A Quadratic Partial Assignment and Packing Model and Algorithm for the Airline Gate Assignment Problem

by

Eric L. Brown

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Industrial and Systems Engineering

APPROVED:

[illegible]

Dr. Hanif D. Sherali, Chairman

Subhash C Sarin

Dr. Subhash C. Sarin

[illegible]

Dr. Antonio A. Trani

March, 1995

Blacksburg, Virginia

c. 2

LD
5655
V855
1995
B769
C.2

A Quadratic Partial Assignment and Packing Model and Algorithm for the Airline Gate Assignment Problem

by

Eric L. Brown

Dr. Hanif D. Sherali, Chairman

Industrial and Systems Engineering

(ABSTRACT)

This thesis is concerned with an Airline Gate Assignment problem that seeks to allocate gates to aircraft at an airport, using the objective of minimizing passenger walking distances. The problem is modelled as a variant of the quadratic assignment problem with set packing constraints. The quadratic objective function is then transformed into an equivalent linearized form by applying the first-order linearization technique of Sherali and Adams [1989, 1990]. In addition to linearizing the problem, the application of this technique generates additional constraints that provide a tighter linear programming representation. A suitable solution process that exploits the structure of the linearized problem is developed. Test results are presented using realistic data obtained from *USAir*.

Acknowledgements

I would like to acknowledge many people who supported me while completing this thesis. First and foremost I would like to thank my advisor, Dr. Hanif D. Sherali, who has been very patient with me every step of the way. I cannot put a price on the knowledge that I have gained from him in the classroom and in completing this thesis. Without his support, this thesis would not have been completed.

Second, I would like to thank Dr. Subhash C. Sarin and Dr. Antonio A. Trani for serving on the thesis committee and for their helpful suggestions. A special thanks to Dr. Sarin for teaching me linear programming as an undergraduate and encouraging me to obtain an advanced degree in a subject that I enjoy so much.

Next, there have been many friends that have helped and encouraged me along the way. These include Hunter Nichols, Patrick Driscoll, Price Smith, Dr. Jeffrey Tew, Dr. and Mrs. Charles Nunnally, Mr. and Mrs. Steve Gilmore, and Eric Brown, an Electrical Engineering undergraduate.

Most importantly, my wife Natalie deserves the most thanks. She made many sacrifices for me over the past two years by working and providing the finances to put me through graduate school. Thank you dear!

Finally, I thank God for his many blessings upon my family and me. Particularly for God's faithfulness, that whatever the challenges in life are, He will see me through them.

Contents

List of Tables

List of Figures

Chapter I

Introduction

The problem of assigning flights to airline gates has been a challenging problem for operations research analysts for the past twenty years. Today, new formulations and algorithms are being investigated to enhance the problem solving capability. As the airlines have changed to a "hub" oriented environment, the number of passengers transferring within the terminal from one flight to the next has grown. With the recent airline mergers, buyouts, and bankruptcies, airlines have increased their flight routes and the number of gates at their hubs.

Once the flight schedule has been developed and a fleet has been assigned, the gate assignment problem can be formulated and solved, assuming that no terminal layout changes are required and that at least one gate is available upon arrival for every flight. The information provided to the model includes the types of aircraft, the time requirements on the ground (i.e. arrival and departure times), and the estimated passenger volumes. The output provides a gate assignment for each flight.

A relevant question to ask is how do passengers from flight a transfer to flight b when flight b arrives after flight a departs (assuming that there are no transfers from flight b to flight a)? Another question is how do we handle international flights, neighboring restrictions, and terminal overcrowding? At the same time, are there existing models to solve the gate assignment for many time periods without running the model more than once to achieve the same purpose?

There have been many papers published presenting ideas on solving gate assignment problems. These papers seek to minimize passenger walking distance, which ideally favors the passenger's interest. Braaksma [1977] proposes to minimize passenger total walking distances by taking a weighted average of passengers transferring from one flight to the next. Babić, Teodorović, and Tosić [1984], Mangoubi and Mathaisel [1985], and Bihr [1990] mathematically show how to minimize passenger walking distance using the generalized assignment problem (GAP). Bihr's [1990] model, however, must run several times during the day as his model takes only one arrival-departure cycle into account. Mangoubi and Mathaisel [1985] propose a second formulation using the quadratic assignment problem (QAP). Constraints for all models include assigning one aircraft to a gate at a time, requiring each aircraft to arrive and depart from the same gate, and restricting a 747 jumbo jet from a gate only large enough for at least a 727.

In another paper, Schröder [1972] proposed to minimize the expected number of passengers having to use the hardstand gates, which are only accessible by bus, versus the pier gates. Hence, this would require the use of fewer transport buses. Here, the three-index assignment model is proposed. This model assigns flights to gates within a horizon equal to one day in duration. (Each day is subdivided into several time periods.) In addition to the previous constraints, another restriction is included to handle domestic and international flights (i.e., customs procedures). Vanderstraeten and Bergeron [1988] proposed to minimize the off-gate handling area, requiring fewer transport buses as well. Their model assigns flights to gates over multiple time periods as well. Certain neighboring constraints are proposed and included in the model to accommodate any large size aircraft requiring the use of more than one parking position.

The objective of *USAir's* gate assignment system is to balance the passenger load within the terminal by utilizing each gate equally. Their model assigns flights to gates for the entire day (all time periods). When a flight departs from a gate, *USAir* maximizes the time between the departure and the next flight arriving at the gate. Once a gate assignment schedule is accepted, it is adopted for a three to four month period. From the passenger's viewpoint, the frequent flyers, by habit, can walk to the same gate every time. From *USAir's* point of view, the baggage handlers and ticket attendants also know where the flights are located on a day by day basis.

USAir implements the following constraints. First, there is a time delay allowance to permit a plane to be pushed out and clear of the ramp area. This time delay can be increased to allow for early arrivals and late departures. In regard to international flights, certain airports require the airplane to be pushed away from customs into another gate slot designated for the particular airline. Second, gate restrictions are included to assign arriving or departing flights to preassigned gates. Some ramp areas can't handle large size airplanes parked next to each other, since the wings can touch. Two gates share the same push-out tug; therefore departure times must not be equal. Flights having the same destination are kept apart from each other to avoid passengers (sometimes mistakenly) jumping from one flight to the other. Other restrictions prohibit flights with similar arrival and departure times from being assigned to gates close to each other in order to avoid terminal overcrowding.

This thesis focuses on extending Mangoubi and Mathaisel's [1985] model formulations in the light of *USAir's* practice in studying the airline gate assignment problem. The quadratic assignment problem is modified via set packing constraints to formulate a problem that seeks to minimize total passenger walking distances. Appropriate constraints are developed, based on the literature and the practice at *USAir*. A suitable solution methodology is then developed, and this is tested using realistic data obtained from *USAir*.

We begin by presenting in Chapter 2, a literature review pertinent to this problem. Chapter 3 contains a statement of the problem along with the QAP mathematical formulation and the proposed methodology. Chapter 4 presents an example problem applied to the methodology developed in Chapter 3. Chapter 5 presents test results.

Chapter II

Literature Review

This chapter presents literature related to the present research. The first section of this chapter reviews literature on gate assignments to minimize passenger walking distances. The second section reviews literature on the gate assignment problem in which the objective is to maximize the number of aircraft assignments at one terminal versus another. The third section reviews literature on the quadratic assignment problem (QAP).

The intent of this literature review is to present the different applications and formulations of the gate assignment problem, mainly utilizing the generalized assignment problem. The section on the quadratic assignment problem will present linearization techniques, solution methodologies such as branch-and-bound, heuristics, graph theoretical techniques, simulated annealing, and tabu search methods. Examining the existing literature, a suitable quadratic assignment with side-constraints model is developed for the problem of determining gate assignments to minimize passenger walking distances.

2.1 Gate Assignment to Minimize Passenger Walking Distance

Braaksma [1977]

This was the first article to propose that an interaction exists between the gate assignment (operation) and walking distance (layout) of any terminal building that can be exploited to improve the performance of a terminal. This interaction can be measured by a *passenger's weighted walking distance*, where the number of passengers on a particular flight is multiplied by the passenger walking distance. The idea demonstrates that walking distances can be reduced by a judicious assignment of flights to gate positions. This can have a profound impact on passenger walking distances without requiring changes in the terminal layout or the flight schedule. Only arriving and departing passengers were considered since connecting passenger's flow data was not readily available. The results indicated that walking distances were reduced significantly. However, as the airline at the terminal under study, namely, Air Canada at Toronoto's International Airport Terminal 2, was in the process of changing aircraft types, some of them being larger in size than before, walking distances increased slightly.

Babić, Teodorović, and Tosić [1984]

A method to find aircraft gate assignments that minimize the average passenger walking distances is proposed. In this study, it is assumed that

1. All aircraft arriving can use any available gate
2. Arrival and departure times are flexible to allow an arriving flight to find at least one available gate, and that
3. Each aircraft arrives and departs from the same gate.

To formulate this problem, the following is defined:

- $Y = y_1, y_2, \ldots, y_j, \ldots, y_n \equiv$ set of aircraft parking positions (gates)
- $B = b_1, b_2, \ldots, b_i, \ldots, b_m \equiv$ set of aircraft that use gates in set Y
- $T^{(a)} = t_1^{(a)}, t_2^{(a)}, \ldots, t_i^{(a)}, \ldots, t_m^{(a)} \equiv$ set of arrival times of set B, where $t_i^{(a)} \leq t_{i+1}^{(a)}$ for $i = 1, 2, \ldots, m - 1$
- $T^{(d)} = t_1^{(d)}, t_2^{(d)}, \ldots, t_i^{(d)}, \ldots, t_m^{(d)} \equiv$ set of departure times of set B where $t_i^{(d)} \leq t_{i+1}^{(d)}$ for $i = 1, 2, \ldots, m - 1$
- $D_a = d_1^a, d_2^a, \ldots, d_j^a, \ldots, d_n^a \equiv$ set of walking distances for arriving passengers from each gate $j = 1, \ldots, n$ to the baggage claim area
- $D_d = d_1^d, d_2^d, \ldots, d_j^d, \ldots, d_n^d \equiv$ set of walking distances for departing passengers from the check- in counter area to each gate $j = 1, \ldots, n$
- $P^{(a)} = p_1^{(a)}, p_2^{(a)}, \ldots, p_i^{(a)}, \ldots, p_m^{(a)} \equiv$ set of the number of arriving passengers where $p_i^{(a)} \geq 0$ for $i = 1, 2, \ldots, m$

- $P^{(d)} = p_1^{(d)}, p_2^{(d)}, \ldots, p_i^{(d)}, \ldots, p_m^{(d)} \equiv$ set of the number of departing passengers where $p_i^{(d)} \geq 0$ for $i = 1, 2, \ldots, m$
- $x_{ij} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \\ 0 & \text{otherwise} \end{cases}$

The objective is to minimize the total weighted walking distance of all arriving and departing passengers by assigning an available gate j to each aircraft i at the moment of arrival $t_i^{(a)}$. Each aircraft must be assigned to some gate and no two aircraft may be assigned to the same gate concurrently. The formal problem is posed as follows:

$$\begin{aligned}
\text{Minimize} \quad & Z = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}(p_i^{(a)}d_j^a + p_i^{(d)}d_j^d) && (2.1)\\
\text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1 \quad \forall\ i = 1, \ldots, m && (2.2)\\
& t_k^{(d)} \leq t_l^{(a)} \quad \text{when } x_{kj} = 1 \text{ and } x_{lj} = 1, && \\
& \qquad\qquad \text{for } k < l \text{ and for } j = 1, 2, \ldots, n && (2.3)\\
& \mathbf{x} \text{ binary.}
\end{aligned}$$

This model runs within the horizon of one day. It handles the maximum allowable number of gates available to assign to flights. To find an optimal solution, a standard branch-and-bound technique is used, slightly modified to accelerate computation. Note that this problem takes arriving and departing passengers into account, not transferring passengers. The optimal solution found was seen to mainly affect the number of passengers who walk maximum distances, decreasing this number, and those who walk minimum distances, increasing this number.

Mangoubi and Mathaisel [1985]

In planning aircraft gate utilization, one consideration is passenger walking distance to reach the departure gate, baggage claim area, or a connecting flight. This study takes Babić et al.'s [1984] formulation and modifies it to allow for transferring passengers. As in Braaksma [1977], Toronto International Airport Terminal 2 is the terminal under study. Minimizing passenger walking distances within the airport terminal area is solved using (1) a linear programming (LP) relaxation of an integer program formulation and (2) a heuristic that provides a good solution to the congestion and walking distance problems. The heuristic is also used to provide an initial solution for the same branch-and-bound technique of Babić et al. [1984].

To formulate this problem, the following are defined:

- Z $\equiv$ the total walking distance for all passengers
- m $\equiv$ the number of flights in the schedule
- n $\equiv$ the number of gates
- p_i^a, p_i^d, p_i^t $\equiv$ respectively, the estimated number of arriving, departing, and transferring passengers using flight i
- d_j^a $\equiv$ the walking distance of an arriving passenger at gate j to the baggage claim area
- d_j^d $\equiv$ the walking distance of a departing passenger from the check-in counter to gate j

- $d^t_j \equiv$ the walking distance for transferring passengers. If w_{jl} is the distance between gate j and gate l then this is taken as the average walking distance for a transfer passenger arriving at gate j, i.e., $d^t_j = \sum_{l=1}^{n} \underline{w_{jl}}/n \ \forall \ j = 1, \ldots, n$

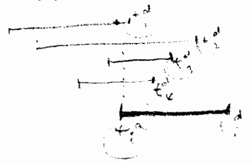
- $t^a_i \equiv$ is the arrival time of flight i

- $t^d_h \equiv$ is the departure time of flight h

- $L(i) \equiv$ the set of all flights h which landed before flight i and are still assigned to their respected gates at the time flight i arrives

- $x_{ij} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \\ 0 & \text{otherwise} \end{cases}$

If the flights are indexed in order of their arrival time, then each set $L(i)$, here termed a "conflict set," can be defined as follows:



$$L(i) = \{h \in 1, \ldots, i-1 \mid t^d_h \geq t^a_i\}.$$

To simplify the formulation, when considering flight i, one only needs to consider those flights belonging to $L(i-1)$ as well as flight $i-1$ itself. Hence, $L(i)$ can be represented as:

$$L(i) = \{h \in \{L(i-1) \ \cup \ (i-1)\} \mid t^d_h \geq t^a_i\}.$$

If any flight conflicts in time with flight i, it cannot be assigned to the same gate j.

The objective is to minimize the total weighted passenger walking distances of all arriving, departing, and transferring passengers by assigning an available gate j to each aircraft i at the moment of arrival. Each aircraft must be assigned to some gate and no two aircraft may be assigned to the same gate concurrently. The formal assignment problem is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n}(p_i^a d_j^a + p_i^d d_j^d + p_i^t d_j^t)x_{ij} \tag{2.4}$$

$$\text{subject to} \quad \sum_{j=1}^{n} x_{ij} = 1 \quad \forall\ i = 1,\ldots,m \tag{2.5}$$

$$\sum_{h\in L(i)} x_{hj} + x_{ij} \le 1 \quad \forall\ i = 1,\ldots,m\ \forall\ j = 1,\ldots,n \tag{2.6}$$

$$\mathbf{x} \text{ binary.}$$

In identifying redundant constraints, we use an example of three aircraft ($p-3$, $p-2$, and $p-1$) on the ground at the time a pth aircraft is scheduled for arrival. The conflict set is $L(p) = \{p-3, p-2, p-1\}$ and the conflict constraint for the pth flight and any gate j is:

$$\sum_{h\in L(p)} x_{hj} + x_{pj} = x_{p-3,j} + x_{p-2,j} + x_{p-1,j} + x_{pj} \le 1. \tag{2.7}$$

Assume that the $(p+1)$st flight arrives, while p nor any of the three flights contained in $L(p)$ depart. The conflict set is $L(p+1) = \{p-3, p-2, p-1, p\} = \{L(p) \cup p\}$ and the conflict constraint for each gate j is:

$$\sum_{h\in L(p+1)} x_{hj} + x_{p+1,j} = x_{p-3,j} + x_{p-2,j} + x_{p-1,j} + x_{pj} + x_{p+1,j} \le 1. \tag{2.8}$$

Any solution satisfying (2.8) (even in the continuous sense) will automatically satisfy (2.7). Therefore, the constraints generated by the pth flight are redundant and can be dropped.

We mention here that three more constraints are proposed in addition to constraints (2.5) and (2.6). First, they suggested running separate models for each airline carrier, requiring the gates to be divided into sets. Today at large airports, however, carriers assign their own aircraft to gates, each requiring their own models.

Second, one may want to exclude some aircraft types from some gates. To do this, if we let

- $B \equiv$ the subset consisting of all flights with a wide-body aircraft,
- $G \equiv$ the subset of all gates which are incompatible with such aircraft,

then the constraint can be accommodated by setting the appropriate decision variables to zero (i.e. $x_{ij} = 0 \ \forall \ i \in B$ and $j \in G$).

