem structure, Friedman tests are applied to test the hypothesis as stated below:

H_0: There is no difference in the total cost obtained for the problem instances using the six versions of tabu search-based heuristics (TS).

H_1: At least one of the tabu search based-heuristics tends to yield smaller the total cost than the others.

The results of Friedman test are summarized in Table 6.5. With α = 0.05, there is no significant difference between the six-tabu search heuristics for all sizes of problem instances.

Table 6.5 Summary of Results from Friedman Tests

Heuristic Algorithms	**Average Total Cost**		
	Small **5-10 depts.**	**Medium** **11-20 depts.**	**Large** **21-26 depts.**
TS1	1204.27	7327.04	20347.8
TS2	1218.17	7336.51	20426.6
TS3	1206.73	7351.49	20298.8
TS4	1204.6	7304.43	20242.4
TS5	1212.66	7308.61	20300.4
TS6	1205.09	7317.87	20251.8
Significant Difference?	No at α = 0.05	No at α = 0.05	No at α = 0.05
Test Statistics	10.805	7.488	10.528
p-value	0.0553	0.1868	0.0615

Since there is no significant difference between the heuristics (p-value of the tests are > 0.05) for all sizes of problem structure, then the pairwise comparisons are conducted only between the heuristics of tabu search. The comparisons between the treatment means of each heuristic are necessary in order to identify which heuristic of tabu search performs significantly better. This is done by applying Wilcoxon signed-rank tests on the average total cost between different heuristics of tabu search.

For the small problem, there is no significant difference among the six heuristics at $\alpha = 0.05$. However, the p-value is very close to rejecting the null hypothesis (p-value = 0.0553). Thus rather than concluding that all six heuristics are equally good, the smallest total cost is selected in order to identify the best heuristic. TS1 is evaluated as the one having the best total cost of 1204.27. At this point TS1 appears to be very attractive. During the design of layout problem, significant effort must be expended to find a good quality solution. Therefore, TS1 is recommended for the small problem structure. However, in comparison to the second best performer (TS4), TS1 is slightly better. TS1 has average total cost of 0.02% less than that of TS4.

The results obtained for the medium problem structure show that there is no significant difference among the six heuristics at $\alpha = 0.05$. Similar to the small problem, the blocking effect in experimentation with the medium problem has proven to be useful. Wilcoxon signed-rank tests are performed on the six heuristics and the results are summarized in Appendix E. It is observed that TS4 is significantly better than TS5, TS6, TS1, TS2 and TS3. TS4 is approximately 0.6% better than TS3 the worst heuristic in average total cost. Thus, TS4, the heuristic with the best average total cost, is recommended for the medium structure.

For the large problem structure shown in Table E.1 (Appendix E) presented that there is no significant difference among the six heuristics at $\alpha = 0.05$. Similar to the small problem, the p-value for the large problem structure is close to 0.05 (p-value = 0.0615). Comparing the best and the worst heuristic, TS4 has the smallest total average cost. TS4 is approximately 0.9% better than TS2 the worst heuristic in average total cost. Thus, TS4 is recommended for the large problem structure.

6.4.1 The Use of Long-Term Memory in Tabu Search-Based Heuristics

Four heuristics, TS2, TS3, TS5 and TS6, have employed the use of long-term memory, while TS1 and TS4 have only employed the use of short-term memory. With the intention of a fair comparison between heuristics that employed the long-term memory and those that did not, two groups of comparison have been made. In the first group TS2 and TS3 are compared to TS1 that are restricted to the heuristics that employed fixed tabu list size. The second group is the comparison that is restricted to the heuristics that employed variable tabu list sizes. In this group, TS5 and TS6 are compared to TS4. Two types of measurements could be used for the comparison: one based on the Wilcoxon signed-rank test and the other based on the numerical difference test. The Wilcoxon signed-rank test applies the result obtained from Appendix E to perform the comparison, while the numerical difference test compares the numerical differences between the average total costs of the two heuristics. The results obtained from the comparison according to each type of measurement are presented in Table 6.6, and can be interpreted as follows:

- A "-" sign means that there is no significant or numerical difference between the two heuristics.
- A "Yes" means that the first heuristic performs better than the second heuristic, and a "No" means vice versa.

Table 6.6 Comparison of the Heuristics that use Long-Term Memory and those that Use Only Short-Term Memory

Measurement	**Size of Tabu List**	**Comparison**	**Problem Size**		
			Small	**Medium**	**Large**
Wilcoxon Signed-Rank Test	Fixed	TS1 & TS2	Yes	-	Yes
		TS1 & TS3	-	-	-
	Variable	TS4 & TS5	Yes	-	-
		TS4 & TS6	-	-	-
Numerical Differences	Fixed	TS1 & TS2	Yes	Yes	Yes
		TS1 & TS3	Yes	Yes	No
	Variable	TS4 & TS5	Yes	Yes	Yes
		TS4 & TS6	Yes	Yes	Yes

From Table 6.6, based on Wilcoxon signed-rank test, none out of the 12 comparisons show that the heuristics with long-term memory are significantly better than the ones without it. In agreement with the results obtained from Wilcoxon signed-rank test, the test based on the numerical difference also shows that the heuristics with long-term memory have a better average total cost in 1 out of 12 comparisons. Thus, it can be concluded that the addition of long-term memory in tabu search does not improve the search to identify a better solution. Although the long-term memory has produced a better average total cost in some case attempted, it is not capable of improving the search in most cases.

As noted in the previous chapter the long-term memory can be divided into 2 types (maximal frequency and minimal frequency). For the comparison of the use of long-term memory based on maximal frequency (LTM_MAX) with the use of long-term memory based on minimal frequency (LTM_MIN), the two types of measurement described above are used again. In this comparison, all of the heuristics that use

LTM_MAX will be compared to the heuristics that use LTM_MIN. The results of the comparison are presented below with the same interpretation as before.

From Table 6.7, based on Wilcoxon signed-rank test, there are three comparisons the favor the heuristics that use LTM_MAX in contrast to none comparison that favors the heuristics that use LTM_MIN. This shows that the LTM_MAX is significantly better than the LTM_MIN. As for the numerical difference test, there are 9 comparisons that favor the heuristics that use LTM_MAX in contrast to only 3 comparisons that favor the use of LTM_MIN. From the results, both Wilcoxon signed-rank test and the numerical difference test indicate that the LTM_MAX has resulted in a better average total cost than the LTM_MIN. Therefore, it can be concluded that the use of long-term memory based on maximal frequency strategy would be preferred than the use of long-term memory based on minimal frequency strategy.

Table 6.7 Comparison of the Heuristics that use LTM_MAX and LTM_MIN

Measurement	**Comparison**	**Problem Size**		
		Small	**Medium**	**Large**
Wilcoxon Signed-Rank Test	TS2 & TS3	No	-	-
	TS2 & TS6	No	-	No
	TS5 & TS3	-	-	-
	TS5 & TS6	-	-	-
Numerical Differences	TS2 & TS3	No	Yes	No
	TS2 & TS6	No	No	No
	TS5 & TS3	No	Yes	No
	TS5 & TS6	No	Yes	No

6.4.2 The Use of Tabu-List in Tabu Search-Based Heuristics

There are two types of tabu list size, which are the fixed tabu list size and variable tabu list size. Of the six heuristics, TS1, TS2 and TS3 employed the fixed tabu list sizes while TS4, TS5 and TS6 employed the variable tabu list sizes. Similar to the comparison performed for the long-term memory, three groups of comparison and two types of measurement are used to compare the performance of fixed versus variable tabu list sizes in tabu search-based heuristics. The first group consists of the comparison among the heuristics that use only the short-term memory. The second group consists of the comparison among the heuristics that use only short-term memory with LTM_MIN. And, the third group consists of the comparison among the heuristics that use long-term memory with LTM_MAX. The two types of measurement are the same as before.

Table 6.8 Comparison of the Heuristics that Use Fixed and Variable Tabu-List Sizes

Measurement	**Memory Feature**	**Comparison**	**Problem Size**		
			Small	**Medium**	**Large**
Wilcoxon Signed-Rank Test	Short	TS1 & TS4	-	No	No
	Long-min	TS2 & TS5	-	No	No
	Long-max	TS3 & TS6	-	-	-
Numerical Differnces	Short	TS1 & TS4	Yes	No	No
	Long-min	TS2 & TS5	No	No	No
	Long-max	TS3 & TS6	No	No	No

Based on Wilcoxon signed-rank test, from Table 6.8, 4 out of the 12 comparisons show that there is significant difference between the heuristics that used fixed tabu list size and variable tabu list size. Contrary to the results obtained from the Wilcoxon signed-rank test, the numerical difference test shows that there are 11 comparisons that favor the use of variable tabu list sizes and only 1 comparison that favor the use of fixed tabu list size. This result indicates that the use of variable tabu list sizes has resulted in a better average total costs than the use of fixed tabu list sizes. Generally, the results obtained from the Wilcoxon signed-rank test are more important than those from the numerical difference test. Two reasons are given for the above claim. First, Wilcoxon signed-rank test is based on statistical analysis, which takes into account the variation that could possibly be introduced in the results of the heuristics. Wilcoxon signed-rank test is certainly more accurate than the numerical difference test in performing comparison between heuristics. Second, based on the numerical difference test, the comparisons between two types of heuristics that favors fixed or variable tabu list sizes are very close to one another. With just one comparison difference between the two types of heuristics, either one could come up as a winner with a slight margin. Therefore, the results obtained from the Wilcoxon signed-rank test would be used. From the results, both Wilcoxon signed-rank test and the numerical difference test indicate that the variable tabu list size has resulted in a better average total cost than the fixed tabu list size. Therefore, it can be concluded that the use of the variable tabu list size would be preferred than the use of fixed tabu list size.

In conclusion, the experimental results expose that the use of short-term memory is essential to the tabu search algorithm developed in this research. It is suggested that the application of variable tabu list size has resulted in a better solution than the application of fixed tabu list sizes. It is also observed that the application of short-term memory has the resulted in a shorter computational time than the long-term memory. Therefore, TS4 which is the tabu search-based heuristic with variable tabu list size and short-term memory, is recommended for solving the unequal area facility layout problem considered in this research.

7. PERFORMANCE COMPARISON BETWEEN CURRENT AND PREVIOUS RESERCH

This chapter illustrates the comparison of solutions obtained in this research with those by previous researchers. All problems applied the tabu search-based heuristic for finding the best solution. The tabu-search is used as a mechanism to assess the quality of the solution determined from the heuristic algorithm developed in Chapter 5. For the equal area department problem, Nugent et al. (1967) reported that it is difficult to identify the global optimal solution, although small problems with 6, 7 or 8 departments have been solved. So, the comparison method is applied for finding the superior solutions. As mentioned in Chapter 4, the mathematical model for the unequal area layout investigated here belongs to the NP-hard class. Therefore, even for a small problem, computationally it would be very difficult to find its optimal solution. Thus the solutions obtained by previous researchers' are compared with that obtained from the tabu-search heuristic, for the same problem instances to access the effectiveness of the latter.

7.1 Data Sets from Previous Researchers

Tabu-search based heuristic is applied to the well-known unequal area facility layout problems previously considered by other researchers. These include the 5-deparment problem originally proposed by Tam and Li (1991), 10-department problem proposed by van Camp *et al.* (1991), 12-department problem proposed by Bazaraa (1975), and 30-department problem by Tam (1992). Hon-Iden (1996), who originally introduced the geometric shape parameter in the unequal area layout problem, slightly modified the shape restriction imposed by these researchers in order to make it correspond to his methodology. For the purpose of comparison, this research uses all of the restrictions that were introduced by Hon-Iden (1996). The modifications are as follows:

- For the van Camp's data, it is assumed that the minimum width or height of the facility is 5 meters.
- For the Bazaraa's data, it is assumed that the minimum width or height of the facility is 1 unit, and non-rectangular departments are invalid.

7.2 Heuristic Algorithms and Objective Function

Among the 6 different versions of the tabu-search based heuristic algorithm, TS4 has reported the best performance. Thus, TS4 is chosen to test the well-known unequal area facility layout problems. The performance of the six different tabu search-based heuristics has been illustrated in the previous chapter.

For a fair comparison with well-known problems, only the first term of the objective function ($\alpha = 100\%$) proposed in equation (0) in Chapter 4 must be considered. The second term that must be taken out is the shape cost, as it had never been considered in the past.

7.3 Results and Final Layouts

For all four test problems, the rectilinear distance between the department pairs is selected to compare the objective function values. The distance cost or material-handling cost obtained is presented as follows:

Table 7.1 Comparison Results with the Past Researches

Number of Departments	**5**	**10**	**12**	**30**
Bazaraa (1975)	-	-	14029	-
van Camp et.al (1991)	-	24445	11910	-
Tam and Li (1991)	127.28	-	-	-
Tam (1992)	-	-	-	26825
Hon-Iden (1996)	-	25126	11625	-
Proposed Method	**112.21**	**21142**	**10286**	**20849**
Percentage Improvement	**11.84%**	**13.51%**	**11.51%**	**22.27%**

The results obtained with the proposed method are better than the previously reported results on all test problems, indicating that the tabu search based-heuristic is an effective method for solving the unequal area facility layout problem. The results and the percentage improvements for each problem are shown in Table 7.1. It is observed that the increased number of departments in the problem has resulted in a higher percentage improvement. However, the unusable areas or empty spaces are unavoidably created at the top of each bay, which are the result from the unequal area departments. The final layouts are shown in Figures 7.1-7.4.