Third, if two flights b and c serve the same large number of transfers, it may be desirable to have them assigned near to each other. If we let

- $w_{zs} \equiv$ the distance between gates z and s,
- $D_{bc}^{max} \equiv$ the maximum allowable distance between flights b and c,

then the assignment can be done while enforcing

$$\sum_{z=1}^{n}\sum_{s=1}^{n} x_{bz} w_{zs} x_{cs} \le D_{bc}^{max}. \tag{2.9}$$

Since this constraint is nonlinear, a second method is proposed. First, the transfer walking distance contribution is dropped from the objective function for x_{bs} and x_{cs} for all gates $s = 1, \ldots, n$; and the problem is solved. If flight b is assigned to a gate, say z, then x_{bz} is fixed at 1 and the following constraint is added:

$$\sum_{s=1}^{n} w_{zs} x_{cs} \le D_{bc}^{max}. \tag{2.10}$$

Another approach for handling the problem of transfer passengers was formulated using the QAP.[1] Keeping all variables as before, the following are defined:

- $p^t_{ik} \equiv$ the expected number of passengers transferring between flights i and k
- $Q \equiv$ a very large constant
- $y_{ijkl} = x_{ij}x_{kl}, \quad i \neq k$ and $j \neq l$
- $y_{ijkl} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \text{ and flight } k \text{ is assigned to gate } l \\ 0 & \text{otherwise.} \end{cases}$

Then the formal problem as posed in the aforementioned paper, is as follows:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}(p^a_i d^a_j + p^d_i d^d_j)x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n}(p^t_{ik}w_{jl})y_{ijkl} \tag{2.11}$$

$$\text{subject to} \quad \sum_{j=1}^{n} x_{ij} = 1 \quad \forall\ i = 1,\ldots,m \tag{2.12}$$

$$\sum_{h\in L(i)} x_{hj} + x_{ij} \leq 1 \quad \forall\ i = 1,\ldots,m\ \forall\ j = 1,\ldots,n \tag{2.13}$$

$$\sum_{k=1}^{m}\sum_{l=1}^{n} y_{ijkl} < Qx_{ij} \quad \forall\ i,j \tag{2.14}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ijkl} < Qx_{kl} \quad \forall\ k,l \tag{2.15}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{n} y_{ijkl} = 1 \quad \forall\ i,k \tag{2.16}$$

$$\mathbf{x}, \mathbf{y} \text{ binary.}$$

Constraints (2.11) and (2.12) are exactly the same as constraints (2.5) and (2.6), respectively. Constraint (2.14) states that if flight i is not assigned to gate j, then the sum over all possible assignments of flights k to gate l in y_{ijkl} must be zero, else

[1] Mangoubi and Mathaisel [1985] thank one of the referees who suggested this formulation; we present their formulation and suggest a correction of it. Although admittedly hard to solve, it may well be an avenue for further research.

the sum is effectively unconstrained. Constraint (2.15) is similar to (2.14), but with respect to x_{kl}. Constraint (2.16) says that for all flight pairs i and k, exactly one y_{ijkl} must equal 1 over all possible combinations of gates j and l.

However, since y_{ijkl} denotes the product of $x_{ij}x_{kl}$, we must have by (2.12) that

$$\sum_{k \neq i} \sum_{l \neq j} y_{ijkl} \; = \; x_{ij} \sum_{k \neq i} \sum_{l \neq j} x_{kl} \; \leq \; (m-1)x_{ij} \quad \forall \; i,j. \tag{2.17}$$

Replacing (2.14), and likewise, rewriting (2.15), we obtain the modified formulation of this problem as follows:

$$\begin{aligned}
\text{Minimize} \quad & \sum_{i=1}^{m}\sum_{j=1}^{n}(p_i^a d_j^a + p_i^d d_j^d)x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n}(p_{ik}^t w_{jl})y_{ijkl} && (2.18)\\
\text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1 \quad \forall \; i = 1,\ldots,m && (2.19)\\
& \sum_{h \in L(i)} x_{hj} + x_{ij} \leq 1 \quad \forall \; i = 1,\ldots,m \; \forall \; j = 1,\ldots,n && (2.20)\\
& \sum_{k \neq i}\sum_{l \neq j} y_{ijkl} \leq (m-1)x_{ij} \quad \forall \; i,j && (2.21)\\
& \sum_{i \neq k}\sum_{j \neq l} y_{ijkl} \leq (m-1)x_{kl} \quad \forall \; k,l && (2.22)\\
& \sum_{j=1}^{n}\sum_{\substack{l=1 \\ l \neq j}}^{n} y_{ijkl} \leq 1 \quad \forall \; i,k && (2.23)\\
& y_{ijkl} \geq x_{ij} + x_{kl} - 1 \quad \forall \; i \neq k, \; j \neq l && (2.24)\\
& \mathbf{x}, \mathbf{y} \text{ binary.}
\end{aligned}$$

Bihr [1990]

A sensible assignment of flights to gates is proposed which takes the passenger's perspective into account. This study looks at the problem of transferring passengers from gate to gate, minimizing the total walking distance, during one arrival-departure cycle. The more the number of passengers on flight X that neded to transfer for destination A, the closer flight X should be positioned to the flight departing for A. To formulate this problem, the following are defined:

- $l \equiv \#$ of arrival gates
- $n \equiv \#$ of departure gates
- $m \equiv \#$ of arrival flights, $m \leq l$
- $PAX(i,j) \equiv \#$ of passengers arriving on flight i departing from gate j, $i = 1,\ldots,m$ and $j = 1,\ldots,n$
- $DIST(i,j) \equiv$ the distance from gate i to gate j, $i = 1,\ldots,l$ and $j = 1,\ldots,n$
- $c_{ij} = \sum_k PAX(i,n) * DIST(j,n) \equiv$ passenger-distance weights from arrival to departure gates[2]
- $x_{ij} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \\ 0 & \text{otherwise.} \end{cases}$

The objective is to minimize the total passenger-distance by assigning an available gate i to each aircraft j upon arrival during one arrival- departure cycle. Only one aircraft can be assigned to each gate and each arriving flight is assigned to some gate. The formal problem is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \qquad (2.25)$$

[2]The author expresses c_{ij} as the elements of the matrix product of $PAX(i,j) * DIST(i,j)^T$, assuming T represents the transpose, but appears to be confusing scaler and matrix quantities. We have stated a representation of his intent.

$$\text{subject to} \quad \sum_{j=1}^{k} x_{ij} = 1 \quad \forall\ i = 1, \ldots, m \tag{2.26}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall\ j = 1, \ldots, n \tag{2.27}$$

$$\mathbf{x} \text{ binary.}$$

Note that if a particular flight is to be preassigned to a particular gate, then the corresponding x_{ij} variable can be fixed at one, thereby removing the corresponding flight and gate from further consideration. This would include any flight and aircraft which must be constrained to particular gates for reasons of servicing, size, and baggage and cargo handling.

It is suggested that the best application of this model is for the determination of a standard gate assignment at a very large hub. However, this model must be run several times each day. After running the model, the assignments would be reviewed, imposing further necessary constraints, and re-running the program. Minor adjustments can be made due to, for example, flight delays. However, frequent disruptions to the assignment could be counter-productive not only for the passengers, but for the ground crews.

2.2 Other Gate Assignment Problems

Schröder [1972]

Two types of gate positions exist: piers where passengers board the airplane directly from the lounge, and hardstands where passengers are moved to or from the aircraft by bus. A model was developed to minimize the expected number of passengers to be transported by bus. Operational conditions include:

1. If ground time permits, aircraft arriving and then departing may be separated by an intermediate push-out and pull-in, thus assigning the aircraft to different gates.
2. Certain gates are reserved for domestic or international flights.
3. For 747's or similar size aircraft:
   - hardstand gates will not handle 747's,
   - some pier gates are too small for 747's,
   - the availability of some adjacent gates are affected by the parking of 747's.
4. Certain flights are handled at the same gate every day.

A direct application, versus a formal model using the three-index assignment problem are explored. To formulate the general model the following are defined:

- $m \equiv$ total number of flights ($i = 1, \ldots, m$)
- $n \equiv$ total number of gates ($j = 1, \ldots, n$)
- $T \equiv$ set of flight i's airport occupancy time units ($t = 1, \ldots, T$)
- $a_{ijt} \equiv$ expected number of passengers on flight i assigned to gate j upon arrival time t
- $D_{ij} \equiv$ length of time flight i occupies gate j

- $d \equiv$ length of time for one period
- $x_{ijt} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \text{ upon arrival time } t \\ 0 & \text{otherwise.} \end{cases}$

The general model is stated as follows. (See below for errors in this formulation):

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{T} a_{ijt}x_{ijt} \tag{2.28}$$

$$\text{subject to} \quad \sum_{j=1}^{n} x_{ijt} = 1 \quad \forall\ i = 1,\ldots,m \text{ and } t = 1,\ldots,T \tag{2.29}$$

$$\sum_{i=1}^{m} x_{ijt} = 1 \quad \forall\ j = 1,\ldots,n \text{ and } t = 1,\ldots,T \tag{2.30}$$

$$\sum_{t=1}^{T} x_{ijt} = 1 \quad \forall\ i = 1,\ldots,m \text{ and } j = 1,\ldots,n \tag{2.31}$$

$$\text{x binary.}$$

In order to maximize the number of passengers on the pier gates, the problem would have to be one of maximization, or a_{ijt} would have to be replaced by $-a_{ijt}$ above. To have a flight remain at the assigned gate for all time periods occupied by it, Schröder suggests that constraint (2.31) should be changed to

$$\sum_{t=1}^{T} x_{ijt} = \frac{D_{ij}}{d} \quad \forall\ i = 1,\ldots,m \text{ and } j = 1,\ldots,n. \tag{2.32}$$

However, problems exist with the current formulation. Equation (2.30) states that all gates will have a flight assigned to it during all time periods, which is not necessarily true. Equation (2.31) states that each flight is assigned to each gate sometime during its entire airport occupancy time.

However, Schröder [1972] states that, "Each plane defined by one index being held constant must contain a solution to the corresponding two-index assignment problem." In formulating his intent, the following are redefined:

- $G_i \equiv$ set of all gates feasible for assigning flight i
- $a_{ij} \equiv$ expected number of passengers handled on pier gates for flight i assigned to gate $j \in G_i$
- $x_{ij} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \\ 0 & \text{otherwise.} \end{cases}$

It is assumed that there are no towing operations involved.

The objective is to find an assignment of each of the m aircraft to the n pier gate positions, if possible, upon their respective arrival times. To minimize the number of arriving passengers using hardstand gates, the model maximizes the number of passengers using a pier gate. If a pier gate is not available, then the flight is directed to a hardstand gate. Each aircraft must be assigned to some gate, pier or hardstand, and no two aircraft can be concurrently assigned to the same gate. Borrowing Mangoubi et al.'s [1985] Equation (2.6) (keeping the definitions of this equation the same), the revised gate assignment problem is stated as follows:

$$\text{Maximize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}x_{ij} \tag{2.33}$$

$$\text{subject to} \quad \sum_{j \in G_i} x_{ij} \le 1 \quad \forall\ i = 1,\ldots,m \tag{2.34}$$

$$\sum_{h \in L(i)} x_{hj} + x_{ij} \le 1 \quad \forall\ i = 1,\ldots,m \text{ and } j \in G_i \tag{2.35}$$

x binary.

The program consists of three stages. The first stage defines the time scale *d*; in this study *d* is taken as 15 minutes. Arrival times are rounded down to the nearest quarter hour, whereas departure times are rounded up. The duration of events was also set at this stage, allowing the extra time needed for aircraft push-outs and push-ins. The second stage lists the events in time order, with priority being given to the shorter event in case of equal valued start times. The third stage assigns events to gates according to the constructed priority scheme. A feasible solution always exists. Branch-and-bound methods have been proposed for solving this problem with considerable success.

It is noted that this model generates a near-optimal assignment of aircraft to gates. It was developed in a short amount of time to accommodate Lufthansa at the new Frankfurt passenger terminal. Because of its simple and fast operation, different alternatives could be run. Extra buffer time on arrival and departure times could be added and gates could be reserved for international or domestic flights.

Vanderstraeten and Bergeron [1988]

This study seeks to minimize the number of aircraft having to use the off-gate handling area at Air Canada's terminal at the Toronto International Airport. The term "event" is introduced to take into account the various types of visits each aircraft may make to the airport. Specifications relative to the gates are:

1. Gates are divided among services (domestic, international, overseas, ...),
2. "Size codes" exist for gates that cannot handle large size airplanes,

3. "Neighboring" restrictions exist for gates when some airplanes, while parked at these gates, prevent the use of one or both adjacent gates, and

4. A time gap is needed for various activities after the departure of an aircraft before the next new arrival.

To formulate the problem, the following are defined:

- $I \equiv$ set of scheduled events to be assigned to the gates
- $G_i \equiv$ set of all gates accommodating the event associated with aircraft i
- $A_{ij} \equiv$ set of all events present at the airport at the same time as event i and which might also be assigned to gate j
- $HA_i \equiv$ arrival time for event i when the aircraft arrives at the gate
- $HD_i \equiv$ departure time for event i when the aircraft leaves the gate
- B_{ijr}, $B_{ijl} \equiv$ set of, respectively, all event-gate assignment combinations at the airport which may restrict aircraft i's assignment to gate j because of a previous assignment at the right or left-handed side of gate j
- $x_{ij} = \begin{cases} 1 & \text{if aircraft } i \text{ is assigned to gate } j \\ 0 & \text{otherwise.} \end{cases}$

Associated with each event $i \in I$ is HA_i, HD_i, and a aircraft size code. An event is considered assigned if there is a gate $j \in J_i$ for which $x_{ij} = 1$, otherwise it will be considered as an off-gate assignment, directed away from the main terminal. The formulation of these events allows the model to run continuously over a period of one day.

The four types of events that exist in this study are:

Short stay: Events where the duration of the aircraft stay is too short to allow any towing operations beyond boarding and deplaning. HA_i and HD_i will be equivalent to the arrival and departure times of the aircraft at the airport. In the case that no gate exists that is common to the two services that the aircraft will perform, the aircraft will automatically have to park at an off-gate area.

Long stay: Events where towing operations are allowed. Two events, i and j, are created. The first event has an arrival time of HA_i and a departure time HD_i equivalent to HA_i, plus the time required for deplaning. The second event will have an arrival time of HD_j minus the time required for boarding and a departure time HD_j.

No local traffic: Events where the aircraft arrives and either boards or deplanes passengers, but never both. An example would be an international flight (i.e. Paris-Montreal-Toronto) where local passenger traffic may not be offered in Montreal. This event is usually equivalent to the short stay event.

Overnight: Events where the aircraft arrives at the end of the day and leaves the following day. Two events, i and j are created. HA_i is the arrival time and HD_i is equivalent to HA_i plus the end of operations time required on an aircraft. The next day, the departure time is taken as HD_j and the arrival time is taken as HD_j minus the boarding time and the "start of activities" time.

The objective is to minimize the number of off-gate events, similar to maximizing the number of events assigned to gates. Each event must be assigned to at most one gate, no two events may occupy the same gate, allowing the earlier event to fulfill its occupation time. Also some aircraft types have neighboring gate restrictions. The formal problem is posed as follows:

$$\text{Maximize} \quad Z = \sum_{i \in I} \sum_{j \in G_i} x_{ij} \tag{2.36}$$

$$\text{subject to} \quad \sum_{j \in G_i} x_{ij} \leq 1 \quad \forall \; i \in I \tag{2.37}$$

$$x_{ij} + \sum_{k \in A_{ij}} x_{kj} \leq 1 \quad \forall \; i \in I, \; j \in G_i \tag{2.38}$$

$$x_{ij} + \sum_{(k,p) \in B_{ijr}} x_{kp} + \sum_{(k,q) \in B_{ijl}} x_{kq} \leq 1 \quad \forall \; i \in I, \; j \in G_i \tag{2.39}$$

$$\mathbf{x} \text{ binary.}$$

2.3 Quadratic Assignment Problem

The quadratic assignment problem (QAP) is a celebrated, notorious class of problems. The problem of assigning interacting, indivisible entities to mutually exclusive locations has long been of interest to engineers, economists, and management scientists. However, problems having greater than fifteen entities are and remain largely unsolvable using the best available algorithms. As stated in Francis, McGinnis, and White [1992], "One of the great ironies of discrete optimization is that QAP is so simple to state but so difficult to solve." Because of the vast literature on the QAP, this section is divided into the following subsections: alternative reformulation techniques, branch-and-bound, heuristics, graph theoretical techniques, simulated annealing, tabu search, genetic algorithms, and survey papers on the QAP.