Figure 7.1 Five-department Problem

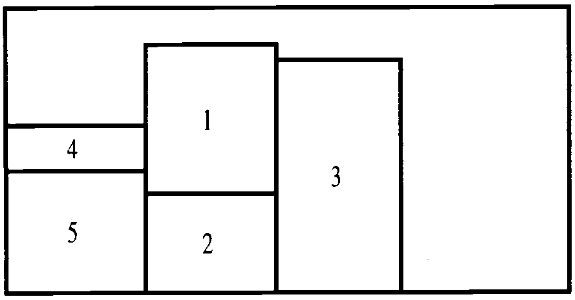


Figure 7.2 Ten-department Problem

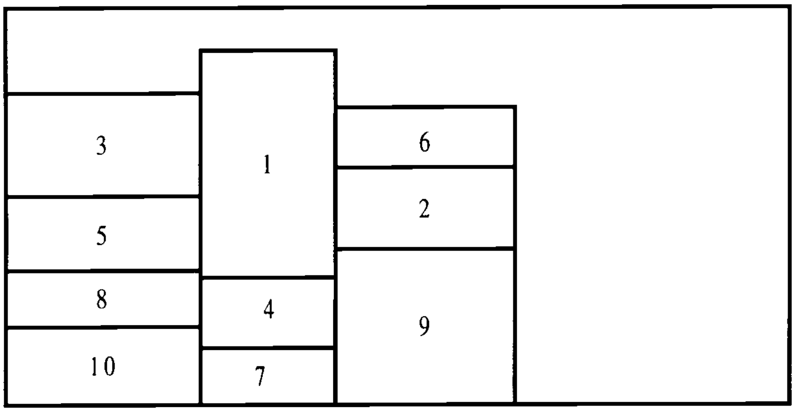


Figure 7.3 Twelve-department Problem

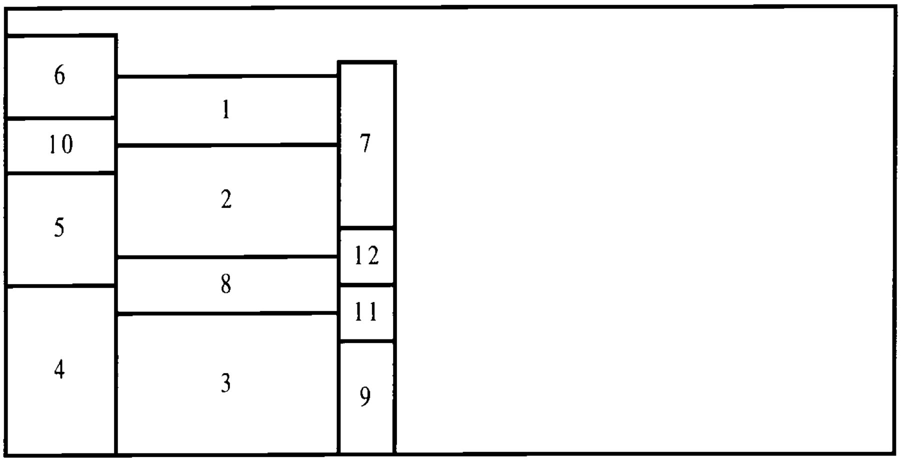
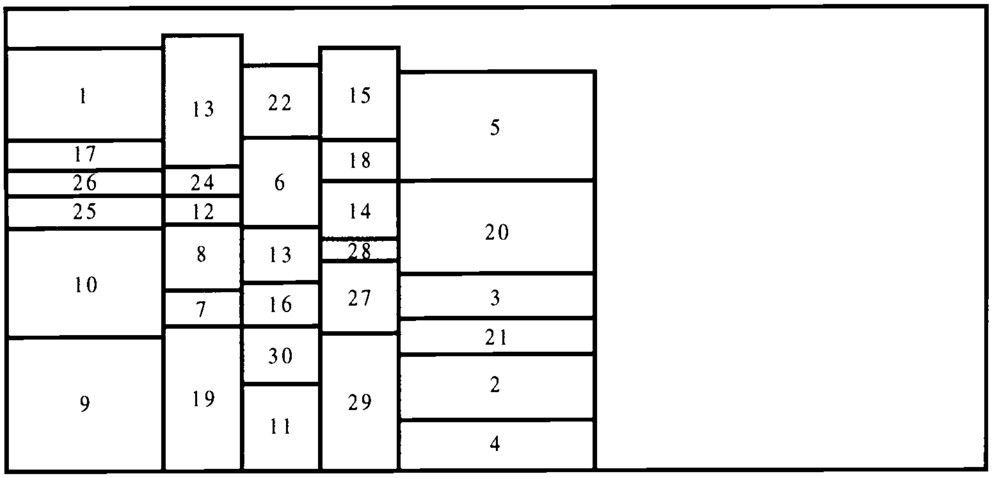


Figure 7.4 Thirty-department Problem



It should be noted that Tam and Li's (1991) method could not be compared to those by other researchers because their objective function is very different from the well-known objective function that is based on distance measure. Thus, a hand-drawn layout is used to measure the distance between departments and to perform an unbiased comparison. Figure 7.5 shows the final layout so established for Tam and Li' s problem. The objective function value of Tam and Li's (1991) 5-department problem, based on distance measure, is evaluated as 127.28 (in Table 7.1).

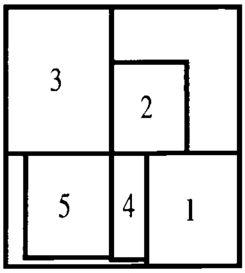


Figure 7.5 The Final Layout for Tam and Li's Problem.

8. CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

Over the years, the research on facility layout has evolved from manually assigning facility layout alternatives to the use of a computer to facilitate the generation of the facility layouts. However, the currently available computer-based layout algorithms cannot replace human judgment and experience, and they generally do not capture the qualitative characteristics in laying out of the departments. These include electric wire setting, piping structure, obstructed column, satisfaction or pleasing layout, etc. The computerized layout algorithms can significantly enhance the productivity of the layout planner and the quality of the final solution by generating and numerically evaluating a large number of layout alternatives in a very short time. Thus, the computerized layout algorithms are a powerful method to assist the layout planner in decision-making. Numerous researchers presented various facility layout algorithms and models that they believed are suitable for capturing the operational constraints in a real facility layout problem. But most of these approaches did not consider one of the important design factors: geometry or shape of the department. Hence, their approaches failed to reflect the needs of the real facility layout. This research proposes a methodology by incorporating both distance and shape-based measures in the unequal area facility layout problem. Consequently, it provides valuable insight into the investigation of facility layout related issues.

The unequal area facility layout problem is formulated as a binary non-linear programming model and is proven to be NP-hard in the strong sense. This rules out the possibility of employing an implicit enumeration-based technique to determine the optimal solution even on problems with moderate size. A higher-level heuristic solution algorithm, based on a concept known as tabu search, is proposed to efficiently solve the problem. The tabu search is implemented on two levels with the outside tabu search operating as the navigator for the entire search, while the inside tabu search makes minor adjustments to the search process for optimal performance. In this research, six different versions of the tabu search-based heuristic algorithm are tested on three different problem structures.

A single factor experiment based on randomized block design has been used to compare the performances of the six different heuristics (TS1-TS6) using the total cost as the criterion. The number of test problems for each problem structure are determined by applying the operating characteristic curves. For the small, medium and large problem, TS1, TS4 and TS4, respectively are recommended. The slight difference (0.02%) of the average total cost between TS1 and TS4 was found in the experiment of small size problem. Thus, TS4 might be recommended for solving all problem structures in the unequal area facility layout problem.

Further research can be performed by incorporating other special cases of equal area facility layout, such as the multi-floor layout problem, the locations of fixed or occupied departments in the layout, the three dimension-based distances, dynamic facility layout, etc. These special cases automatically become much more complicated problems when the unequal area issue is applied to them. This work may also be extended to include the shape cost in the objective function for determining the effect of different size of departments in the unequal area facility layout problem.

BIBLIOGRAPHY

Armour, G. C., and Buffa, E. S., 1963, A heuristic algorithm and simulation approach to relative location of facilities, *Management Science*, 6: 294-309.

Bazaraa, M. S., 1975, Computerized Layout Design: A branch and bound approach, *AIIE Transactions*, 7 (4): 432-438.

Bazaraa, M. S., and Kirca, O., 1983, A branch-and-bound-based heuristic for solving the QAP, Naval Research Logistics Quarterly, 30: 287-304.

Bhatnagar, R., 1989, An efficient heuristic procedure for plant layout analysis, Master Thesis, University of Toronto, Toronto.

Bozer, Y. A., and Meller, R. D., 1997, A reexamination of the distance-based facility layout problem, *IIE Transactions*, 29: 549-560.

Burbidge, J. L., 1979, *Group Technology in the Engineering Industry* (London: Mechanical Engineering Publications).

Burbidge, J. L., and Dale, B. G., 1984, Planning the introduction and predicting the benefits of group technology, *Engineering Costs and Production Economics*, 8: 117-128.

Burkard, R. E., and Stratman, K. H., 1978, Numerical investigations on quadratic assignment problems, *Naval Research Logistics Quarterly*, 25: 129-144.

Chiang, W. C., and Kouvelis, P., 1996, An improved tabu search heuristic for solving facility layout design problems, *International Journal of Production Research*, 34 (9): 2565.

Chiang, W. C., and Chiang, C., 1997, Intelligent local search strategies for solving facility layout problems with the quadratic assignment problem formulation, *European Journal of Operational Research*, 106: 457-488.

Das, S. K., 1993, A facility layout method for flexible manufacturing systems, *International Journal of Production Research*, 31 (2): 279-297.

Francis, R. L., McGinnis, L. F., and White, J. A., 1992, *Facility Layout and Location: An Analytical Approach,* 2nd ed. (Englewood Cliffs, New Jersey: Prentice Hall).

Francis, R. L., and White, J. A., 1974, *Facility Layout and Location: An Analytical Approach* (Englewood Cliffs, New Jersey: Prentice Hall).

Fred, E. M., 1993, *Plant Layout and Material Handling*, (New Jersey: Prentice Hall).

Gallagher, C. C., and Knight, W. A., 1986, *Group Technology and Production Methods in Manufacture* (Chichester: Ellis Horwood).

Glover, F., 1986, Future paths for integer programming and links to artificial intelligence, *Computers and Operations Research*, 13: 533-549.

Glover, F., 1989, Tabu search-Part I, *ORSA Journal on Computing*, 1: 190-206.

Glover, F., 1990a, Tabu search-Part II. *ORSA Journal on Computing*, 1: 4-32.

Glover, F., 1990b, Tabu search: A tutorial, *Interfaces*, 20: 74-94.

Glover, F., and Laguna, M. M., 1992, Tabu search, *In Modern Heuristic Techniques for Combinatiorial Problems*, C. R. Reeves (ed.) (New York: Halsted Press).

Kim, J. G., Lee G. C., and Kim Y. D., 1997, A study on facility layout planning using graph theory, 741-744.

Kim, J. Y., and Kim, Y. D., 1995, Graph theoretic heuristics for unequal-sized facility layout problems, *Omega*, 23: 391-401.

Koopmans, T. C., and Beckmann, M., 1957, Assignment problems and the locations of economic activities, *Econometrica*, 25: 53-76.

Kusiak, A., and Heragu, S. S., 1987, The facility layout problem, *European Journal of Operational Research*, 28: 229-251.

Kusiak, A., and Heragu, S. S., 1988, Machine Layout Problem in Flexible Manufacturing Systems, *Operations Research*, 36 (2): 258-268.

Kusiak, A., and Heragu, S. S., 1990, Machine Layout: An Optimization and Knowledge-Based Approach, International Journal of Production Research, 28 (4): 615-635.

Lehmann, E. L.,1975, *Nonparametrics: Statistical Methods Based on Ranks*, (San Francisco, Holden-Day, Inc.).

Meller, R. D., and Gau, K. Y., 1996, The facility layout problem: Recent and Emerging Trends and Perspecitves, *Journal of Manufacturing Systems*, 15 (5): 351-366.

Montgomery, D. C., 1997, *Design and Analysis of Experiments*, 4th ed. (John Wiley & Sons, Inc.).

Moore, J. M., 1974, Computer aided facilities design: an international survey, *International Journal of Production Research*, 12 (1), 21-44.

Nugent, C. E., Vollmann, T. E., and Ruml, J., 1968, An experimental comparison of techniques for the assignment of facilities to locations, *Operations Research*, 16: 150-173.

Pierce, J. F., and Crowston, W. B., 1971, Tree-search algorithms for quadratic assignment problems, *Naval Research Logistics Quarterly*, 18: 1-36.

Rajasekharan, M., Peters, B. A., and Yang, T., 1998, A genetic algorithm for facility layout design in flexible manufacturing systems, *International Journal of Production Research*, 35(1): 95-110.

Rosenblatt, M. J., and Golany, B., 1992, A distance assignment approach to the facility layout problem, *European Journal of Operational Research*, 57: 253-270.

Russell, D. M., and Kai-Yin, G., 1996, The Facility Layout Problem: Recent and Emerging Trends and Perspectives, *Journal of Manufacturing Systems*, 15 (5): 351-366.

Sahni, S., and Gonzalez, T., 1976, P-complete approximation problem, *Journal of ACM*, 23: 555-565.

Sirinaovakul, B., and Thajchayapong, P., 1994, A knowledge base to assist a heuristic search approach to facility layout, *International Journal of Production Research*, 32: 141-160.

Sule, D. R., 1988, *Manufacturing Facilities: Location, Planning, and Design* (Boston, MA: PWS-KENT Publishing Co.).

Tam, K. Y., 1992, A simulated annealing algorithm for allocating space to manufacturing cells, *International Journal of Production Research*, 30 (1): 63-87.

Tam, K. Y., 1992, Genetic algorithms, function optimization and facility layout design, *European Journal of Operational Research*, 63: 322-346.

Tam, K. Y., and Shih, G. L., 1991, A hierarchical approach to the facility layout problem, *International Journal of Production Research*, 29 (1): 165-184.

Tate, D. M., and Smith, A. E., 1995, Unequal area facility layout using Genetic search, *IIE Transactions*, 27 (4): 465-472.

Tompkins, J. A., White, J. A., Bozer, Y. A., Frazelle E. H., Tanchoco, J. M. A., and Trevono, J., 1996, *Facilities Planning*, 2nd ed (New York: John Wiley & Sons).

van Camp, D. J., Carter, M. W., and Vannelli, A., 1991, A nonlinear optimization approach for solving facility layout problems, *European Journal of Operational Research*, 57: 174-189.

Wemmerlov, U., and Hyer, N. L., 1989, Survey of users of cellular manufacturing in U.S. industry, *International Journal of Production Research*, 27 (9): 1511-1530.

APPENDICES

APPENDIX A

MATHEMATICAL MODEL

APPENDIX A.1

The Binary Programming

```
SETS:


depti/1..3/:i;
deptj/1..10/:j;
xvalue(depti,deptj):x;
ENDSETS


min = @abs (a-1275*4/10)+@abs (b-1275*3/10)+@abs (c-1275*3/10);
@for (xvalue (i,j):@BIN (x(i,j)));


a-238*x (1,1)-112*x (1,2)-160*x (1,3)-80*x (1,4)-120*x (1,5)-80*x (1,6)
-60*x (1,7)-85*x (1,8)-221*x (1,9)-119*x (1,10)=0;
b-238*x (2,1)-112*x (2,2)-160*x (2,3)-80*x (2,4)-120*x (2,5)-80*x (2,6)
-60*x (2,7)-85*x (2,8)-221*x (2,9)-119*x (2,10)=0;
c-238*x (3,1)-112*x (3,2)-160*x (3,3)-80*x (3,4)-120*x (3,5)-80*x (3,6)
-60*x (3,7)-85*x (3,8)-221*x (3,9)-119*x (3,10)=0;


x (1,1)+x (2,1)+x (3,1)=1;
x (1,2)+x (2,2)+x (3,2)=1;
x (1,3)+x (2,3)+x (3,3)=1;
x (1,4)+x (2,4)+x (3,4)=1;
x (1,5)+x (2,5)+x (3,5)=1;
x (1,6)+x (2,6)+x (3,6)=1;
x (1,7)+x (2,7)+x (3,7)=1;
x (1,8)+x (2,8)+x (3,8)=1;
x (1,9)+x (2,9)+x (3,9)=1;
x (1,10)+x (2,10)+x (3,10)=1;


end
```

APPENDIX A.2

Result from the Example problem

```
Rows = 14 Vars = 46 No. Integer vars = 30
Nonlinear rows= 1 Nonlinear vars = 3
Nonlinear constraints= 0
Nonzero= 77 Constraint nonz=   63 Density=0.117
No. < :  0 No. =:  13 No. > :  0, Obj=MIN Single cols=  13
Optimal solution found at step:     59396
Objective value: 3.000000
Branch count: 1357


     Variable        Value        Reduced Cost
            A      511.0000        0.0000000
            B      383.0000        0.0000000
            C      381.0000        0.0000000
         I( 1)    0.0000000         0.0000000
         I( 2)    0.0000000         0.0000000
         I( 3)    0.0000000         0.0000000
         J( 1)    0.0000000         0.0000000
         J( 2)    0.0000000         0.0000000
         J( 3)    0.0000000         0.0000000
         J( 4)    0.0000000         0.0000000
         J( 5)    0.0000000         0.0000000
         J( 6)    0.0000000         0.0000000
         J( 7)    0.0000000         0.0000000
         J( 8)    0.0000000         0.0000000
         J( 9)    0.0000000         0.0000000
        J( 10)    0.0000000         0.0000000
      X( 1, 1)    0.0000000         0.0000000
      X( 1, 2)     1.000000         0.0000000
      X( 1, 3)     1.000000         0.0000000
      X( 1, 4)    0.0000000         0.0000000
      X( 1, 5)     1.000000         0.0000000
      X( 1, 6)    0.0000000         0.0000000
      X( 1, 7)    0.0000000         0.0000000
      X( 1, 8)    0.0000000         0.0000000
      X( 1, 9)    0.0000000         0.0000000
     X( 1, 10)     1.000000         0.0000000
      X( 2, 1)     1.000000         0.0000000
      X( 2, 2)    0.0000000         0.0000000
      X( 2, 3)    0.0000000         0.0000000
      X( 2, 4)    0.0000000         0.0000000
      X( 2, 5)    0.0000000         0.0000000
      X( 2, 6)    0.0000000         0.0000000
      X( 2, 7)     1.000000         0.0000000
      X( 2, 8)     1.000000         0.0000000
      X( 2, 9)    0.0000000         0.0000000
     X( 2, 10)    0.0000000         0.0000000
      X( 3, 1)    0.0000000         0.0000000
```