Before starting, let us define the following:

- $p_{ik} \equiv$ flow from facility i to facility k
- $d_{jl} \equiv$ distance from location j to location l
- $a_{ij} \equiv$ fixed cost of assigning facility i to location j
- $b_{ijkl} \equiv$ interactive cost of assigning facility i to location j when facility k is assigned location l
- $x_{ij} = \begin{cases} 1 & \text{if facility } i \text{ is assigned to location } j \\ 0 & \text{otherwise} \end{cases}$
- $y_{ijkl} = x_{ij}x_{kl} \quad i < k$ and $j \neq l$. Equivalently,
- $y_{ijkl} = \begin{cases} 1 & \text{if facilities } i \text{ and } k \text{ are assigned to locations } k \text{ and } l \text{ respectively} \\ 0 & \text{otherwise} \end{cases}$
- $x \in X_A \equiv \begin{cases} \sum_{i=1}^{m} x_{ij} = 1 & \text{for} \quad j = 1, \ldots, m \\ \sum_{j=1}^{m} x_{ij} = 1 & \text{for} \quad i = 1, \ldots, m \\ \text{x binary.} \end{cases}$

2.3.1 Alternative (Re-)Formulations

Koopmans and Beckmann [1957]

This is the first published statement of the QAP. The objective is to locate plants to sites based on the economic activity between each pair of facilities, minimizing the total interaction cost between the facilities. A one-to-one assignment between plants (facilities) and sites (locations) is required. This problem can be posed as follows:

$$\underset{x \in X_A}{\text{Minimize}} \quad \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik} d_{jl} x_{ij} x_{kl}. \tag{2.40}$$

Lawler [1963]

Two generalizations of the QAP are considered. The first method proposes to incorporate a fixed cost a_{ij} of assigning facility i to location j, leading to the following formulation:

$$\underset{x \in X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij} x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik} d_{jl} x_{ij} x_{kl}. \tag{2.41}$$

The second method discusses a multicommodity case of a flow of products $h = 1, \ldots, H$. Accordingly, p_{ik}^{h} is the flow of each product h from facility i to facility k and d_{jl}^{h} is the distance related cost per unit flow of product h from location j to location l. This leads to the problem formulation

$$\underset{x \in X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij} x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m}\sum_{h=1}^{H} p_{ik}^{h} d_{jl}^{h} x_{ij} x_{kl}. \tag{2.42}$$

Lawler's QAP formulations can be linearized by defining m^4 y_{ijkl} variables, as introduced earlier. Considering (2.42) for example, the equivalent linearized integer program can be posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} s_{ijkl}y_{ijkl} \tag{2.43}$$

$$\text{subject to} \quad \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} y_{ijkl} = m^2 \tag{2.44}$$

$$x_{ij} + x_{kl} - 2y_{ijkl} \geq 0 \quad i,j,k,l = 1,\ldots,m \tag{2.45}$$

$$x \in X_A, \quad \text{y binary}$$

$$\text{where} \quad s_{ijkl} = \begin{cases} p_{ik}d_{jl} & \text{if} \quad i \neq k \quad \text{or} \quad j \neq l \\ a_{ij} + p_{ii}d_{jj} & \text{if} \quad i = k \quad \text{and} \quad j = l. \end{cases}$$

<u>Graves and Whinston [1970]</u>

This study proposed to incorporate into the objective function a cost component b_{ijkl} that would depend on a pair of assignments. Thus, extending (2.42), for example, the problem becomes the following:

$$\underset{x \in X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m}\sum_{h=1}^{H} b^h_{ijkl}x_{ij}x_{kl}$$
$$+\sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m}\sum_{h=1}^{H} p^h_{ik}d^h_{jl}x_{ij}x_{kl}. \tag{2.46}$$

<u>Pierce and Crowston [1971]</u>

In this paper, Lawler's [1963] form of the objective function (2.42) was considered while incorporating Graves and Whinston's [1970] component b_{ijkl}. The quadratic assignment problem is then posed as follows:

$$\underset{x \in X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m}\sum_{h=1}^{H} s_{ijkl}x_{ij}x_{kl} \tag{2.47}$$

$$\text{where} \quad s_{ijkl} = \begin{cases} \sum_{h=1}^{H} b_{ijkl}^{h} + \sum_{h=1}^{H} p_{ik}^{h} d_{jl}^{h} & \text{if} \quad i \neq k \quad \text{or} \quad j \neq l \\ a_{ij} + p_{ii}^{h} d_{jj}^{h} & \text{if} \quad i = k \quad \text{and} \quad j = l \end{cases}$$

Three classes of branch-and-bound algorithms, single assignment, pair assignment, and pair exclusion were then presented in great detail. The different algorithms can give rise to the elaboration of different partial trees of solutions with differing numbers of nodes. Since the time required to evaluate a single node in a tree can differ among the algorithms, it is difficult to measure the relative efficiencies of the different algorithms. In practice, these relative efficiencies may well turn out to be highly dependent on the particular form of the QAP being solved.

Beale and Tomlin [1972]

This study sought to relocate departments of the British Civil Service, situated in London, to other areas in Britain. The objective is to monetarily reduce the cost of office accommodation and to socially provide employment in development areas. The unique features in this study are:

1. Facilities m are greater in number than sites n, meaning that there might exist a multiple assignment of facilities to each location.

2. Total employment at any site must fall between an upper and a lower bound if any facility is to be located at this site.

Keeping all variables as before, the following are defined:

- $E_i \equiv$ number of persons employed by facility i

- $F_i \equiv$ number of facilities communicating with facility $i \equiv$ number of non-zero p_{ik}, $k = 1, \ldots, m$
- U_j, $L_j \equiv$ respectively, upper and lower bounds on the employment at site j, in case any facility is located at the site
- $s_j \equiv$ nonnegative slack on the employment upper bound at site j
- $y_{ijkl} = \begin{cases} 1 & \text{if } i < k,\ a_{ij} \neq 0, \text{ and } x_{ij}x_{kl} = 1 \\ 0 & \text{otherwise} \end{cases}$
- $\xi_j = \begin{cases} 1 & \text{if any facility is located at } j \\ 0 & \text{otherwise.} \end{cases}$

Mathematically, the problem has $mn + 2n + \frac{m(m-1)}{2} + m$ constraints, with the last $\frac{1}{2}m(m-1)$ being generalized upper bounding types of restrictions, and has $nm + 2n + \frac{mn(m-1)(n-1)}{2}$ variables. The formal assignment problem is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}x_{ij} + \sum_{i=1}^{m-1}\sum_{j=1}^{n}\sum_{k=i+1}^{m}\sum_{\substack{l=1 \\ l \neq j}}^{m} p_{ik}d_{jl}y_{ijkl} \tag{2.48}$$

$$\text{subject to} \quad \sum_{k=i+1}^{m}\sum_{l=1}^{n} y_{ijkl} + \sum_{k=1}^{m-1}\sum_{l=1}^{n} y_{klij} - F_i x_{ij} = 0 \quad i = 1, \ldots, m \quad j = 1, \ldots, n \tag{2.49}$$

$$\sum_{i=1}^{m} E_i x_{ij} - U_j\xi_j + s_j = 0 \quad j = 1, \ldots, n \tag{2.50}$$

$$(L_j - U_j)\xi_j + s_j \leq 0 \quad j = 1, \ldots, n \tag{2.51}$$

$$\sum_{j=1}^{n}\sum_{l=1}^{n} y_{ijkl} = 1 \quad 1 \leq i < k \leq m \tag{2.52}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, \ldots, m \tag{2.53}$$

$\mathbf{x}$, $\mathbf{y}$ binary.

Sharpe and Brotchie [1972]

This is a modification to the QAP model of Pierce et al. [1971], considering T time periods ($t = 1, \ldots, T$). Each variable is defined as before, except with the t subscript signifying the time period. The formal problem is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{t=1}^{T} a_{ijt}x_{ijt} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m}\sum_{h=1}^{H}\sum_{t=1}^{T} p_{ikt}^{h} d_{jlt}^{h} x_{ijt}x_{klt} \tag{2.54}$$

$$\text{subject to} \quad \sum_{i=1}^{m} x_{ijt} = 1 \quad \forall\ j = 1, \ldots, m \text{ and } t = 1, \ldots, T \tag{2.55}$$

$$\sum_{j=1}^{m} x_{ijt} = 1 \quad \forall\ i = 1, \ldots, m \text{ and } t = 1, \ldots, T \tag{2.56}$$

x binary.

Foulds and Robinson [1976]

A graph theoretic formulation of a special case of the QAP is presented. For each pair of facilities, a benefit from locating them adjacent to each other is known, and the objective is to maximize the sum of benefits over all pairs of adjacent facilities. Defining,

- $G = (V, E) \equiv$ a weighted graph with V as a nonempty set of vertices (facilities), and E as a set of edges
- $w_{ik} \equiv$ closeness rating indicating the desirability of locating facility i adjacent to facility k
- $N \equiv$ set of pairs of facilities which must be adjacent in any feasible solution
- $F \equiv$ set of pairs of facilities which must not be adjacent in any feasible solution
- $E' = \{(i, k) : x_{ik} = 1, (i, k) \in E\}$

and letting other notation be defined as before, the graph theoretic formulation is given as follows:

$$\text{Maximize} \quad Z = \sum_{(i,k)} \sum_{\in E} w_{ik} x_{ik} \tag{2.57}$$

$$\text{subject to} \quad x_{ik} = 1 \quad (i,k) \in N \tag{2.58}$$

$$x_{ik} = 0 \quad (i,k) \in F \tag{2.59}$$

$$(V, N \subseteq E') \text{ is a planar graph}$$

$$\text{x binary.}$$

Love and Wong [1976]

An integer programming formulation of the QAP is presented using rectangular distances. It is assumed that the m locations are given as points on a two-dimensional plane and transportation costs are proportional to weighted rectangular distances. Keeping all variables as defined before, we define the following:

- $w_{ik} \equiv$ nonnegative weight or flow between facilities i and k
- $R_{ik} = \begin{cases} \text{the horizontal distance between facilities } i \text{ and } k \text{ if } i \text{ is to the right of } k \\ 0 \text{ otherwise} \end{cases}$
- $L_{ij} = \begin{cases} \text{the horizontal distance between facilities } i \text{ and } k \text{ if } i \text{ is to the left of } k \\ 0 \text{ otherwise} \end{cases}$
- $A_{ij} = \begin{cases} \text{the vertical distance between facilities } i \text{ and } k \text{ if } i \text{ is located above } k \\ 0 \text{ otherwise} \end{cases}$

- $B_{ij} = \begin{cases} \text{the vertical distance between facilities } i \text{ and } k \text{ if } i \text{ is located below } k \\ 0 \text{ otherwise} \end{cases}$

- $(\alpha_i, \beta_i) \equiv$ location of facility i, for $i = 1, \ldots, m$

- $s_j \equiv$ sum of coordinates of location j, for $j = 1, \ldots, m$

- $d_j \equiv$ difference of coordinates of location j, value of the first coordinate minus value of the second coordinate

The formulation is in a two dimensional space, having m^2 variables and $(m^2 + 3m)$ constraints. The problem is posed as follows:

$$\underset{x \in X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} w_{ik}(R_{ik} + L_{ik} + A_{ik} + B_{ik}) \tag{2.60}$$

$$\text{subject to} \quad R_{ik} - L_{ik} = \alpha_i - \alpha_k \quad i = 1, \ldots, m-1, \; j = i+1 \tag{2.61}$$

$$A_{ik} - B_{ik} = \beta_i - \beta_k \quad i = 1, \ldots, m-1, \quad j = i+1 \tag{2.62}$$

$$\alpha_i + \beta_i = \sum_{j=1}^{m} s_j x_{ij} \quad i = 1, \ldots, m \tag{2.63}$$

$$\alpha_i - \beta_i = \sum_{j=1}^{m} d_j x_{ij} \quad i = 1, \ldots, m. \tag{2.64}$$

Kaufman and Broeckx [1978]

Considering Koopmans and Beckmann's QAP (2.40) with symmetric flow and distance matrices, the objective function may be rewritten as:

$$Z = \sum_{i=1}^{m} \sum_{j=1}^{m} x_{ij} \left(\sum_{k=1}^{m} \sum_{l=1}^{m} p_{ik} d_{jl} x_{kl} \right).$$

Introducing m^2 continuous variables as

$$w_{ij} = x_{ij} \sum_{k=1}^{m} \sum_{l=1}^{m} p_{ik} d_{jl} x_{kl}$$

and defining

$$c_{ij}^{+} \;=\; \max \; \{\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik}d_{jl}, 0\} \;\text{ for }\; i,j = 1,\ldots,m,$$

we can equivalently minimize the sum of the w_{ij}'s. The problem, having $m^2 + 2m$ constraints, is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} w_{ij} \tag{2.65}$$

$$\text{subject to} \quad c_{ij}^{+}x_{ij} + \sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik}d_{jl}x_{kl} - w_{ij} \le c_{ij}^{+} \tag{2.66}$$

$$w_{ij} \ge 0 \quad \forall \quad i,j = 1,\ldots,m$$

$$x \in X_A.$$

Bazaraa and Sherali [1980]

Using the formulation in Pierce et al. [1971], it is shown by using a convenient transformation that, by denoting

$$c_{ijkl} \;=\; \frac{a_{ij} + a_{kl}}{(m-1)} \;+\; \sum_{h=1}^{R}(b_{ijkl}^{h} + b_{klij}^{h}) \;+\; \sum_{h=1}^{R}(p_{ik}^{h}d_{jl}^{h} + p_{ki}^{h}d_{lj}^{h})$$
$$i = 1,\ldots,m-1; \;\; k = i+1,\ldots,m; \;\; l,j = 1,\ldots,n, l \ne j$$

the problem can be equivalently stated as

$$\underset{x \,\in\, X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m-1}\sum_{j=1}^{m}\sum_{k=i+1}^{m}\sum_{\substack{l=1\\ l\ne j}}^{m} c_{ijkl}x_{ij}x_{kl}. \tag{2.67}$$

Equation (2.67) is then reformulated into a linear mixed integer problem, having $m^2(m-1)^2/2$ new variables and $2m(m-1)$ new linear constraints. The resulting formulation is as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m-1} \sum_{j=1}^{m} \sum_{k=i+1}^{m} \sum_{\substack{l=1 \\ l \neq j}}^{m} c_{ijkl} y_{ijkl} \tag{2.68}$$

$$\text{subject to} \quad \sum_{k=i+1}^{m} \sum_{\substack{l=1 \\ l \neq j}}^{m} y_{ijkl} = (m-i) x_{ij} \quad i = 1, \ldots, m-1, \tag{2.69}$$

$$j = 1, \ldots, m$$

$$\sum_{i=1}^{k-1} \sum_{\substack{j=1 \\ j \neq l}}^{m} y_{ijkl} = (k-1) x_{ij} \quad k = 2, \ldots, m; \quad l = 1, \ldots, m \tag{2.70}$$

$$x \in X_A, \quad \mathbf{0} \leq \mathbf{y} \leq \mathbf{1}.$$

Christofides, Mingozzi, Toth [1980]

Assuming p_{ik} and d_{jl} to be symmetric, the QAP can be formulated as follows:

$$\underset{x \in X_A}{\text{Minimize}} \quad Z = \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} \sum_{k=i+1}^{m} \sum_{l=j+1}^{m} p_{ik} d_{jl} x_{ij} x_{kl}. \tag{2.71}$$

Introducing n^4 y_{ijkl} variables, an equivalent linear mixed integer program is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} \sum_{k=i+1}^{m} \sum_{l=j+1}^{m} p_{ik} d_{jl} y_{ijkl} \tag{2.72}$$

$$x_{ij} + x_{kl} \leq y_{ijkl} + 1 \tag{2.73}$$

$$x \in X_A, \quad \mathbf{y} \text{ binary.}$$

Frieze and Yadegar [1983]

Three linearized integer programming representations of the first QAP formulation due to Lawler [1963] (Equation (2.41)) are presented. The first of these can be stated as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik}d_{jl}y_{ijkl} \tag{2.74}$$

$$\text{subject to} \quad \sum_{i=1}^{m} y_{ijkl} = x_{kl} \quad j,k,l = 1,\ldots,m \tag{2.75}$$

$$\sum_{j=1}^{m} y_{ijkl} = x_{kl} \quad i,k,l = 1,\ldots,m \tag{2.76}$$

$$\sum_{k=1}^{m} y_{ijkl} = x_{ij} \quad i,j,l = 1,\ldots,m \tag{2.77}$$

$$\sum_{l=1}^{m} y_{ijkl} = x_{ij} \quad i,j,k = 1,\ldots,m \tag{2.78}$$

$$y_{ijij} = x_{ij} \quad i,j = 1,\ldots,m \tag{2.79}$$

$$x \in X_A, \quad \mathbf{0} \leq \mathbf{y} \leq \mathbf{1}.$$

In the second formulation, an equivalence can be shown by summing subsets of constraints within each of (2.75)- (2.78) so that the resulting constraints from (2.75) and (2.76) are identical, and so are the ones obtained from (2.77) and (2.78). The equivalent problem is posed as follows:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik}d_{jl}y_{ijkl} \tag{2.80}$$

$$\text{subject to} \quad \sum_{i=1}^{m}\sum_{j=1}^{m} y_{ijkl} = mx_{kl} \quad k,l = 1,\ldots,m \tag{2.81}$$

$$\sum_{k=1}^{m}\sum_{l=1}^{m} y_{ijkl} = mx_{ij} \quad i,j = 1,\ldots,m \tag{2.82}$$

$$y_{ijij} = x_{ij} \quad i,j = 1,\ldots,m \tag{2.83}$$

$$x \in X_A, \quad \mathbf{0} \leq \mathbf{y} \leq \mathbf{1}.$$

The third formulation adopts the form of a Lagrangean relaxation subproblem of the first formulation. Keeping everything defined as before, let:

- $\alpha_{jkl} \equiv$ a multiplier selected for constraint (2.75)

- $\beta_{ikl} \equiv$ a multiplier selected for constraint (2.76).

The Lagrangean subproblem, that is equivalent to the original problem in this case, is then defined by

$$\begin{aligned}
\text{Minimize} \quad Z = & \sum_{i=1}^{m}\sum_{j=1}^{m} \bar{a}_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} \bar{p}_{ik}\bar{d}_{jl}y_{ijkl} && (2.84)\\
\text{subject to} \quad & \sum_{k=1}^{m} y_{ijkl} = x_{ij} \quad i,j,l = 1,\ldots,m && (2.85)\\
& \sum_{l=1}^{m} y_{ijkl} = x_{ij} \quad i,j,k = 1,\ldots,m && (2.86)\\
& y_{ijkl} = y_{klij} && (2.87)\\
& y_{ijij} = x_{ij} \quad i,j = 1,\ldots,m && (2.88)\\
& x \in X_A, \ \ \mathbf{0} \le \mathbf{y} \le \mathbf{1},\\
\text{where} \quad & \bar{a}_{ij} = a_{ij} + \sum_{l=1}^{m} \alpha_{jkl} + \sum_{k=1}^{m} \beta_{ikl}\\
& \bar{p}_{ik}\bar{d}_{jl} = p_{ik}d_{jl} - \alpha_{jkl} - \beta_{ikl}.
\end{aligned}$$

<u>Assad and Xu [1985]</u>

An equivalent mixed-integer reformulation of the QAP is presented as follows:

$$\begin{aligned}
\text{Minimize} \quad Z = & \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik}d_{jl}y_{ijkl} && (2.89)\\
\text{subject to} \quad & \sum_{i=1}^{m}\sum_{j=1}^{m} y_{klij} = m x_{kl} \quad \forall \ k,l = 1,\ldots,m && (2.90)\\
& \sum_{i=1}^{m} y_{ijkl} = x_{kl} \quad \forall \ j,k,l = 1,\ldots,m && (2.91)\\
& \sum_{j=1}^{m} y_{ijkl} = x_{kl} \quad \forall \ i,k,l = 1,\ldots,m && (2.92)\\
& x \in X_A, \ \ \mathbf{y} \ge \mathbf{0}.
\end{aligned}$$

Johnston [1992]

A mixed integer linear reformulation of the QAP is presented. The continuous relaxation provides a tight lower bound on the optimal objective function value. These bounds are shown to be at least as large as any alternate linear representation, and moreover, the constraints of a majority of the previous linear reformulations can be obtained by surrogating constraints of the following representation:

$$\text{Minimize} \quad Z = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij}x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m}\sum_{l=1}^{m} p_{ik}d_{jl}y_{ijkl} \tag{2.93}$$

$$\text{subject to} \quad \sum_{i=1}^{m} y_{ijkl} = x_{kl} \quad \forall\ j,k,l = 1,\ldots,m \tag{2.94}$$

$$\sum_{j=1}^{m} y_{ijkl} = x_{kl} \quad \forall\ i,k,l = 1,\ldots,m \tag{2.95}$$

$$y_{ijkl} = y_{klij} \quad \forall\ i,j,k,l = 1,\ldots,m \tag{2.96}$$

$$x \in X, \quad \mathbf{y} \geq \mathbf{0}.$$

2.3.2 Branch-and-Bound Techniques

The branch-and-bound technique is a popular exact solution procedure for the QAP. Many researchers have developed such solution procedures for the QAP including: Gilmore [1962], Lawler [1963], Gavett and Plyter [1966], Graves and Whinston [1970], Pierce and Crownston [1971], Heider [1973], Kaufman and Broeckx [1978], Burkard and Stratman [1978], Bazaraa and Elshafei [1979], Bazaraa and Sherali [1980], Christofides, Mingozzi, and Toth [1980], Bazaraa and Kirca [1983], Rendl [1985], Kaku and Thompson [1986], Roucairol [1987], Christofides and Benavent [1989], Carraresi and Malucelli [1992a], and Rendl and Wolkowicz [1992].

After applying a linearization technique, the problem is relaxed and solved as a continuous problem at node zero of the enumeration tree. Since the x_{ij}'s are binary, each node, if necessary, is partitioned along two branches, namely, $x_{ij} = 0$ and $x_{ij} = 1$. Each node represents a restricted subproblem that satisfies the conditions imposed on all the branches on the chain from it to node zero. If the solution is infeasible, then the node is fathomed. To complete the assignment, a feasible permutation of the remaining m', say, facilities on the remaining locations must be obtained. However, in this enumeration, there can be as many as $m'!$ solutions to enumerate in order to complete the assignment. To avoid the enumeration of all such permutations, lower bounds are computed. These lower bounds investigate whether there could exist a solution having an objective value smaller than the existing solution. If it can be proven that no such solution exists, then the current node can be fathomed.

The lower bound computation for any partial solution can be divided into the following three components:

- $C_1 \equiv$ interaction cost between the assigned facilities,
- $C_2 \equiv$ interaction cost between the assigned facilities and the unassigned facilities, and
- $C_3 \equiv$ interaction cost between the unassigned facilities.

The importance of lower bounds is tremendous, since these lower bounds can fathom a great percentage of the enumeration tree, significantly reducing the overall computational effort.

The literature on branch-and-bound procedures for the QAP differs mainly in the computation of the lower bound. These include:

Single assignment algorithms: A particular facility is associated to correspond to each level in the branching process. For each location, the distance from all other locations is computed and arranged in a nonincreasing order. Facilities are then assigned to each level in this order.

Pair assignment algorithms: Pairs of facilities i and k are assigned to locations j and l. If the flow from i to j and j to i is symmetric, then there are $\frac{m(m-1)}{2}$ pairs of facilities and $\frac{m(m-1)}{2}$ pairs of locations. If the flow between the two facilities is asymmetric, there are $m(m-1)$ pairs of facilities and locations. Feasible assignments of these pairs does not guarantee a feasible solution to the original problem. In initializing this process, one facility may be assigned to two locations. To obtain a feasible assignment to the original problem, additional constraints are added to the linear pair assignment problem. These constraints ensure that if facilities i and k are allocated to locations j and l, then the remaining allocations of facilities i and k will involve locations j and l, and vice versa. The solution technique proceeds level by level, where pairs of facilities are fixed to pairs of locations. After a node is fathomed, a standard backtracking technique is employed.

[illegible]

Pair exclusion algorithms: First proposed by Pierce and Crownston [1971], this algorithm proceeds on the basis of a stage-by-stage exclusion of assignments from a solution to the problem. As in the pair assignment algorithm, pairs of assignments are created and the additional constraints are imposed. If the relaxed problem assigns infeasible pairs, then these assignments are infeasible in the optimal solution. Hence, the search branches into as many nodes as there are conflicting assignments, eliminating the conflicting assignments at each ensuing node. Further branching is necessary if no lower bound that fathoms the node is found or if the problem remains infeasible. If a feasible solution is obtained, then the node is fathomed. At any stage, the procedure selects another node with the lowest bound and proceeds as before, continuing until all nodes are fathomed.

2.3.3 Heuristic Techniques

Recognizing that branch-and-bound techniques could only solve the QAP for up to dimension size 15, heuristic techniques have been widely applied for solving larger sized QAP's arising in real practical problems. The literature on heuristic techniques include the following: Steinberg [1961], Hillier and Connors [1961], Gaschütz and Ahrens [1968], Elshafei [1977], Burkard and Derigs [1980], Lashkari and Jaisingh [1980], Bazaraa and Sherali [1980, 1982], Liggett [1981], Murtagh, Jefferson, and Sornprasit [1982], Bazaraa and Kirca [1983], Burkard and Bönniger [1983], West [1983], Picone and Wilhelm [1984], Sherali and Rajgopal [1986], Kaku, Thompson, and Morton [1991], Carraresi and Malucelli [1992b], Hadley, Rendl, and Wolkowicz [1992a, 1992b] and Chakrapani and Skorin-Kapov [1993].

The heuristic procedures that have been developed for solving the QAP can be classified into three groups:

Construction: This scheme is an n-stage decision process for intelligently building a solution from scratch. Facilities are selected and placed at locations sequentially, proceeding until all facilities have been placed. Facilities with larger flows are generally placed closer together while facilities with smaller flows are placed further away. This procedure generally produces a fast solution, however, the solution is not very competitive, until an improvement scheme is applied.

Improvement: This scheme begins with an initial solution and attempts to incrementally improve it. The most common improvement scheme is the pairwise interchange process. Taking the construction scheme, it systematically improves the solution, randomly or by some rule involving less computational effort, by interchanging two pairs of facilities at a time to check for any improvement. Other improvement schemes in the literature are considerably more complex than the pairwise interchange technique, providing more complex heuristic techniques. Examples include running exact algorithms on smaller sized subproblems.

Adaptation of Exact Schemes: By either prematurely terminating an exact solution procedure, or else, by selectively adopting judicious steps within some formal exact algorithm, this strategy attempts to prescribe heuristic solutions with a reasonable effort.

A brief summary of some of the proposed heuristic techniques for solving the QAP are presented below.

Elshafei [1977]

In using a direct application of the QAP to a hospital layout, consisting of locating 19 facilities to 19 locations to minimize patient travel, a heuristic was developed. A general construction scheme ranks the locations in ascending order by distance and ranks the facilities in descending order by flow, and matches these ranked facilities to the ranked location. In the improvement scheme, pairwise exchanges of assignments are conducted, consisting of 2-way exchanges and 3-way exchanges. The heuristic led to an improvement of 19.2% in the total patient travel. The layout was adopted by hospital management.

Lashkari and Jaisingh [1980]

The algorithm presented here employs a sequential search technique, constructing a matrix of lower bounds on the costs of locating facilities at different locations $(p_{ik}d_{jl})$. Modifications of the elements of this matrix are made by solving a succession of linear assignment problems. After all the elements of the matrix are determined, a feasible assignment is obtained, resulting in an improved objective function value of the QAP. The procedure is repeated until the desired objective function value accuracy is obtained or if no improvement in the objective function value continues. The algorithm produces near-optimal results and is independent of a starting solution. The results obtained compare reasonably well with the best known solutions to date.

Liggett [1981]

This paper employs a constructive technique to generate an initial solution to a problem, allowing the solution to be modified using a variety of heuristic improvement procedures. The construction technique combines a general enumerative procedure with probability theory to yield an implicit enumeration algorithm, modified to include a back-tracking strategy. A simple pairwise exchange is incorporated as the standard improvement procedure. Good solutions were produced in very reasonable time.

Bazaraa and Sherali [1982]

This study addressed the use of exact and heuristic cutting plane methods for solving the QAP. The cutting plane methods used are intersection cuts and disjunctive cuts. It is recognized that the use of exact cutting plane methods would require a huge computational effort in finding the solution. Therefore, several heuristics are derived from the cutting planes to produce optimal or good quality solutions early on in the search process. Initial starting solutions were generated, then the heuristics further improved upon the initial run solutions. The heuristic produced the best known solution available in literature to date.

Murtagh, Jefferson, and Sornprasit [1982]

The procedure presented here consists of a simple scheme to get a good feasible starting point, then solve the problem as a nonlinear (quadratic) program using MINOS, ignoring the integrality conditions, and finally to convert the near integer solution into an integer feasible solution using a heuristic procedure. This heuristic procedure obtains the 'closest' integer feasible solution by partitioning the variables into sets determined by each facility, determining the largest variable in each set, ordering the sets to be considered in decreasing value of this largest member, and then prescribing the assignments. Computational tests show that the relative performance of this method improves as the problem size gets larger.

<u>Bazaraa and Kirca [1983]</u>

A branch-and-bound algorithm is proposed by using the symmetric properties of the problem to eliminate "mirror image" branches, thus reducing the search effort. Several routines transforming the procedure into an efficient heuristic are implemented, including 2-way and 4-way exchanges, selective branching rules, and the use of variable upper-bounding techniques for enhancing the speed of fathoming. As an exact scheme, the problem solved the 12 facility problem of Nugent et al. [1968] and the 19 facility problem of Elshafei [1977]. As a heuristic, best known solutions were produced for all well known problems, and produced improved solutions in several cases.

<u>Burkard and Bönniger [1983]</u>

An algorithm is presented for the quadratic Boolean program, with a direct application to the QAP. Cutting planes are obtained directly without the use of Benders' decomposition by a linearization technique. This new heuristic needs a storage capacity of only $6n^2 + 11n$ bytes, being independent of the number of iterations and cuts. Computational tests exhibited a stable behavior: independent of the starting solution, the suboptimal solution deviated from the best known solution by less than 5% in the worst case, and in average, less than 1%. The best known values have not been found by applying a fixed method just once, but turned out more or less incidentally in a long series of tests with different methods.

West [1983]

The heuristic presented is intended for the many applications in which the QAP's are large ($m > 15$), but a good suboptimal solution is acceptable. An algorithm is developed to examine all possible pairwise exchanges among the assignments, after some facilities have been located. Tests were conducted for assymmetric and symmetric assignments.

Sherali and Rajgopal [1986]

This paper describes a flexible, polynomial-time heuristic procedure, effective with respect to the quality of solutions obtained, efficient with respect to computational effort, and capable of compromising between these two measures of effectiveness as may be necessary. The parameters varied within the heuristic determine the size and number of quadratic assignment subproblems that are solved exactly as well as the number of loops through the improvement routines. They may be varied as desired in order to trade-off between solution quality and computational effort. Computational experience shows that solutions found were near to the best known solutions existing, in much less computational effort than before.

2.3.4 Graph Theoretical Techniques

A graph G is defined by the set (V, E), where V is a nonempty set of vertices and E is a set of links. Each link has a cost of a_{ij}.

Christofides and Gerrard [1981]

The QAP is expressed in terms of graph multiplication of a flow graph G^f with a distance graph G^d. When G^f is decomposed into simpler graphs, a general unified procedure to calculate bounds is given, generating two previously known bounds as special cases and also generating some new bounds. By enumerating all practical decompositions, it is shown that no better bounds within this class can be computed in reasonable time, other than the bounds presented herein.

Flood [1990]

This study presented algorithms for finding exact and approximate solutions for the weighted feedback arc set problem. The objective was to determine a minimum-cardinality set of arcs that breaks all cycles in a directed graph. This problem is also that of finding a group rank ordering given the rank orders for each member of a group. The algorithm developed by Edwards [1980] was utilized, given that this problem is a special case of the QAP.

Assad and Xu [1992]

This paper shows that the QAP can be polynomially transformed into the quadratic minimum spanning tree problem (QMST), which is therefore also an NP-hard problem. A lower bound is presented, followed by two exact branch-and-bound techniques, utilizing this lower bound. Recognizing that solving this to optimality is highly time consuming even for problems of moderate size, two heuristic algorithms are presented. Computational results favor the heuristic methods much more than the branch-and-bound scheme.

2.3.5 Simulated Annealing

Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller [1953] were the first to derive a Monte-Carlo method for simulating the collection of atoms in contact with a heat-bath, achieving thermal equilibrium. A brief statement of the procedure is as follows:

> Given a configuration of the elements of the system, randomly displace the elements, one at a time by a small amount, and calculate the resulting change in energy, ΔE. If $\Delta E < 0$ then accept the displacement and use the resulting configuration as the starting point for the next iterations. If $\Delta \mathrm{E} \geq 0$, then the displacement is accepted with probability $P(\Delta E) = \exp(-\Delta E/k_b T)$, where T is the temperature and k_b is Boltzmann's constant. (This constant is not required when applying the Metropolis et al. algorithm to combinatorial problems.)

This improvement procedure bearing some similarity to the biased sampling approach has been termed **simulated annealing**. The Monte Carlo sampling aspect is incorporated by comparing $P(\Delta E)$ with a random variable drawn from a (0,1) uniform distribution. This process continues until equilibrium is achieved. The temperature is then lowered according to the annealing schedule and the procedure is repeated until the system freezes. The annealing schedule allows the simulation to proceed long enough for the system to reach steady state at each temperature.

Applying simulated annealing to the QAP may be an effective and efficient way of solving this problem. If each feasible assignment of facilities to sites in a QAP is viewed as a configuration of atoms in the mechanical system, and the value of the objective function is viewed as the energy of the system, then determining the least cost assignment of facilities to sites is analogous to finding the arrangement of atoms in the mechanical system which results in the lowest energy state. As the temperature decreases sufficiently, the random search process is frozen and the best solution available thus far is the prescribed heuristic solution.