(Continued)

```
    X( 3, 2)     0.0000000       0.0000000
    X( 3, 3)     0.0000000       0.0000000
    X( 3, 4)      1.000000       0.0000000
    X( 3, 5)     0.0000000       0.0000000
    X( 3, 6)      1.000000       0.0000000
    X( 3, 7)     0.0000000       0.0000000
    X( 3, 8)     0.0000000       0.0000000
    X( 3, 9)      1.000000       0.0000000
   X( 3, 10)     0.0000000       0.0000000

     Row   Slack or Surplus    Dual Price
      1     3.000000         0.0000000
      2    0.0000000          0.0000000
      3    0.0000000          0.0000000
      4    0.0000000          0.0000000
      5    0.0000000          0.0000000
      6    0.0000000          0.0000000
      7    0.0000000          0.0000000
      8    0.0000000          0.0000000
      9    0.0000000          0.0000000
     10     0.0000000          0.0000000
     11     0.0000000          0.0000000
     12     0.0000000          0.0000000
     13     0.0000000          0.0000000
     14     0.0000000          0.0000000
```

APPENDIX A.3

The Average Linkage Values for each Department Pairs

Dept.	Dept.	Avg. Linkage
8	10	0.023
2	9	0.067
3	5	0.081
1	6	0.089
4	7	0.234
11	13	0.324
14	15	0.542
16	12	0.556
18	17	0.735

APPENDIX B

EXPERIMENTAL DATA

APPENDIX B.1

Experimental Data for Small Problem

Table B.1 The areas for small problem instances

Department *(i)*	Problem Instances (5-10 department problems)											
	1	2	3	4	5	6	7	8	9	10	11	12
1	20	40	80	60	30	80	80	70	20	60	80	30
2	70	80	30	40	80	30	30	50	20	40	30	40
3	50	80	60	70	30	60	60	30	80	30	60	70
4	80	40	50	50	30	50	50	60	30	30	50	60
5	50	80	80	60	80	80	80	70	40	30	80	50
6	-	-	70	50	70	70	70	20	60	40	70	50
7	-	-	-	-	20	50	50	60	30	70	50	70
8	-	-	-	-	-	-	20	40	50	50	20	20
9	-	-	-	-	-	-	-	-	20	70	70	60
10	-	-	-	-	-	-	-	-	-	-	50	20
Total area	270	221	370	330	340	420	440	400	350	420	560	470

The flow matrix for small size problems

Five departments

$$\begin{matrix} 0 & 4 & 8 & 5 & 2 \\ 0 & 0 & 7 & 8 & 0 \\ 0 & 0 & 0 & 7 & 4 \\ 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 1 & 4 & 8 & 0 \\ 0 & 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 6 & 3 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Six departments

$$\begin{matrix} 0 & 5 & 0 & 8 & 4 & 6 \\ 0 & 0 & 8 & 9 & 7 & 2 \\ 0 & 0 & 0 & 4 & 9 & 9 \\ 0 & 0 & 0 & 0 & 4 & 9 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 3 & 2 & 2 & 7 & 3 \\ 0 & 0 & 5 & 2 & 7 & 4 \\ 0 & 0 & 0 & 9 & 9 & 6 \\ 0 & 0 & 0 & 0 & 5 & 9 \\ 0 & 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Seven departments

0	0	9	2	3	7	3
0	0	5	1	10	6	4
0	0	0	5	3	4	2
0	0	0	0	6	8	5
0	0	0	0	0	6	2
0	0	0	0	0	0	4
0	0	0	0	0	0	0

0	0	8	4	6	8	9
0	0	7	2	4	9	9
0	0	0	4	9	1	4
0	0	0	0	8	0	1
0	0	0	0	0	2	2
0	0	0	0	0	0	6
0	0	0	0	0	0	0

Eight departments

0	8	4	6	8	9	7	2
0	0	4	9	9	4	9	1
0	0	0	4	8	0	1	2
0	0	0	0	2	6	3	2
0	0	0	0	0	0	7	4
0	0	0	0	0	0	9	5
0	0	0	0	0	0	0	4
0	0	0	0	0	0	0	0

0	8	5	7	4	3	2	2
0	0	7	3	5	2	7	4
0	0	0	9	9	6	5	9
0	0	0	0	8	6	8	7
0	0	0	0	0	3	3	3
0	0	0	0	0	0	5	7
0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0

Nine departments

0	10	6	4	5	3	4	2	6
0	0	8	5	6	2	4	8	7
0	0	0	5	6	8	1	6	1
0	0	0	0	4	3	9	0	8
0	0	0	0	0	10	10	8	4
0	0	0	0	0	0	5	2	6
0	0	0	0	0	0	0	3	10
0	0	0	0	0	0	0	0	7
0	0	0	0	0	0	0	0	0

0	5	5	5	4	9	0	3	0
0	0	7	7	10	6	4	2	6
0	0	0	7	4	0	4	8	8
0	0	0	0	9	8	4	6	7
0	0	0	0	0	2	9	6	6
0	0	0	0	0	0	2	5	9
0	0	0	0	0	0	0	3	7
0	0	0	0	0	0	0	0	4
0	0	0	0	0	0	0	0	0

Ten departments

0	6	8	9	7	2	4	9	9	4
0	0	9	1	4	8	0	1	2	2
0	0	0	6	3	2	0	7	4	9
0	0	0	0	5	4	8	5	2	7
0	0	0	0	0	8	0	7	4	8
0	0	0	0	0	0	5	7	4	3
0	0	0	0	0	0	0	2	2	7
0	0	0	0	0	0	0	0	3	5
0	0	0	0	0	0	0	0	0	2
0	0	0	0	0	0	0	0	0	0

0	4	3	9	0	8	10	10	8	4
0	0	5	2	6	3	10	7	4	7
0	0	0	3	4	9	7	2	8	6
0	0	0	0	1	2	6	6	4	6
0	0	0	0	0	5	0	0	3	0
0	0	0	0	0	0	4	7	1	0
0	0	0	0	0	0	0	6	6	0
0	0	0	0	0	0	0	0	0	2
0	0	0	0	0	0	0	0	0	6
0	0	0	0	0	0	0	0	0	0

APPENDIX B.2

Experimental Data for Medium Problem

Table B.2 The areas for medium problem instances

Department *(i)*	Problem Instances (11-20 department problems)						
	1	2	3	4	5	6	7
1	80	70	80	30	40	60	80
2	30	60	30	60	40	80	30
3	60	50	60	70	60	70	30
4	50	70	50	50	70	70	80
5	80	20	80	60	40	50	30
6	70	50	70	30	40	60	60
7	50	70	50	40	60	70	80
8	20	80	20	70	50	20	60
9	70	50	70	60	70	80	80
10	50	60	50	50	50	80	20
11	60	30	60	50	70	60	20
12	-	80	70	70	50	30	70
13	-	-	80	20	30	80	50
14	-	-	-	60	60	50	80
15	-	-	-	-	80	70	70
16	-	-	-	-	70	40	60
17	-	-	-	-	-	80	40
18	-	-	-	-	-	-	30
19	-	-	-	-	-	-	30
20	-	-	-	-	-	-	30
Total area	620	690	770	720	880	1050	1030

The flow matrix for medium size problems

Eleven departments

0	8	9	7	2	4	9	9	4	9	1
0	0	4	8	0	1	2	2	6	3	2
0	0	0	0	7	4	9	5	4	8	5
0	0	0	0	2	7	8	0	7	4	8
0	0	0	0	0	5	7	4	3	2	2
0	0	0	0	0	0	7	3	5	2	7
0	0	0	0	0	0	0	4	9	9	6
0	0	0	0	0	0	0	0	5	9	8
0	0	0	0	0	0	0	0	0	6	8
0	0	0	0	0	0	0	0	0	0	7
0	0	0	0	0	0	0	0	0	0	0

Twelve departments

0	9	7	2	4	9	9	4	9	1	4	8
0	0	0	1	2	2	6	3	2	0	7	4
0	0	0	9	5	4	8	5	2	7	8	0
0	0	0	0	7	4	8	5	7	4	3	2
0	0	0	0	0	2	7	3	5	2	7	4
0	0	0	0	0	0	9	9	6	5	9	8
0	0	0	0	0	0	0	6	8	7	3	3
0	0	0	0	0	0	0	0	3	5	7	3
0	0	0	0	0	0	0	0	0	8	6	4
0	0	0	0	0	0	0	0	0	0	7	5
0	0	0	0	0	0	0	0	0	0	0	4
0	0	0	0	0	0	0	0	0	0	0	0