Given below is a discussion on the literature dealing with applications of simulated annealing to the QAP.

Burkard and Rendl [1984]

This was the first article to apply the simulated annealing procedure to the QAP. A heuristic is presented by specializing a simulated annealing procedure that is applicable for general combinatorial optimization problems to the QAP. Problems of dimension up to 36 were tested. Overall, the procedure produced solutions within one to two percent of the best known solutions within reasonable computation effort.

By starting the procedure several times at different solutions, all known minimal objective function values were reached.

Bonomi and Lutton [1986]

The idea of statistical mechanics is presented in applying it to the QAP. The formalization of this allows the exhibition, in the limit of large problems, of asymptotic behavior of the optimal value of the cost function. The temperature is used as an external parameter which controls fluctuations of the random walk, among the set of admissible solutions, generated by the simulation. It is shown that a very modest cost saving can be obtained for small systems. But this economy disappears quickly as the dimension of the problem increases.

Wilhelm and Ward [1987]

Stating that Burkard and Rendl [1984] did not provide very many details regarding the settings of the control parameters used in the algorithm, an algorithm was developed, performing better than that of Burkard et al. [1984]. The motivation was to avoid excessive run times, particularly at low temperatures when the system is "freezing-up". Likewise, if the desired number of acceptances of location interchanges is not achieved at three successive temperatures, the system is considered frozen and the last, least cost facility assignment is considered the best facility location assignment. Problems of up to dimension 100 were tested, and a larger number of runs were made for the simulated annealing algorithm under different combinations of parametric values. However, it is concluded that simulated annealing is a promising approach for solving combinatorial optimization problems like the QAP, warranting more research.

Connolly [1990]

A much-improved simulated annealing scheme is presented, performing well on a range of examples, finding improved solutions for several of the largest problems available in literature, requiring only modest amounts of computational effort. This much-improved algorithm controls how the attempted assignments are selected and how the temperature is determined and reduced. Problems of dimension up to 100 were tested, comparing results from Nugent, et al. [1968], Reeves [1985], and Wilhelm and Ward [1987]. From this study, four conclusions were reached:

- Simulated annealing is an extremely efficient heuristic for the QAP.
- A sequential generation of neighbors is superior to a random selection method in an annealing scheme.
- There exists a fixed temperature at which the performance of an annealing scheme is optimized.

A scheme incorporating these ideas has been written which performs extremely well, finding improved solutions for several of the largest problems in the literature in only modest amounts of CPU time without the need to "tune" the system for each new data set.

2.3.6 Tabu Search

As stated in Glover [1989], "Tabu search is a strategy for solving combinatorial optimization problems whose applications range from graph theory and matroid settings to general pure and mixed integer programming problems." Tabu search was introduced as a technique to overcome local optimality with the idea of forbidding some search directions (moves) at a present iteration in order to avoid cycling. To date, tabu search techniques applied to the QAP have obtained the best results reported in the literature.

We present a brief overview of the tabu search as stated in Glover [1990]. The following is defined:

- c(x) $\equiv$ objective function, may be linear or nonlinear, and may also incorporate penalty function components to drive toward satisfying certain types of constraints.

- $x \in X_T \equiv$ constraining conditions which will be maintained at each step of the search, and in many contexts of interest, will require specified components of x to receive discrete values, except in special strategic variations.

- $s : X_T(s) \rightarrow X_T \equiv$ a move s that leads from one trial solution to another being viewed as a mapping defined on a subset $X(s)$ of X_T.

- $S(x) \equiv$ a set consisting of the moves $s \in S$ that can be applied to x; i.e. $S(x) = \{s \in S : x \in X_T(s)\}$. Hence, $X_T(s) = \{x \in X_T : s \in S(x)\}$.

- $T \equiv$ set of tabu moves, used as a starting point.

- OPTIMUM $\equiv$ evaluator function.

The optimization problem is presented as:

$$\text{(P)} \quad \text{Minimize} \quad \text{c}(x) : x \in X_T \text{ in } R_m.$$

The following is an algorithm for a *simple tabu search*:

1. Select an initial $x \in X_T$ and let $x^* := x$. Set the iteration counter $k = 0$ and begin with T empty.

2. If $S(x) - T$ is empty, go to Step 4.

   Otherwise, set $k := k+1$ and select $s_k \in S(x) - T$ such that $s_k(x) = \text{OPTIMUM}(s(x) : s \in S(X_T) - T)$.

3. Let $x := s_k(X_T)$. If $\text{c}(x) < \text{c}(x^*)$, where x^* denotes the best solution currently found, let $x^* : x$.

4. If a chosen number of iterations has elapsed either in total or since x^* was last improved, or if $S(x) = \oslash$ upon reaching this step directly from Step 2, stop. Otherwise, update T and return to Step 2.

The following is a brief summary of literature that applies the tabu search technique to the QAP.

Skorin-Kapov [1990]

This was the first instance of applying the tabu search to the QAP. A heuristic procedure was developed for tabu-navigation, consisting of a construction phase to develop an initial solution to the QAP, and an improvement phase. The method implemented is flexible in that it allows the user to interact and change the parameters (tabu list size, iteration limit, a search diversification parameter, and the number of new starting solutions) during the run.

Results indicate that good tabu list sizes increase with the dimension of the problem. Problems of dimension 42 through 90 were tested, delivering, in some cases, better solutions than the simulated annealing technique, but requiring greater effort as the problem size increased. Nonetheless, the method appeared to be very efficient when tested on the standard problems of Nugent, et al. [1968].

Skorin-Kapov [1991]

This paper presents another tabu-search heuristic algorithm for the QAP. The distinct feature of the method is its continuous learning and dependence on past searches, using the knowledge about good solutions previously encountered in the search process. It is a refinement of the previous tabu search adaptation by Skorin-Kapov [1990] in three directions. First, the evaluation function is modified by incorporating the idea of target analysis to guide the nonimproving moves. Second, the possibility to change the definition of the neighborhood where the evaluation function operates at different stages in the search process is provided. Third, the tabu list size is changed dynamically by incorporating dynamic gaps in it corresponding to deleting the tabu status of a subset of moves at certain iterations. The algorithm was found to perform well for larger size problems.

Taillard [1991]

The tabu search has been applied to the QAP by performing multiple solution passes for each problem, starting from different initial solutions and modifying parameters between the solutions. This paper proposes a more robust form of the tabu search for the QAP by fixing its parameters a priori, leaving a minimum number of free parameters and being very easy to implement. The method is efficient and succeeds in obtaining sub-optimal solutions for problems up to size 64. However, the search for sub-optimal solutions for larger size problems must be done by other procedures, using more elaborate concepts of the tabu search.

Chakrapani and Skorin-Kapov [1992a]

The purpose of this paper was to apply a Boltzmann machine, a connectionist model that uses simulated annealing, and a related connectionist model in which the escape from local optima is performed in a deterministic way using tabu search to the QAP. The paper determined that the deterministic approach was an improvement upon a Boltzmann machine. Further analysis led to a new, improved, massively parallel computational model based on connectionist architecture.

The model enables simulated annealing to obtain starting solutions, then makes use of tabu search to improve the resulting upon those solutions, finding better solutions to larger size problems. The tabu search is favored over the stochastic simulated annealing procedure.

Chakrapani and Skorin-Kapov [1992b]

This paper presents a heuristic algorithm based on a new, efficient tabu search strategy for the QAP, with special emphasis on applicability to larger problem sizes, and provides a massively parallel implementation of this strategy. The algorithm, known as *Par_tabu*, was developed to perform effectively while keeping the number of iterations relatively small. It uses a preliminary phase that terminates quickly, identifying the best solution found thus far. An intensification phase then focuses the search around the current best solution found until no improvement is obtained for a fixed number of iterations. Then long term memory is invoked to diversify the search toward unexplored regions. At the same time, a new intensification strategy based on intermediate term memory to restrict neighborhoods is proposed. This is known as *Augmented Par_tabu.*

The computations were run on the Connection Machine system CM-2, a massively parallel machine. The advantages of the CM-2 is that the benefits increase with the dimension of the problem. QAP problems of dimension between 42 and 100 were tested. The tabu search strategy proved to be very efficient in terms of solution quality, obtaining best known or close to best known solutions for problems of sizes up to 90, and improved upon known solutions for problems of size 100. The *Augmented Par_tabu* resulted in further improved solutions to larger problems, at the cost of performing more iterations. The increase in time per iteration was found to be a logarithmic function of the size of the problem.

2.3.7 Genetic Algorithms

Genetic algorithms are a family of parallel, randomized-search optimization heuristics which emulate biological natural selection on a population of feasible solutions. It is widely recognized that genetic algorithms are not well suited to performing finely-tuned local search. Like natural systems, genetic algorithms progress by virtue of changing the distribution of high performance substructures in the overall population where individual structures are not the focus of attention. Once the high performance regions of the source space are identified by the genetic algorithm, it may be useful to invoke a local search routine to optimize the members of the final population. Genetic algorithms contain the following features:

1. One or more "populations" of feasible solutions.

2. A mechanism for generating new feasible solutions by combining features from multiple previously-known solutions (**breeding** or **reproduction**).

3. A mechanism for generating a new feasible solution by a random perturbation of a single previously-known solution (**mutation**).

4. A mechanism for selecting individual solutions from the population(s), giving preference to those with better objective function values (**selection**).

5. A mechanism for removing solutions from the population(s) (**culling**).

The basic flow of any genetic algorithm implementation is given by the following:

1. Create an initial population or sub-populations of solutions, usually feasible and random.
2. Repeat:
   (a) **Select** parents.
   (b) **Breed** offsprings, and add them to the population.
   (c) **Mutate** certain members of the current population.
   (d) **Cull** certain members of the current population.
3. Terminate when set criteria is met.

The genetic algorithm has proven to be most effective on nonconvex optimization problems for which it is relatively easy to assess the quality of a given feasible solution, but difficult to systematically improve solutions by deterministic iterative methods. Most NP-complete combinatorial problems, such as the QAP, fall into this category.

The following is a brief summary of the literature that applies genetic algorithms to the QAP.

Huntley and Brown [1991]

This study describes a heuristic procedure, combining a parallel hybrid of simulated annealing and the genetic algorithm, with its intended use on parallel computers. The genetic algorithm used is specifically adapted to the QAP through its crossover operator and parent selection procedure. The crossover operator inserts a permutation of one parent directly into the data structure of its mate. The parent selection procedure is biased toward selecting very good solutions in the population, which drives the algorithm more rapidly in the direction of superior candidate assignments. In a similar fashion, the simulated annealing portions were designed to account for the quality of the solutions passed by the genetic algorithm.

Computational results show that the algorithm is not as efficient for problems of smaller size. However, large-scale problems show the algorithm to be superior. The authors conclude that additional work needs to be conducted for improving the algorithm.

<u>Tate and Smith [1992]</u>

An investigation was conducted using a particular genetic algorithm for the QAP. The motivation of using the genetic algorithm was to improve upon the heuristic techniques already applied to the QAP. It was found that the genetic algorithmic approach yielded solutions comparable to those of the best previously reported heuristics without extreme computational requirements, where the best solutions found over a moderate number of test runs were invariably within a few percentage points of these best known solutions. There appears to be strong advantages to encodings and generation operators that are as problem specific as possible, and that preserve feasibility in all cases. It is concluded that the extension of the genetic algorithmic approach to more complex assignment and placement problems will also be effective and practical.

2.3.8 Survey and Application Papers

Survey papers on the QAP have basically presented other authors methods of linearizing the QAP, techniques for solving the QAP, and in some cases, presenting a new technique themselves. Application literature on the QAP have taken a developed technique, modified it, and applied it to a real world problem. Related papers include: Hanan and Kurtzberg [1972], Foulds and Robinson [1976], Geoffrion and Graves [1976], Sahni and Gonzalez [1976], Burkard and Fincke [1983], Heragu and Kusiak [1988], LaPorte and Mercure [1988], Burkard [1990], and Cohoon, Hegde, Martin, and Richards [1991]. The following presents summaries on the survey literature.

Nugent, Vollman, and Ruml [1968]

This paper has turned out to be one of the classic papers on the QAP. Solution techniques on the QAP, to date, are presented. But what makes this paper a classic is the published set of test problems of sizes 5, 6, 7, 8, 12, 15, 20, and 30. Since 1968, most authors have used this data to compare their algorithm against other competing procedures.

Parker [1976]

This study examined the QAP heuristic approaches to date. Four "construction" and nine "improvement" algorithms were selected for investigation. The construction procedures were chosen primarily to evaluate the effects of the quality of starting solution on the improvement methods. The improvement methods were selected to test some basic strategies of pairwise-interchanges of components. Seventy-five problems were generated and tested, comparing them with respect to the produced solution quality and CPU run-time requirements. A construction approach due to Graves and Whinston [1970] produced the best results, both when used to generate starting solutions for the improvement methods and when evaluated on its own merit against the improvement methods using other construction approaches.

Burkard and Stratmann [1978]

A survey of the numerical behavior of different algorithms for solving the QAP is presented. After a discussion of branch-and-bound algorithms, modifications by different researchers are presented. Finally, tables are presented based on the different numerical results from these different algorithms and modifications.

Brujis [1984]

This paper compares the heuristic solutions of a QAP of dimension size 19 of Elshafei [1977], Lashkari and Jaisingh [1980], and Murtagh, Jefferson, and Sornprasit [1982]. The author shows that the concept of a 'good initial solution' is misleading in the way that the quality of the initial and final solution prove to be rather weakly (or even negatively) correlated. A simple, but broad and randomized search results in significantly better solutions, while simple statistical aids provide an estimate of the quality of these solutions. He concludes, based on Elshafei [1977], that when solving a QAP heuristically, it is advisable to continue the search until a statistical evaluation of the results does deliver a satisfactory confidence interval for the value of the optimal solution.

Burkard [1984]

This paper discusses which author, to date, presents the smallest linearized reformulation of the QAP, presents the use of branch-and-bound techniques, remarks on the need for good heuristics, and on the use of graph theoretical techniques use in solving a special case of the QAP. The idea of simulated annealing is also presented.

Finke, Burkard, and Rendl [1987]

Many applications of the QAP are discussed along with what different researchers have done in the actual context of applying it to real world problems. Integer programming formulations, solution methodologies such as exact branch-and-bound, heuristic, and eigenvalue approaches are presented. It is concluded that there exists no common trend for the different types and sizes of problems, but probably, this is simply a characteristic of a combinatorial problem of such extreme difficulty.

Kusiak and Heragu [1987]

The linearization techniques discussed above are presented in this review in a much scaled down version, stating only the major developments. Computational complexity is shown to grow, almost exponentially, as the problem size increases. Algorithms such as branch-and-bound, cutting plane, construction, improvement, hybrid, and graph theoretic are summarized. A table is presented showing the computational times on selected algorithms.

Burkard [1990]

Just as in Finke, Burkard, and Rendl [1987], many applications of the QAP are presented in an updated version which includes simulated annealing. Again, no common trend exist for the size and type of problems.

Francis, McGinnis, and White [1992]

In this second edition of their book, a section is presented on the QAP. Different subsections present the problem formulation and various construction heuristics, improvement heuristics, and simulated annealing approaches.

Chapter 3 presents a quadratic model with set packing types of constraints for the gate assignment problem to minimize total passenger walking distances subject to airport/airline regulatory constraints. Preprocessing steps for model refinements, along with a linearization scheme are also presented. Tasks for exploring possible solution techniques are outlined in Chapter 4.

Chapter III

Model Construction and Preprocessing Routines

This chapter presents a model formulation to minimize total passenger walking distances. We start with the model assumptions and a description of the problem, followed by a formulation of the model. Preprocessing routines and a linearized reformulation of the problem are also presented. Our analysis is based on the literature dealing with gate assignments and the practice at *USAir*.

In particular, our model follows the same concepts as in Mangoubi and Mathaisal [1985]. However, we present the constraints differently, avoiding the inclusion of redundant constraints and variables more directly, and we also consider the walking distances of transfer passengers more accurately by explicitly modelling their arrival and departure flight locations via a quadratic term. Additional features such as neighboring restrictions, the assignment or preclusion of a pair of flights to neighboring gates when enforced, and customs gate requirements are also incorporated.

Throughout this chapter, an example consisting of 8 flights and 4 gates will be utilized. Figure 3.1 displays the layout of the terminal. Table 3.1 represents the partial distance matrix, assumed to be symmetric, between the 4 gates, check-in, and baggage claim. The diagonal entries are equal to zero. Table 3.2 presents the flight schedule which includes the cities each flight arrives from and departs to, time of arrival and departure, flight numbers, and the type of aircraft for each flight. Table 3.3 contains the data regarding transferring passengers between the arriving and departing flights. (Note: **CI** denotes Check-in, **BC** denotes baggage claim, **I** denotes International and **D** denotes Domestic.)

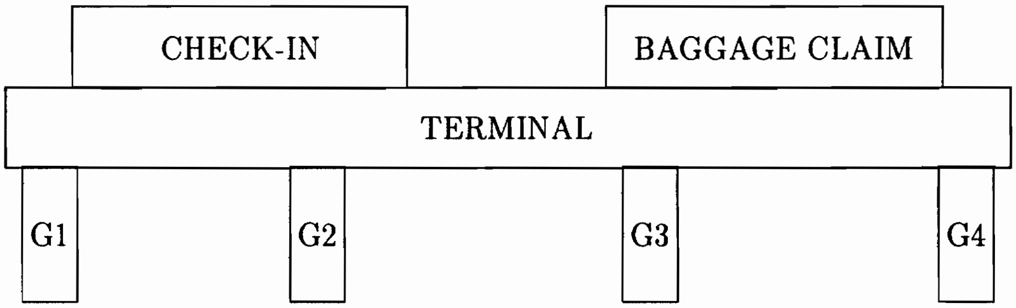


Figure 3.1: **Terminal Layout**

Table 3.1: **Distances Between Gates, Check-In, and Baggage Claim**

	G1	G2	G3	G4	CI	BC
G1		200	450	700	200	700
G2			250	500	200	450
G3				250	450	200
G4					700	200

Table 3.2: **Flight Schedule**

FLIGHT	CITY FROM	CITY TO	TIME OF ARRIVE	TIME OF DEPART	FLIGHT # IN	FLIGHT # OUT	AIRCRAFT SIZE
F1	FRA	MIA	12:00	13:00	100	101	737
F2	SEA	IAD	12:15	13:00	200	201	757
F3	HNL	BOS	12:45	13:15	300	301	762
F4	LAX	RDU	13:00	13:15	400	401	737
F5	DEN	FRA	13:30	14:30	500	501	74L
F6	RDU	ORD	13:45	14:30	600	601	737
F7	MEM	BWI	13:45	14:30	700	701	737
F8	ATL	AUS	14:00	14:30	800	801	737

RESTRICTIONS: **Gates 1,2 restrict 757's parking next to each other**
Gates 1,2,3 restrict the occupancy of large size aircraft
Gate 4 is the customs gate

Table 3.3: **Passenger Distribution Between Flights, CI, and BC**

		To Flights									
		F1	F2	F3	F4	F5	F6	F7	F8	CI	BC
	F1	10	5	7	8	10	6	20	30		6
	F2	7	10	4	2	3	7	10	5		8
	F3		2	3	7	8	9	5	6		10
	F4				6	7	5	3	1		7
From Flights	F5				3	1	7	6	8		2
	F6					8	7	5	9		9
	F7					8	15	4	6		20
	F8					10	3	4	7		9
	CI	12	15	17	14	12	11	21	5		
	BC										

This chapter is divided into the following sections: assumptions, definitions, model formulation, preprocessing routines, and the linearized reformulation of the model.

3.1 Assumptions

- Terminal layout remains constant (i.e. no changes).
- Check-In and Baggage Claim are fixed positions.

- No changes can be made to the flight schedule and fleet assignment.
- The flight schedule is sorted by order of departure times, with ties broken using arrival times.
- At least one gate is available upon arrival for every flight.
- Each flight has a 15 minute buffer added onto arrival and departure times.
- Distance from gate g_1 to gate g_2 is the same as from gate g_2 to gate g_1. (The distance from gate g to g is zero.)
- Passengers may transfer from flight b to c at the same gate g, given flight b's departure time occurs before flight c's arrival time.
- Transfer passengers are subject to geographical considerations (i.e. a passenger flying from Miami to Pittsburgh will most likely transfer to a flight departing northbound, like Boston, versus southbound, like Memphis).
- Transfer passenger waiting times do not exceed three hours.
- No transfers exist from:
    1. Check-In to Baggage Claim,
    2. Baggage Claim to Check-In,
    3. Baggage Claim to Flights, and
    4. Flights to Check-In.

3.2 Definitions

The objective is to assign aircraft to gates such that the total *passenger walking distance* is minimized. With each aircraft to gate assignment, a fixed cost is incurred. A prohibitively large neighboring cost can also be used to disallow the assignment of an aircraft to a gate when another conflicting aircraft is assigned to a neighboring gate. (For example, US Air adopts the practice of restricting large size aircraft from parking next to each other.)

Consider the following definitions:

- $m \equiv$ total number of flights $(i = 1, \ldots, m)$
- $n \equiv$ total number of gates $(j = 1, \ldots, n)$
- arrtime_i, $\text{deptime}_i \equiv$ respectively, the arrival and departure times for flight i $(i = 1, \ldots, m)$
- $G_i \equiv$ set of all gates feasible for assigning flight i $(i = 1, \ldots, m)$
- S_{jq}, $q = 1, \ldots, Q_j \equiv$ maximal sets of flights feasible for gate j which overlap with respect to airport occupancy times, so that at most one flight from each such set can be assigned to gate j $(j = 1, \ldots, n)$
- $\delta_{ik} \equiv$ an indicator function $= \begin{cases} 1 & \text{if flights } i \text{ and } k \text{ overlap in airport occupancy time} \\ 0 & \text{otherwise} \end{cases}$
- $p_i^a \equiv$ the number of passengers arriving on flight i and proceeding to baggage-claim $(i = 1, \ldots, m)$

- $p_i^d \equiv$ the number of passengers departing on flight i directly after check-in ($i = 1, \ldots, m$)

- $p_{ik} \equiv$ the number of transfer passengers from flight i to flight k (possibly, $p_{ik} \neq p_{ki}$) ($i, k = 1, \ldots, m$, $i \neq k$)

- $d_j^a \equiv$ walking distance from gate j to the baggage-claim area ($j = 1, \ldots, n$)

- $d_j^d \equiv$ walking distance from the check-in to gate j ($j = 1, \ldots, n$)

- $d_{jl} \equiv$ walking distance for passengers transferring between gates j and l ($d_{jl} = d_{lj}$ for $j \neq l$ and $d_{jl} = 0$ for $j = l$) ($j, l = 1, \ldots, n$)

- $a_{ij} \equiv$ direct cost of assigning flight i to gate j ($i = 1, \ldots, m$, $j = 1, \ldots, n$)

- $b_{ijkl} \equiv$ cost restricting the simultaneous assignment of flight i to gate j and flight k to gate l ($i = 1, \ldots, m-1$, $k = i+1, \ldots, m$, $j, l = 1, \ldots, n$, $j \neq l$). (This cost is typically used to model the physical regulatory pairwise assignment restrictions discussed earlier, by setting it at a relatively large value in such cases, and at zero otherwise.)

- Decision Variables: $x_{ij} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \\ 0 & \text{otherwise} \end{cases}$

  for $i = 1, \ldots, m$, $j = 1, \ldots, n$.

3.3 Model Formulation for the Gate Assignment Problem (GAP)

The objective is to minimize the total walking distance for

- transfer passengers from flight i at gate j to flight k at gate l, $\forall\ (i,j,k,l)$, $j \in G_i$, $l \in G_k$, $i \neq k$, $j \neq l$,
- arriving passengers from flight i at gate j to baggage claim, $\forall\ (i,j)$, $j \in G_i$, and
- departing passengers from check-in to flight i at gate j, $\forall\ (i,j)$, $j \in G_i$,

plus the

- cost of assigning flight i to gate j, $\forall\ (i,j)$, $j \in G_i$, and the
- interaction cost for simultaneously assigning flight i to gate j and flight k to gate l, $\forall\ (i,j,k,l)$, $j \in G_i$, $l \in G_k$, $i \neq k$, $j \neq l$.

The formal gate assignment problem is posed as follows:

$$\begin{aligned}
\text{GAP:}\quad \text{Minimize}\quad f(x) = {} & \sum_{i=1}^{m} \sum_{j \in G_i} [\, a_{ij} + (p_i^a d_j^a + p_i^d d_j^d)\,]\, x_{ij} && (3.1)\\
& + \sum_{i=1}^{m-1} \sum_{j \in G_i} \sum_{k=i+1}^{m} \sum_{\substack{l \in G_k \\ l \neq j \text{ if } \delta_{ik} = 1}} [\, b_{ijkl} \\
& + (p_{ik} + p_{ki}) d_{jl}\,]\, x_{ij} x_{kl}
\end{aligned}$$

$$\begin{aligned}
\text{subject to}\quad & \sum_{j \in G_i} x_{ij} = 1 \quad \forall\ i = 1,\ldots,m && (3.2)\\
& \sum_{i \in S_{jq}} x_{ij} \leq 1 \quad \forall\ q = 1,\ldots,Q_j,\ \ j = 1,\ldots,n && (3.3)\\
& \text{x binary.} && (3.4)
\end{aligned}$$

The objective function (3.1) is comprised of two terms: (i) a linear term representing a fixed assignment cost and a cost of passengers walking between their flights and the check-in and baggage-claim areas, and (ii) a quadratic term representing pairwise assignment costs and the cost of transfer passengers walking between one flight location and another. Note that the cost terms a_{ij} and b_{ijkl} are assumed to be appropriately scaled so as to be commensurate with the walking distances, by dividing them with a factor that reflects the cost per unit distance. Constraints (3.2) are partial assignment constraints, and reflect the restrictions that each aircraft must be assigned to some gate. Constraints (3.3) are set packing types of restrictions that effectively prohibit two overlapping aircraft from being assigned to the same gate concurrently. Before proceeding, let us make some remarks pertaining to the model GAP.

Remark 1: In specifying the sets G_i of admissible gates for each flight i, consideration is given to the size of the aircraft and the volume of passengers being handled by flight i, and whether flight i is an international flight (thereby requiring access to customs facilities), or whether it is restricted to use only designated "market gates" due to the city it is arriving from or departing to. □

Remark 2: For passengers transferring from some flight i to flight k, one may wish to reflect in the model the fact that it is relatively more important to have these flights assigned closer to each other if the time interval between the arrival of flight i and the departure of flight k is smaller, than if this difference is larger, given the same number of interchanging passengers. Accordingly the distance terms in the objective function can be replaced by average required transfer *velocity* terms, with the fixed and quadratic assignment cost terms also being scaled by dividing them with a measure of cost per unit velocity. A suitable weighted average of these two objective functions can also be adopted. □

3.4 Preprocessing Routines for Developing the Model

This section presents routines for preprocessing constraints (3.2) and (3.3). After reading the flight schedule and the passenger distribution, the first pair of routines determine gate feasibilities and reduce the set of feasible assignments, thereby generating the sets G_i $(i = 1, \ldots, m)$. The third routine then generates the sets S_{jq} $(q = 1, \ldots, Q_j)$, $(j = 1, \ldots, n)$. While the size of the problem is reduced, the tightness of the continuous relaxation of the problem is not worsened. For simplicity, one such model can be developed for each "hub" separately.

Flight Information Routine

This routine reads in the flight schedule. The information collected includes the arrival and departure cities, times, and flight numbers, and the aircraft size. Within this routine, a time of 15 minutes is subtracted from the arrival time and is added to the departure time to allow for early arrival, late departure, and push-out durations.

Passenger Distribution Routine

This routine reads in the passenger distribution data. The information collected includes the number of passengers arriving, departing, and transferring between flights.

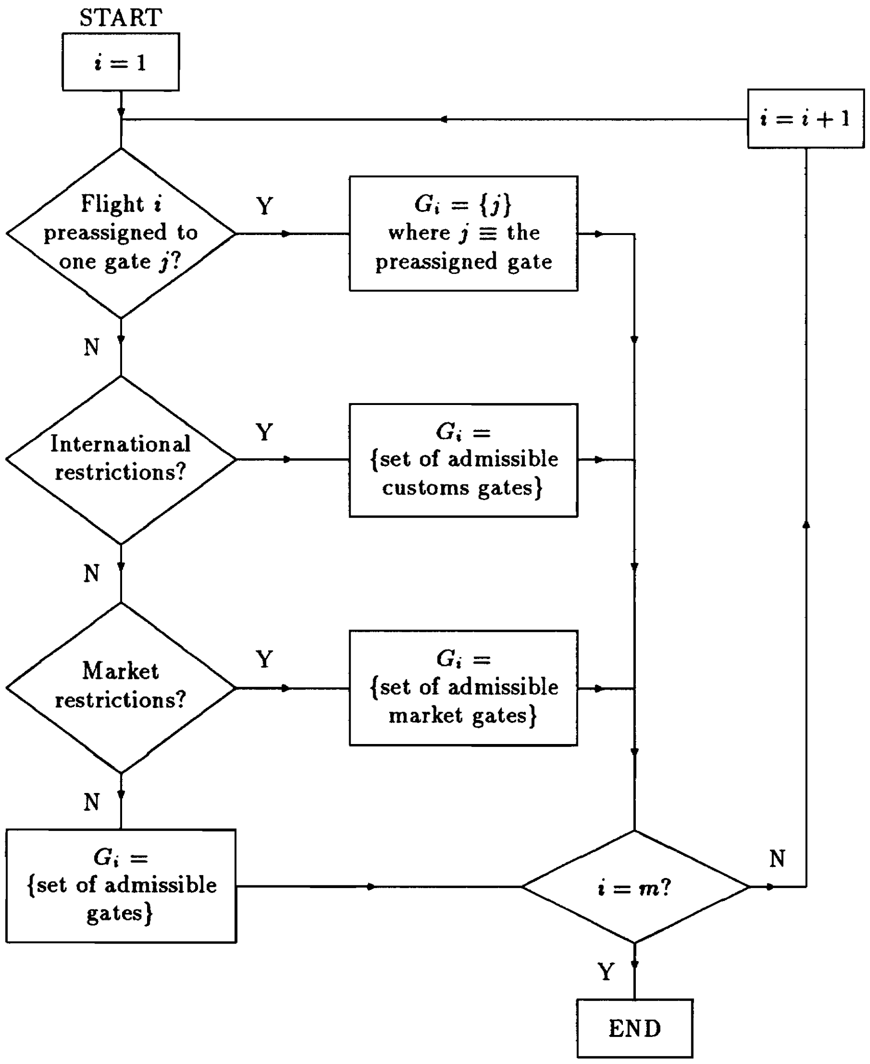

START
$i = 1$
$i = i + 1$
Flight i preassigned to one gate j?
Y
$G_i = \{j\}$ where $j \equiv$ the preassigned gate
N
International restrictions?
Y
$G_i =$ {set of admissible customs gates}
N
Market restrictions?
Y
$G_i =$ {set of admissible market gates}
N
$G_i =$ {set of admissible gates}
$i = m$?
N
Y
END

Figure 3.2: **Gate Feasibility Routine**

Gate Feasibility Routine

Figure 3.2 provides a flow chart for determining the set of feasible gates for each flight. Flights that are preassigned to one gate, or special flights such as international flights and market restriction flights (flights that are arriving from or departing to certain restricted cities) are taken into account separately due to airline managerial procedures. For all other flights, we iterate through the set of gates, and if such a flight i is feasible for a gate j, we set $G_i \leftarrow G_i \bigcup \{j\}$. Table 3.4 presents the set of constraints (3.2) for the example problem that would be generated by utilizing this routine.