Thirteen departments

0	7	2	4	9	9	4	9	1	4	8	0	1
0	0	2	2	6	3	2	0	7	4	9	5	4
0	0	0	8	5	2	7	8	0	7	4	8	5
0	0	0	0	7	4	3	2	2	7	3	5	2
0	0	0	0	0	7	4	9	9	6	5	9	8
0	0	0	0	0	0	6	8	7	3	3	3	5
0	0	0	0	0	0	0	7	3	8	6	4	7
0	0	0	0	0	0	0	0	5	4	7	6	8
0	0	0	0	0	0	0	0	0	10	5	9	2
0	0	0	0	0	0	0	0	0	0	10	3	3
0	0	0	0	0	0	0	0	0	0	0	9	7
0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0

Fourteen departments

0	1	4	3	9	0	8	10	10	8	4	5	2	6
0	0	3	10	7	4	7	3	4	9	7	2	8	6
0	0	0	1	2	6	6	4	6	5	0	0	3	0
0	0	0	0	4	7	1	0	6	6	0	0	2	6
0	0	0	0	0	1	4	6	7	7	1	5	4	4
0	0	0	0	0	0	2	7	7	7	5	6	1	5
0	0	0	0	0	0	0	7	9	3	3	9	2	8
0	0	0	0	0	0	0	0	9	2	2	0	1	6
0	0	0	0	0	0	0	0	0	2	8	2	2	10
0	0	0	0	0	0	0	0	0	0	4	3	3	4
0	0	0	0	0	0	0	0	0	0	0	4	6	1
0	0	0	0	0	0	0	0	0	0	0	0	0	5
0	0	0	0	0	0	0	0	0	0	0	0	0	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sixteen departments

0	7	5	1	7	4	1	6	8	7	10	10	1	4	5	3
0	0	6	0	6	7	10	8	7	4	6	8	1	9	9	6
0	0	0	3	9	5	7	4	10	1	6	3	9	3	7	1
0	0	0	0	4	5	4	6	6	1	9	8	8	8	7	2
0	0	0	0	0	3	6	5	1	1	3	4	5	9	6	3
0	0	0	0	0	0	5	6	2	8	10	6	0	8	6	7
0	0	0	0	0	0	0	1	4	4	2	8	8	5	10	1
0	0	0	0	0	0	0	0	9	8	4	6	10	6	2	4
0	0	0	0	0	0	0	0	0	2	5	4	5	6	1	3
0	0	0	0	0	0	0	0	0	0	6	2	6	5	5	5
0	0	0	0	0	0	0	0	0	0	0	5	5	9	3	4
0	0	0	0	0	0	0	0	0	0	0	0	3	4	3	4
0	0	0	0	0	0	0	0	0	0	0	0	0	5	7	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Seventeen departments

0	1	6	3	9	3	7	1	4	5	4	6	6	1	9	8	8
0	0	8	7	2	3	6	5	1	1	3	4	5	9	6	3	5
0	0	0	6	2	8	10	6	0	8	6	7	1	4	4	2	8
0	0	0	0	8	5	10	1	9	8	4	6	10	6	2	4	2
0	0	0	0	0	5	4	5	6	1	3	6	2	6	2	6	5
0	0	0	0	0	0	5	5	9	3	4	3	4	3	4	5	7
0	0	0	0	0	0	0	3	1	4	5	0	7	7	3	3	7
0	0	0	0	0	0	0	0	8	10	5	9	5	8	8	2	4
0	0	0	0	0	0	0	0	0	5	7	6	5	4	1	4	4
0	0	0	0	0	0	0	0	0	0	3	9	8	9	6	0	6
0	0	0	0	0	0	0	0	0	0	0	8	10	2	7	5	9
0	0	0	0	0	0	0	0	0	0	0	0	7	2	4	2	10
0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	8	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Twenty departments

0	4	9	5	8	5	5	5	4	9	0	3	0	7	7	10	6	4	2	6
0	0	7	4	0	4	8	8	9	8	4	6	7	4	6	6	7	5	8	5
0	0	0	3	7	4	6	7	4	4	7	8	4	4	6	6	7	5	8	5
0	0	0	0	2	7	10	8	7	5	1	7	4	1	6	8	7	10	10	1
0	0	0	0	0	4	5	3	6	0	6	7	10	8	7	4	6	8	1	9
0	0	0	0	0	0	9	6	3	9	5	7	4	10	1	6	3	9	3	7
0	0	0	0	0	0	0	1	4	5	4	6	6	1	9	8	8	8	7	2
0	0	0	0	0	0	0	0	3	6	5	1	1	3	4	5	9	6	3	5
0	0	0	0	0	0	0	0	0	6	2	8	10	6	0	8	6	7	1	4
0	0	0	0	0	0	0	0	0	0	4	2	8	8	5	10	1	9	8	4
0	0	0	0	0	0	0	0	0	0	0	6	10	6	2	4	2	5	4	5
0	0	0	0	0	0	0	0	0	0	0	0	6	1	3	6	2	6	2	6
0	0	0	0	0	0	0	0	0	0	0	0	0	5	5	5	9	3	4	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	4	5	7	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	5	0	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	3	3	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	10	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APPENDIX B.3

Experimental Data for Large Problem

Table B.3 The areas for large problem instances

Department (*i*)	Problem Instances (21-26 department problems)				
	1	2	3	4	5
1	80	70	40	40	70
2	30	70	40	80	80
3	60	80	60	80	30
4	50	70	50	30	70
5	80	40	70	80	60
6	70	60	50	60	50
7	50	70	70	20	60
8	20	30	50	60	20
9	70	80	30	40	60
10	50	50	60	30	70
11	60	60	80	30	80
12	70	30	70	20	20
13	80	50	60	20	60
14	70	80	50	70	40
15	30	40	20	20	60
16	40	60	60	80	50
17	80	40	40	40	60
18	80	60	20	70	40
19	40	60	50	80	30
20	80	40	70	30	30
21	20	40	60	30	50
22		60	80	40	70
23			80	20	50
24				40	70
25					20
26					40
Total area	1210	1240	1260	1110	1340

The flow matrix for small size problems

Twenty-one departments

0	4	8	0	1	2	2	6	3	2	0	7	4	9	5	4	8	5	2	7	8
0	0	0	7	4	8	5	7	4	3	2	2	7	3	5	2	7	4	9	9	6
0	0	0	5	9	8	6	8	7	3	3	3	5	7	3	8	6	4	7	5	4
0	0	0	0	7	6	8	10	5	9	2	10	3	3	9	7	1	0	9	2	3
0	0	0	0	0	7	3	5	1	10	6	4	5	3	4	2	6	8	5	6	2
0	0	0	0	0	0	4	8	7	5	6	8	1	6	1	4	3	9	0	8	10
0	0	0	0	0	0	0	10	8	4	5	2	6	3	10	7	4	7	3	4	9
0	0	0	0	0	0	0	0	7	2	8	6	1	2	6	6	4	6	5	0	0
0	0	0	0	0	0	0	0	0	3	0	4	7	1	0	6	6	0	0	2	6
0	0	0	0	0	0	0	0	0	0	1	4	6	7	7	1	5	4	4	2	7
0	0	0	0	0	0	0	0	0	0	0	7	7	5	6	1	5	7	9	3	3
0	0	0	0	0	0	0	0	0	0	0	0	9	2	8	9	2	2	0	1	6
0	0	0	0	0	0	0	0	0	0	0	0	0	2	8	2	2	10	4	3	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	6	1	0	5	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	3	2	9	2	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	7	9	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	4	9	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	3	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Twenty-two departments

0	8	4	4	6	6	7	5	8	5	2	7	10	8	7	5	1	7	4	1	6	8
0	0	7	10	10	1	4	5	3	6	0	6	7	10	8	7	4	6	8	1	9	9
0	0	0	6	3	9	5	7	4	10	1	6	3	9	3	7	1	4	5	4	6	6
0	0	0	0	1	9	8	8	8	7	2	3	6	5	1	1	3	4	5	9	6	3
0	0	0	0	0	5	6	2	8	10	6	0	8	6	7	1	4	4	2	8	8	5
0	0	0	0	0	0	10	1	9	8	4	6	10	6	2	4	2	5	4	5	6	1
0	0	0	0	0	0	0	3	6	2	6	2	6	5	5	5	9	3	4	3	4	3
0	0	0	0	0	0	0	0	4	5	7	3	1	4	5	0	7	7	3	3	7	8
0	0	0	0	0	0	0	0	0	10	5	9	5	8	8	2	4	5	7	6	5	4
0	0	0	0	0	0	0	0	0	0	1	4	4	3	9	8	9	6	0	6	8	10
0	0	0	0	0	0	0	0	0	0	0	2	7	5	9	7	2	4	2	10	4	0
0	0	0	0	0	0	0	0	0	0	0	0	8	9	3	3	9	1	9	7	8	2
0	0	0	0	0	0	0	0	0	0	0	0	0	5	1	9	6	3	4	9	4	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	6	1	5	1	8	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	6	9	3	9	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	9	1	3	10
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	5	3	7	10
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	7	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Twenty-three departments

0	1	4	5	3	6	0	6	7	10	8	7	4	3	8	1	9	9	6	3	9	5	7
0	0	4	10	1	6	3	9	3	7	1	4	5	4	6	6	1	9	8	8	8	7	2
0	0	0	3	6	5	1	1	3	4	5	9	6	3	5	6	2	8	10	6	0	8	6
0	0	0	0	7	1	4	4	2	8	8	5	10	1	9	8	4	6	10	6	2	4	2
0	0	0	0	0	5	4	5	6	1	3	6	2	6	2	6	5	5	5	9	3	4	3
0	0	0	0	0	0	4	3	4	5	7	3	1	4	5	0	7	7	3	3	7	8	10
0	0	0	0	0	0	0	5	9	5	8	8	2	4	5	7	6	5	4	1	4	4	3
0	0	0	0	0	0	0	0	9	8	9	6	0	6	8	10	2	7	5	9	7	2	4
0	0	0	0	0	0	0	0	0	2	10	4	0	8	9	3	3	9	1	9	7	8	2
0	0	0	0	0	0	0	0	0	0	5	1	9	6	3	4	9	4	1	2	0	6	1
0	0	0	0	0	0	0	0	0	0	0	5	1	8	4	8	0	6	9	3	9	1	1
0	0	0	0	0	0	0	0	0	0	0	0	2	9	1	3	10	2	5	3	7	10	8
0	0	0	0	0	0	0	0	0	0	0	0	0	7	1	1	0	3	8	10	0	8	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	2	8	3	8	7	2	6	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	2	6	2	5	6	7	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	3	8	3	9	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	2	5	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	6	5	1	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	8	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	10	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Twenty-four departments

0	0	8	10	1	6	1	3	1	2	4	1	2	9	4	5	1	0	2	8	9	2	10	7
0	0	9	5	5	3	1	3	2	9	1	0	4	7	9	6	2	3	2	0	4	6	1	8
0	0	0	10	7	8	2	2	3	7	10	6	5	8	8	7	7	8	10	5	8	6	3	7
0	0	0	0	4	9	5	1	7	5	2	7	1	10	2	2	6	0	3	3	0	1	4	8
0	0	0	0	0	9	0	7	8	7	10	7	7	3	9	3	1	7	10	2	4	1	4	9
0	0	0	0	0	0	4	3	4	7	7	9	8	5	6	1	5	5	9	9	5	8	7	0
0	0	0	0	0	0	0	1	5	0	2	3	9	3	3	4	8	2	5	5	3	4	9	8
0	0	0	0	0	0	0	0	9	5	8	1	1	4	4	5	10	3	3	6	5	2	5	5
0	0	0	0	0	0	0	0	0	8	8	9	6	6	6	1	2	4	6	3	8	4	0	6
0	0	0	0	0	0	0	0	0	0	6	2	1	9	3	2	0	6	5	1	6	4	1	4
0	0	0	0	0	0	0	0	0	0	0	2	9	6	9	6	8	4	10	1	5	5	3	4
0	0	0	0	0	0	0	0	0	0	0	0	1	5	8	1	2	1	8	1	9	5	10	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	4	4	4	9	7	10	2	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	5	0	6	4	8	8	4	3	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	5	4	8	4	9	2	4	9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	1	7	0	6	10	4	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	10	2	6	10	3	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	4	5	8	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	7	9	5	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	9	4	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	3	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Twenty-six departments

0	3	1	10	1	4	5	7	1	1	5	8	5	3	1	4	3	10	9	10	9	0	1	8	5	6
0	0	1	6	1	8	0	3	9	5	1	9	1	8	10	1	8	4	10	1	0	10	3	9	2	7
0	0	0	7	8	1	5	7	7	6	9	0	1	7	4	2	5	5	7	3	2	1	0	10	0	0
0	0	0	0	8	3	6	8	6	4	5	7	3	0	9	4	6	5	7	4	7	9	6	1	0	7
0	0	0	0	0	3	4	0	3	7	5	4	1	8	1	5	8	6	0	9	10	8	5	1	2	2
0	0	0	0	0	0	5	10	3	4	0	2	3	1	3	6	2	5	8	5	9	3	5	5	7	0
0	0	0	0	0	0	0	5	8	9	3	10	5	6	1	5	0	4	0	4	7	10	10	1	3	7
0	0	0	0	0	0	0	0	2	8	3	0	6	8	10	7	4	2	9	5	3	2	7	9	8	9
0	0	0	0	0	0	0	0	0	3	9	9	6	2	10	4	9	1	9	1	7	2	0	8	8	7
0	0	0	0	0	0	0	0	0	0	3	2	0	6	5	6	6	4	1	0	10	0	9	10	5	6
0	0	0	0	0	0	0	0	0	0	0	0	1	8	9	8	5	5	7	8	7	4	3	6	0	2
0	0	0	0	0	0	0	0	0	0	0	0	2	1	7	1	5	9	1	2	6	5	2	3	2	1
0	0	0	0	0	0	0	0	0	0	0	0	0	9	7	6	2	6	4	2	9	8	1	5	0	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	8	6	9	5	5	6	7	5	0	9	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	9	2	3	10	6	4	1	3	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	8	4	10	6	8	2	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	1	8	10	7	2	6	8	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	6	1	6	6	3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	1	8	2	6	5
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	7	3	3	9	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	9	4	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	9	1	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APPENDIX C