Table 3.4: **Constraint (3.2) for the Example Problem**

Flight	Constraint
1	$x_{14} = 1$
2	$x_{21} + x_{22} + x_{23} + x_{24} = 1$
3	$x_{31} + x_{32} + x_{33} + x_{34} = 1$
4	$x_{41} + x_{42} + x_{43} + x_{44} = 1$
5	$x_{54} = 1$
6	$x_{61} + x_{62} + x_{63} + x_{64} = 1$
7	$x_{71} + x_{72} + x_{73} + x_{74} = 1$
8	$x_{81} + x_{82} + x_{83} + x_{84} = 1$

Feasible Assignment Reduction Routine

This routine fixes flight i to gate j if $G_i = \{j\}$ is a singleton, and accordingly eliminates this gate from being assigned to another flight that overlaps with flight i in airport occupancy duration. Successive reductions of this type are conducted until a reduced problem having $| G_i | \geq 2 \ \forall \ i$ results. Figure 3.3 gives a self-explanatory flow

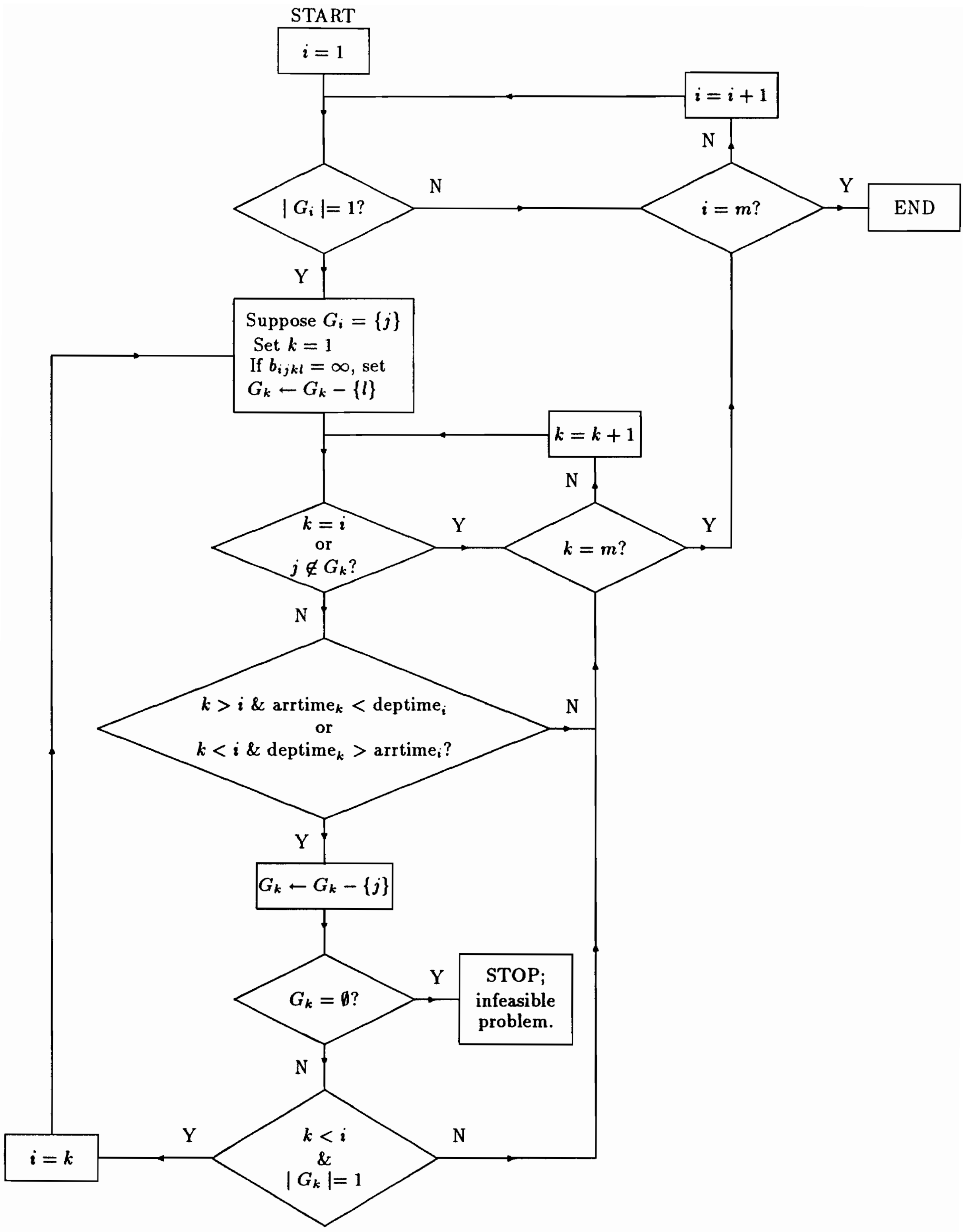

START
$i = 1$
$i = i + 1$
N
$| G_i |= 1?$
N
$i = m?$
Y
END
Y
Suppose $G_i = \{j\}$
Set $k = 1$
If $b_{ijkl} = \infty$, set
$G_k \leftarrow G_k - \{l\}$
$k = k + 1$
N
$k = i$ or $j \notin G_k$?
Y
$k = m?$
Y
N
$k > i$ & $\text{arrtime}_k < \text{deptime}_i$ or $k < i$ & $\text{deptime}_k > \text{arrtime}_i$?
N
Y
$G_k \leftarrow G_k - \{j\}$
$G_k = \emptyset$?
Y
STOP; infeasible problem.
N
$i = k$
Y
$k < i$ & $| G_k |= 1$
N

Figure 3.3: **Feasible Assignment Reduction Routine**

chart for this procedure and Table 3.5 presents feasible assignments for our example problem.

Table 3.5: **Constraint (3.2) Revised for the Example Problem**

Flight	Constraint
1	$x_{14} = 1$
2	$x_{21} + x_{22} + x_{23} = 1$
3	$x_{31} + x_{32} + x_{33} = 1$
4	$x_{41} + x_{42} + x_{43} = 1$
5	$x_{54} = 1$
6	$x_{61} + x_{62} + x_{63} = 1$
7	$x_{71} + x_{72} + x_{73} = 1$
8	$x_{81} + x_{82} + x_{83} = 1$

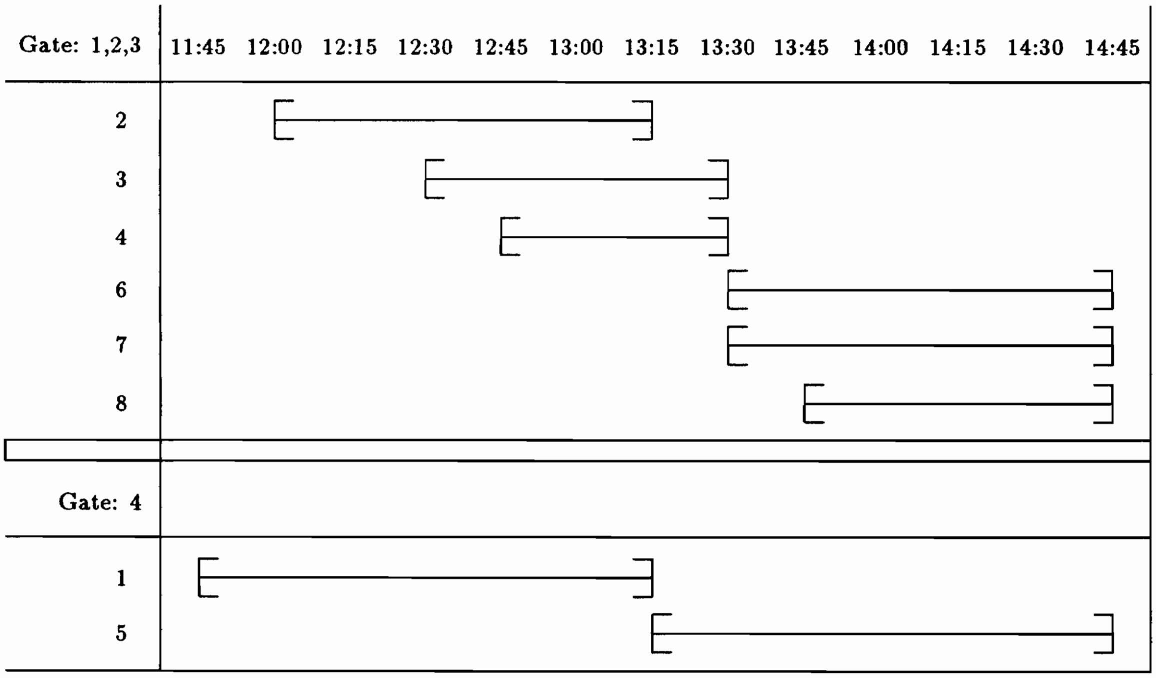
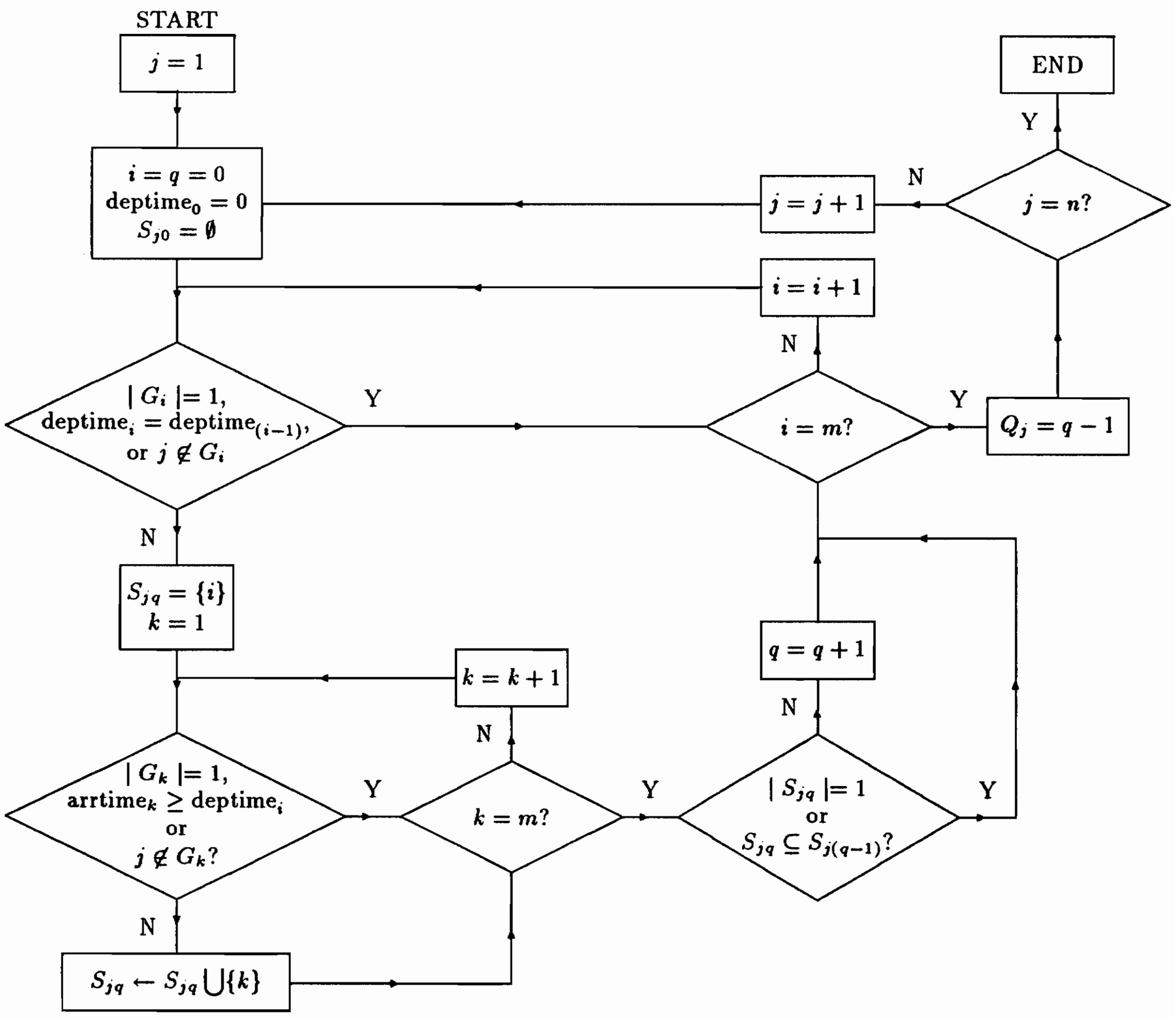


Figure 3.4: **Diagram Displaying the Overlapping Flights By Gate**

Routine for Generating the Sets S_{jq}, $q = 1, \ldots, Q_j$, $j = 1, \ldots, n$

Figure 3.4 illustrates for our example how the flights assignable to each gate overlap in airport occupancy time. Flights 1 and 5 can only be assigned to gate 4, while the rest of the flights are all feasible for assignment to gates 1, 2, and 3. Figure 3.5 provides a flow chart for generating the sets S_{jq}, $q = 1, \ldots, Q_j$ for each gate $j = 1, \ldots, n$. The routine explores each gate separately, examining only the flights feasible for assignment to this particular gate. For $k > i$, due to our ordering of the flights, flight k overlaps flight i when flight k's arrival time is strictly less than flight i's departure time. If this condition is true, flight k is added to S_{jq}. We make two observations. First, the required sets S_{jq} all correspond to overlapping flights i at some time, deptime$_i$, where $j \in G_i$, since for any other time, we can increase the time and still have the same overlapping pattern continue with respect to the next departure time. Hence, this motivates our arranging the flights in increasing order of departure times. Second, if overlapping sets are constructed for each distinct time, deptime$_i$, in this increasing order of departure times, then each set that is not a subset of the previous set gives a new nonredundant constraint. This follows since by the nature of the construction, no set can be a subset of a following set, and any set that is a subset of an earlier set is also a subset of the previous one. Hence, when any overlapping flight set S_{jq} is constructed, it is inspected to verify whether or not it is a subset of the previous set, $S_{j(q-1)}$ for $q > 0$, or if only one flight exists in the set. If this condition is true, then the current set S_{jq} is rejected. Otherwise, it is accepted for constructing a new constraint (3.3). Once a maximal set $(1, \ldots, k)$, say, has been constructed, any new set must contain flight $k+1$ or higher.

Table 3.6 presents the set of constraints (3.3) for the example problem that would be generated by utilizing this routine.



START
$j = 1$
$i = q = 0$
$\text{deptime}_0 = 0$
$S_{j0} = \emptyset$
$j = j + 1$
N
$j = n?$
Y
END
$i = i + 1$
N
$| G_i | = 1$, $\text{deptime}_i = \text{deptime}_{(i-1)}$, or $j \notin G_i$
Y
$i = m?$
Y
$Q_j = q - 1$
N
$S_{jq} = \{i\}$
$k = 1$
$k = k + 1$
N
$q = q + 1$
N
$| G_k | = 1$, $\text{arrtime}_k \geq \text{deptime}_i$ or $j \notin G_k?$
Y
$k = m?$
Y
$| S_{jq} | = 1$ or $S_{jq} \subseteq S_{j(q-1)}?$
Y
N
$S_{jq} \leftarrow S_{jq} \bigcup \{k\}$

Figure 3.5: **Routine for Generating the sets S_{jq}, $q = 1, \ldots, Q_j$ for all gates $j = 1, \ldots, n$.**

Table 3.6: **Constraint (3.3) for the Example Problem**

S_{jq}	Constraint (3.3)
S_{11}	$x_{21} + x_{31} + x_{41} \leq 1$
S_{12}	$x_{61} + x_{71} + x_{81} \leq 1$
S_{21}	$x_{22} + x_{32} + x_{42} \leq 1$
S_{22}	$x_{62} + x_{72} + x_{82} \leq 1$
S_{31}	$x_{23} + x_{33} + x_{43} \leq 1$
S_{32}	$x_{63} + x_{73} + x_{83} \leq 1$

3.5 Linearized Reformulation of the Model

In reformulating the QAP, we transform the quadratic objective function into an equivalent linearized form. We present three particular equivalent, linearized reformulations of the gate assignment problem CAP by suitably adapting the first-order Reformulation-Linearization Technique (RLT) of Sherali and Adams [1989, 1990]. (Also see Adams and Sherali [1986, 1990]). Besides producing a linearization of the objective function, this procedure also introduces new constraints that serve to tighten the continuous relaxation of the underlying linear mixed-integer program, and hence results in an improved representation of the problem.

Toward this end, let us first generate implied nonlinear constraints via the following products:

$$\begin{aligned}
&\text{constraint (3.2)} \ * \ x_{kl}, \quad k > i, \ \ l \in G_k \\
&\text{constraint (3.2)} \ * \ x_{kl}, \quad k < i, \ \ l \in G_k \\
&\text{constraint (3.3)} \ * \ x_{kl}, \quad \forall \ k, \ \ l \in G_k, \ \ l \neq j \text{ if } k \in S_{jq} \\
&\text{constraint (3.3)} \ * \ 1 - x_{kl}, \quad \forall \ k, \ \ l \in G_k, \ \ l \neq j \text{ if } k \in S_{jq}.
\end{aligned}$$

Let us then add these quadratic constraints to the problem GAP and subsequently linearize the entire problem upon using the substitution

$$y_{ijkl} = x_{ij}\, x_{kl} \qquad i = 1,\ldots,m-1,\;\; k = i+1,\ldots,m, \qquad (3.5)$$
$$j,l = 1,\ldots,n,\;\; j \neq l \text{ if } \delta_{ik} = 1.$$

In other words, for the defined combinations of indices, we have,

- $y_{ijkl} = \begin{cases} 1 & \text{if flight } i \text{ is assigned to gate } j \text{ and flight } k \text{ is assigned to gate } l \\ 0 & \text{otherwise.} \end{cases}$

This produces the following 0-1 mixed-integer programming problem MIP1. Note that the constraints (3.3) have been omitted since they are implied by the constraints (3.10) and (3.11), even in the continuous sense, as seen by summing respective pairs of constraints from these sets. Also, note that constraints (3.8)-(3.11) have been generated upon linearizing constraints (3.5)-(3.5), respectively, via the substitution (3.5).

$$\text{MIP1: Minimize } \sum_{i=1}^{m}\sum_{j\in G_i} c_{ij}x_{ij} + \sum_{i=1}^{m-1}\sum_{j\in G_i}\sum_{k=i+1}^{m}\sum_{\substack{l\in G_k \\ l\neq j \text{ if } \delta_{ik}=1}} q_{ijkl}y_{ijkl} \qquad (3.6)$$

$$\text{subject to } \sum_{j\in G_i} x_{ij} = 1 \quad \forall\, i = 1,\ldots,m \qquad (3.7)$$

$$\sum_{\substack{j\in G_i \\ j\neq l \text{ if } \delta_{ik}=1}} y_{ijkl} = x_{kl} \quad \forall\, i,\, k > i,\, l \in G_k \qquad (3.8)$$

$$\sum_{\substack{j\in G_i \\ j\neq l \text{ if } \delta_{ik}=1}} y_{klij} = x_{kl} \quad \forall\, i,\, k < i,\, l \in G_k \qquad (3.9)$$

$$\sum_{\substack{i\in S_{jq} \\ i<k}} y_{ijkl} + \sum_{\substack{i\in S_{jq} \\ i>k}} y_{klij} \leq x_{kl} \qquad (3.10)$$
$$\forall\, j,\, q,\, k,\, l \in G_k,\, l \neq j \text{ if } k \in S_{jq}$$

$$\sum_{i\in S_{jq}} x_{ij} - 1 \leq \sum_{\substack{i\in S_{jq} \\ i<k}} y_{ijkl} + \sum_{\substack{i\in S_{jq} \\ i>k}} y_{klij} - x_{kl} \qquad (3.11)$$
$$\forall\, j,\, q,\, k,\, l \in G_k,\, l \neq j \text{ if } k \in S_{jq}$$

$$\mathbf{x} \text{ binary}, \qquad \mathbf{x}, \mathbf{y} \geq \mathbf{0} \qquad (3.12)$$

$$\text{where} \quad \left.\begin{aligned} c_{ij} &\equiv a_{ij} + (p_i^a d_j^a \ + \ p_i^d d_j^d) \\ q_{ijkl} &\equiv b_{ijkl} \ + \ (p_{ik} \ + \ p_{ki}) d_{jl}. \end{aligned}\right\} \tag{3.13}$$

Remark 3: By Sherali and Lee [1993], it follows that equations (3.7)-(3.9) and (3.12) imply that

$$y_{ijkl} \leq x_{ij}, \quad y_{ijkl} \leq x_{kl} \tag{3.14}$$

$$y_{ijkl} \geq 0, \ \text{ and } \ y_{ijkl} \geq x_{ij} + x_{kl} - 1, \ \ \forall \ i, j, k > i, l \neq j \ \text{ if } \ \delta_{ik} = 1. \tag{3.15}$$

Hence, we have (3.5) holding true for any binary $\mathbf{x}$ feasible to MIP1, therefore validating the equivalence of Problem MIP1 to the original quadratic model (3.1)-(3.4).