NORMAL PROBABILITY PLOTS

Normal Probability Plot

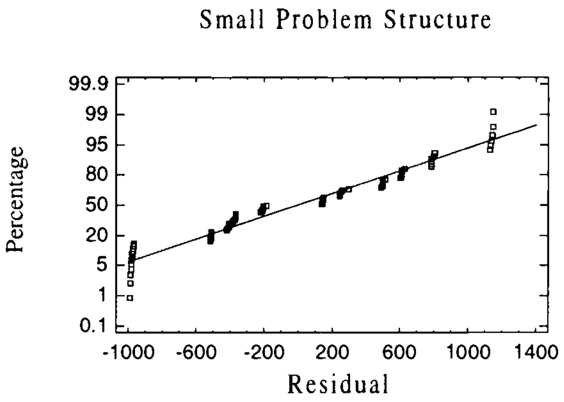


Figure C.1 Normal probability plot for small problem

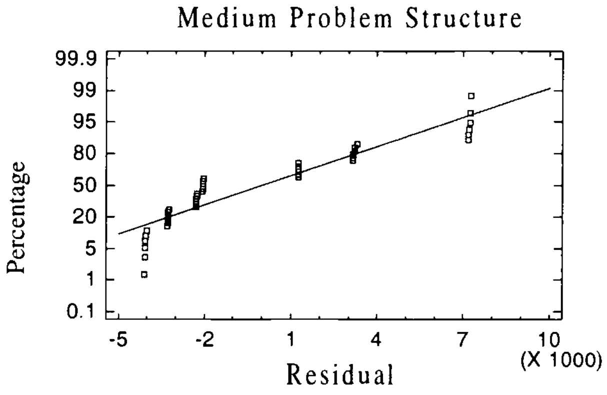


Figure C.2 Normal probability plot for medium problem

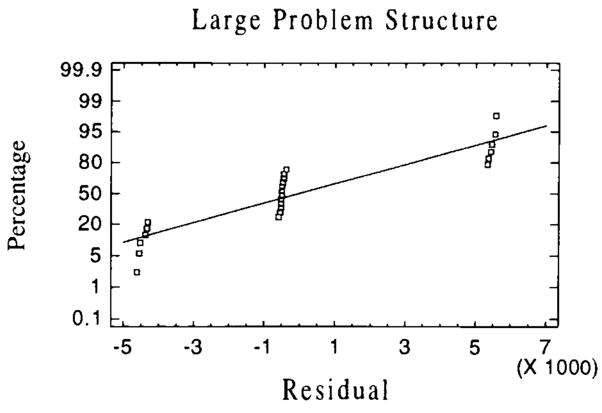


Figure C.3 Normal probability plot for large problem

APPENDIX D

EXPERIMENTAL RESULTS

Table D.1 Results Obtained for Small Problem

Problem	TS1		TS2		TS3		TS4		TS5		TS6	
Instance	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
1	235.3	0.5	235.3	0.7	235.3	0.8	235.3	0.3	235.3	1.1	235.3	1.3
2	220.8	0.7	240.7	0.7	220.8	1	220.8	0.6	220.8	1.6	220.8	2.3
3	800.2	0.9	800.2	1.1	819.8	1.4	800.2	2.3	800.2	3	800.2	2.4
4	697.6	1.2	706.0	1.3	697.6	1.3	697.6	3.5	697.6	3.5	697.6	6.1
5	837.8	2.5	837.8	5.2	841.4	4.3	838.1	3.1	841.4	5.8	841.4	6.1
6	999.3	7.6	999.3	11.8	999.3	10.6	999.3	5.1	1027.3	8.7	999.3	6.9
7	1351.7	12.4	1366.8	20.5	1351.7	19.2	1351.7	10.2	1354.3	18.1	1354.3	12.1
8	1450.9	13.7	1516.8	21	1460.4	18.2	1450.9	12.6	1471.3	13.9	1450.9	13.2
9	1704.7	20.6	1704.7	28.6	1704.7	30.5	1704.7	23.3	1726.2	29.3	1704.7	31.8
10	1814.6	31.4	1847.6	38.6	1814.6	40.2	1814.6	35.2	1814.6	48.6	1814.6	49.6
11	2347.6	85.2	2344.6	101.5	2344.6	107.8	2351.3	87.4	2344.6	125.1	2351.3	131.1
12	1990.6	42.3	2018.2	59.7	1990.6	57.3	1990.6	44.7	2018.2	71.5	1990.6	71.1

Table D.2 Results Obtained for Medium Problem

Problem	TS1		TS2		TS3		TS4		TS5		TS6	
Instance	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
1	3241.5	152.5	3301.9	187.1	3241.5	201.4	3241.5	169.4	3301.9	190.5	3241.5	178.6
2	4051.0	212.4	4058.1	510.2	4044.8	503.4	4051.0	245.4	4044.8	512.1	4051.0	523.4
3	5284.7	346.1	5312.7	567.2	5275.8	595.1	5261.4	342.1	5255.4	582.7	5275.8	599.1
4	5033.6	255.3	5049.3	543.0	5053.9	560.7	5031.6	292.1	5056.7	536.7	5053.9	540.5
5	8569.5	724.4	8588.6	1020.4	8609.4	1079.5	8569.5	799.1	8569.5	1120.4	8582.9	1198.1
6	10534.0	841.2	10502.0	1404.4	10660.0	1494.2	10517.0	842.1	10460.0	1412.4	10460.0	1471.3
7	14575.0	909.7	14543.0	1602.3	14575.0	1711.1	14459.0	2159.3	14472.0	2112.6	14560.0	2601.3

Table D.3 Results Obtained for Large Problem

Problem	TS1		TS2		TS3		TS4		TS5		TS6	
Instance	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
1	15965	2374	16039	3859	15675	3777	15632	6128	15606	10109	15831	8376
2	19736	1989	19872	3691	19736	2960	19699	7866	19687	12139	19736	12265
3	20615	2021	20639	2251	20600	3105	20597	8041	20620	13212	20457	13651
4	19697	1961	19860	3479	19697	3379	19697	6668	19835	9914	19697	6677
5	25726	3812	25723	4124	25786	4875	25587	10689	25754	17769	25538	16037

Note: Time (seconds)

APPENDIX E

ANALYSIS OF VARIANCE FOR RANDOMIZED BLOCK DESIGN EXPERIMENT

Table E.1 Results of Wilcoxon signed-rank test on total cost

Comparisons	Significant Difference at Alfa = 0.05		
	Small Problem	Medium Problem	Large Problem
TS1 vs TS2	Yes	-	Yes
TS1 vs TS3	-	-	-
TS1 vs TS4	-	No	No
TS1 vs TS5	Yes	-	-
TS1 vs TS6	Yes	-	No
TS2 vs TS3	No	-	-
TS2 vs TS4	No	No	No
TS2 vs TS5	-	No	No
TS2 vs TS6	No	-	No
TS3 vs TS4	-	No	No
TS3 vs TS5	-	-	-
TS3 vs TS6	-	-	-
TS4 vs TS5	Yes	-	-
TS4 vs TS6	-	-	-
TS5 vs TS6	-	-	-

APPENDIX F

PSEUDO CODE FOR
TABU SEARCH-BASED HEURISTIC ALGORITHM

MAIN PROGRAM-OUTSIDE SEARCH

```
Generate the intial bay assignment
Generate the initial department identification
Determine the tabu search parameters
Evaluate the total cost for initial department location configuration by Call subroutine (INSIDE SEARCH)
Admit the initial Outside Candidate List (OCL) and Outside Index List (OIL)
Initialize the outside tabu-list (out_TL)
Set the initial bay location (outside) configuration with department location (inside) configuration as the current parent node
        DO
        {
                Evaluate the bay locations seeds configuration
                Evaluate the total cost for each bay location seed configuration by Call subroutine (INSIDE SEARCH)
                Use the evaluated total cost to sort the seeds of bay location configuration
                For each seed generated from the current parent node
                {
                        the best outside solution $\leftarrow$ large number
                        IF (seed $\in$ OCL), skip it
                        IF (out_move status $\neq$ tabu) or (out_move status = tabu, but out_AL criteria is satisfied)
                                IF (seed < the best outside solution)
                        {
                                out_tabu list $\leftarrow$ location of bay that was moved to the next adjacent position
                                OCL $\leftarrow$ current move
                                the best outside solution $\leftarrow$ current seed
                                update out_AL

                        }
                }

                IF (there is an improvement in total cost)
                {
                        OWI = 0
                        Update best outside solution
                }
                ELSE
                {
                        OWI = OWI +1
                }
        } WHILE (OWI has not exceeded specified numbers)

Terminate the Outside search

Return the best outside solution (bay location configuration) together with its best inside solution (department location layout) as the best solution found so far.
```

SUBROUTINE-INSIDE SEARCH

```
Start with the initial department location identification passed by outside search
Determine the parameters of tabu search used for the Inside search
DO
{
        Initialized the Inside tabu-list (in_TL)
        Initialized the Inside Candidate List (ICL) and the Inside Index List (IIL)
        Initialized the Inside long-term memory (IN_LTM frequency matrix)
        // all heuristics exept TS1 and TS4 //

        DO
        {
                Generate the neighborhood solutions by applying swap moves to the current seed
                For each neighborhood solution generated from the current seed
                {
                        Evaluate the total cost
                        the best inside solution ← large number
                        IF  (seed ∈ ICL), skip it
                        IF (in_move status ≠ tabu) or (in_move status = tabu, but in_AL criteria is
                        satisfied)
                                IF (seed < the best outside solution)
                        {
                                in_TL ← current move
                                ICL ← current seed
                                the best inside solution ← current seed
                                update in_AL

                        }
                }
                Update IIL
                Update IN_LTM frequency matrix   // all heuristics exept TS1 and TS4 //
                IF (the next seed < current seed)
                {
                        IWI = 0
                        Update the best inside solution
                ELSE
                        IWI = IWI +1
                IF (current seed = local optima)
                {
                        IIL ← current seed
                        Entries into Inside Index List (IIL) is increased by 1
                }
                Updated IN_LTM matrix
                current seed  ← the next seed

        } WHILE (both IWI and IIL have not exceeded specified numbers)
        Indentify the new restart by using the LTM matrix
        Next intial solution ← new restart solution
} WHILE (the number of restart has not reached the specified number)

Terminate the Inside search
Return the best inside solution to the outside search
```