Remark 4: In light of Remark 3, note that a more compact equivalent linearization, albeit negligent of the tightness of the resulting relazation, would be one that includes constraints (3.2), (3.3), (3.14), (3.15), and (3.12) along with the objective function (3.6). However, because $q_{ijkl} \geq 0 \ \forall \ (i, j, k, l)$, we can omit constraints (3.14) and have them automatically holding true at optimality because, given that $\mathbf{x}^*$ is part of an optimal solution to this problem, an accompanying set of optimal $\mathbf{y}$ variables is given by $y^*_{ijkl} = \max \ \{0, x^*_{ij} + x^*_{kl} - 1\} \ \forall \ (i, j, k, l)$. This must satisfy (3.14) because if, for example, $x^*_{ij} < y^*_{ijkl} \equiv \max \ \{0, x^*_{ij} + x^*_{kl} - 1\}$, then the maximum must be given by the second term in the maximand as $x^*_{ij} \geq 0$, and this would in turn imply that $x^*_{kl} > 1$, a contradiction. Hence, a more compact, but weaker, linearization of GAP can be derived as the following mixed-integer program MIP0.

$$
\begin{aligned}
\text{MIP0:}\quad \text{Minimize}\quad & \sum_{i=1}^{m}\sum_{j\in G_i} c_{ij}x_{ij} + \sum_{i=1}^{m-1}\sum_{j\in G_i}\sum_{k=i+1}^{m}\sum_{\substack{l\in G_k \\ l\neq j \text{ if } \delta_{ik}=1}} q_{ijkl}y_{ijkl} \\
\text{subject to}\quad & \sum_{j\in G_i} x_{ij} = 1 \quad \forall\ i = 1,\dots,m \\
& \sum_{i\in S_{jq}} x_{ij} \leq 1 \quad \forall\, q = 1,\dots,Q_j,\ j = 1,\dots,n \\
& y_{ijkl} \geq x_{ij} + x_{kl} - 1 \quad \forall\, i,j,k>i, l\neq j \text{ if } \delta_{ik}=1 \\
& \mathbf{x} \text{ binary}, \qquad \mathbf{x},\mathbf{y} \geq \mathbf{0}.
\end{aligned}
$$

Later, we will present some computational comparisons on the relative bounds obtained using MIP0 versus the other proposed linearizations. □

We now present two other relaxed versions of MIP1 that we found to be computationally effective among several others that we investigated. Both these versions are equivalent representations of the original gate assignment problem. In the first of these reformulations, we delete the constraints (3.10) that upper bound the sums of **y** variables (motivated by the fact that $q_{ijkl} \geq 0\ \forall\ (i,j,k,l)$), but now, we add back the original constraints (3.3) that are no longer implied. In the second reformulation, we additionally delete the constraints (3.11). These reformulations are specified below - their equivalence to GAP follows by Remark 3. Note also that if we denote $\nu(\cdot)$ to be the optimal value of any problem $(\cdot)$, and if we denote the continuous relaxation of MIPi by $\overline{\text{MIP}i}$ for $i = 0,\dots,3$, we have,

$$\nu(\text{GAP}) \geq \nu(\overline{\text{MIP1}}) \geq \nu(\overline{\text{MIP2}}) \geq \nu(\overline{\text{MIP3}}) \geq \nu(\overline{\text{MIP0}}) \qquad (3.16)$$

$$\text{MIP2:}\quad \text{Minimize} \sum_{i=1}^{m}\sum_{j\in G_i} c_{ij}x_{ij} + \sum_{i=1}^{m-1}\sum_{j\in G_i}\sum_{k=i+1}^{m}\sum_{\substack{l\in G_k\\ l\neq j \text{ if } \delta_{ik}=1}} q_{ijkl}y_{ijkl}$$

$$\text{subject to}\quad \sum_{j\in G_i} x_{ij} = 1 \quad \forall\ i=1,\dots,m$$

$$\sum_{i\in S_{jq}} x_{ij} \le 1 \quad \forall\ q=1,\dots,Q_j,\ \ j=1,\dots,n$$

$$\sum_{\substack{j\in G_i\\ j\neq l \text{ if } \delta_{ik}=1}} y_{ijkl} = x_{kl} \quad \forall\, i,\, k>i,\, l\in G_k$$

$$\sum_{\substack{j\in G_i\\ j\neq l \text{ if } \delta_{ik}=1}} y_{klij} = x_{kl} \quad \forall\, i,\, k<i,\, l\in G_k$$

$$\sum_{i\in S_{jq}} x_{ij} - 1 \le \sum_{\substack{i\in S_{jq}\\ i<k}} y_{ijkl} + \sum_{\substack{i\in S_{jq}\\ i>k}} y_{klij} - x_{kl}$$

$$\forall\, j,\, q,\, k,\, l\in G_k,\, l\neq j \text{ if } k\in S_{jq}$$

$$\mathbf{x} \text{ binary}, \qquad \mathbf{x}, \mathbf{y} \ge \mathbf{0}.$$

$$\text{MIP3:}\quad \text{Minimize} \sum_{i=1}^{m}\sum_{j\in G_i} c_{ij}x_{ij} + \sum_{i=1}^{m-1}\sum_{j\in G_i}\sum_{k=i+1}^{m}\sum_{\substack{l\in G_k\\ l\neq j \text{ if } \delta_{ik}=1}} q_{ijkl}y_{ijkl}$$

$$\text{subject to}\quad \sum_{j\in G_i} x_{ij} = 1 \quad \forall\ i=1,\dots,m$$

$$\sum_{i\in S_{jq}} x_{ij} \le 1 \quad \forall\ q=1,\dots,Q_j,\ \ j=1,\dots,n$$

$$\sum_{\substack{j\in G_i\\ j\neq l \text{ if } \delta_{ik}=1}} y_{ijkl} = x_{kl} \quad \forall\, i,\, k>i,\, l\in G_k$$

$$\sum_{\substack{j\in G_i\\ j\neq l \text{ if } \delta_{ik}=1}} y_{klij} = x_{kl} \quad \forall\, i,\, k<i,\, l\in G_k$$

$$\mathbf{x} \text{ binary}, \qquad \mathbf{x}, \mathbf{y} \ge \mathbf{0}.$$

Remark 5: Note that due to the combinatorial nature of the (i,k,j,l) combinations, the sizes of the linearized reformulations of GAP presented above can get fairly large as problem size increases. More specifically, denote the number of $\mathbf{x}$ and y variables as, respectively,

$$n_x = \sum_{i=1}^{m} \mid G_i \mid \quad \text{and} \quad n_y = \frac{1}{2}\sum_{i=1}^{m}\{\mid G_i \mid \sum_{k \neq i} \mid G_k \mid\}. \tag{3.17}$$

(Actually, the number of y variables is lesser than n_y because of certain undefined products $x_{ij}x_{kj}$ when $\delta_{ik} = 1$ for $i < k$, where $j \in G_i \cap G_k$, as evident from (3.6).) Also, denote by T_1 the number of constraints in (3.3) and by T_2 the number of constraints in (3.10) or (3.11). Hence, we have,

$$T_1 = \textstyle\sum_{j=1}^{n} Q_j \quad \text{and} \quad T_2 = \displaystyle\sum_{j=1}^{n} [\, Q_j \sum_{k=1}^{m} \{ \mid G_k \mid \, - \, \delta_{jqk} \, \} \,] \tag{3.18}$$

$$\text{where } \delta_{jqk} = \begin{cases} 1 & \text{if } j \in G_k \text{ and } k \in S_{jq} \\ 0 & \text{otherwise} \end{cases} \quad \forall \ (j,q,k).$$

Based on this, we can construct the following table.

Problem	Number of Structural Constraints (M)	Number of Variables (N)
GAP	$m + T_1$	n_x
MIP0	$m + T_1 + n_y$	$n_x + n_y$
MIP1	$m + (m-1)n_x + 2T_2$	$n_x + n_y$
MIP2	$m + T_1 + (m-1)n_x + T_2$	$n_x + n_y$
MIP3	$m + T_1 + (m-1)n_x$	$n_x + n_y$

(3.19)

Because of the potential burden of computing lower bounds or deriving heuristic solutions via $\overline{\text{MIP}}i$ for $i \in \{0,1,2,3\}$ for larger sized problems, we also investigated the use of the following (obvious) lower bounding linear program (LBLP).

$$\text{LBLP:} \quad \text{Minimize} \quad F(x) = \sum_{i=1}^{m}\sum_{j \in G_i} \bar{c}_{ij}x_{ij} \tag{3.20}$$

$$\text{subject to} \quad \sum_{j \in G_i} x_{ij} = 1 \quad \forall \ i = 1,\ldots,m \tag{3.21}$$

$$\sum_{i \in S_{jq}} x_{ij} \leq 1 \quad \forall \ q = 1,\ldots,Q_j, \ \ j = 1,\ldots,n \tag{3.22}$$

$$x_{ij} + x_{kl} \leq 1 \quad \text{if } b_{ijkl} = \infty \ \forall \text{ defined } (i,j,k,l) \tag{3.23}$$

$$\mathbf{x} \geq 0 \tag{3.24}$$

$$\text{where} \quad \bar{c}_{ij} \equiv c_{ij} + \frac{1}{2} \sum_{\substack{k=1 \\ k \neq i}}^{m} \Big[\underset{\substack{l \in G_k \\ l \neq j \text{ if } \delta_{ik}=1}}{\text{minimum}} \{q_{ijkl}\} \Big] \qquad \forall\ j \in G_i, \quad i = 1, \ldots, m. \tag{3.25}$$

Note that LBLP has the same number of variables and constraints as does GAP. Also, note that if some pairwise assignment $x_{ij} = 1$ and $x_{kl} = 1$ has been precluded by taking the corresponding b_{ijkl} to be (virtually) infinitely large, then this is specifically imposed as a constraint (3.23) in LBLP, since this feature would otherwise not be reflected in the revised lower bounding objective function (3.20). We also remark here that because of the nature of the constraints in LBLP, its solution often turns out to be integer valued, hence providing promising feasible (upper bounding) solutions for Problem GAP. □

3.6 Results of the Linearized Reformulations

In order to compare the relative tightness of the lower bounds derived via ν(LBLP) and $\nu(\overline{\text{MIP}i})$, $i = 0, 1, 2, 3$, along with the effort required to solve these corresponding linear programming problems, we generated test problems of various sizes based on realistic data obtained from *USAir*, and ran these problems on a SunSparc IIpx workstation, using CPLEX 2.0 as the LP solver. Table 3.7 summarizes the results obtained. Note that for all these test problems, the linear programming bounding problems $\overline{\text{MIP1}}$ and $\overline{\text{MIP2}}$ both achieve optimal integer solutions. As problem size increases, however, the task of solving these linear programs becomes prohibitively expensive. The linear program $\overline{\text{MIP3}}$ also affords fairly tight lower bounds, but

Table 3.7: **Comparison of various lower bounding linear programming problems**

	# Gates	# Flights	n_x	n_y	Opt value	Bound			Time	
Problem	n	m	(3.17)	(3.17)	(if known)	Problem	Rows	Iters	(secs)	Soln
1	4	8	16	50	190300	MIP1:	200	6	0.08	190300*
						MIP2:	137	6	0.05	190300*
						MIP3:	128	8	0.05	190300*
						MIP0:	65	16	0.22	190300*
						LBLP:	16	6	0.02	189250
2	4	12	31	299	257800	MIP1:	1080	84	2.37	257800*
						MIP2:	675	97	1.93	257800*
						MIP3:	371	23	0.55	257800*
						MIP0:	328	120	1.15	250050
						LBLP:	30	16	0.03	253150
3	4	16	40	859	322100	MIP1:	2118	175	8.47	322100*
						MIP2:	1291	188	6.42	322100*
						MIP3:	643	61	1.32	322100*
						MIP0:	568	259	2.98	313500
						LBLP:	43	20	0.07	309850
4	4	20	50	1480	438800	MIP1:	3580	367	30.38	438800*
						MIP2:	2150	384	22.47	438800*
						MIP3:	1006	71	1.93	438800*
						MIP0:	914	726	12.20	433425
						LBLP:	56	36	0.08	425500
5	4	24	63	2462	578150	MIP1:	5892	1127	171.48	578150*
						MIP2:	3524	1258	136.33	578150*
						MIP3:	1520	77	2.85	578150*
						MIP0:	1548	779	135.28	539575
						LBLP:	71	53	0.13	538800
6	4	32	89	3131	785350	MIP1:	11823	3627	1304.08	785350*
						MIP2:	7026	4534	1160.05	785350*
						MIP3:	2857	163	9.12	785350*
						MIP0:	3224	5424	227.52	735675
						LBLP:	98	72	0.18	711750
7	4	36	99	3890	880050	MIP1:	14494	4960	1791.15	880050*
						MIP2:	8620	6985	1907.35	880050*
						MIP3:	3537	394	31.02	875200
						MIP0:	3993	6453	328.84	810925
						LBLP:	108	81	0.22	790950
8	5	10	28	219	301600	MIP1:	580	76	1.47	301600*
						MIP2:	380	80	1.25	301600*
						MIP3:	273	62	0.73	301600*
						MIP0:	239	125	0.98	267017
						LBLP:	21	8	0.02	258350
9	5	15	51	938	464000	MIP1:	2913	648	57.16	464000*
						MIP2:	1755	1170	82.50	464000*
						MIP3:	757	348	6.90	464000*
						MIP0:	986	598	10.28	365400
						LBLP:	43	20	0.05	379850
10	5	20	67	1689	594500	MIP1:	5547	1410	239.62	594500*
						MIP2:	3295	10671	1194.99	594500*
						MIP3:	1333	907	25.18	594500*
						MIP0:	1748	865	26.32	469000
						LBLP:	60	35	0.10	495050
11	5	25	84	2749	774400	MIP1:	9156	26721	7417.52	774400*
						MIP2:	5392	18521	3969.50	774400*
						MIP3:	2093	1772	66.89	774400*
						MIP0:	2825	3693	20.42	616263
						LBLP:	77	60	0.17	643200
12	6	12	44	648	502100	MIP1:	1884	826	49.08	502100*
						MIP2:	1133	824	35.43	502100*
						MIP3:	521	366	5.47	463500
						MIP0:	681	264	3.42	372725
						LBLP:	37	18	0.05	381900
13	6	18	77	2326	763950	MIP1:	6186	28830	4293.40	763950*
						MIP2:	3657	25344	6366.25	763950*
						MIP3:	1368	2975	86.61	693575
						MIP0:	2381	1138	43.93	512300
						LBLP:	59	31	0.12	541800
14	7	14	64	1500	665000	MIP1:	4084	17271	2496.33	665000*
						MIP2:	2385	12188	1208.75	665000*
						MIP3:	881	1087	28.67	636620
						MIP0:	1545	584	15.62	518536
						LBLP:	49	23	0.07	513750

*: Optimum found by linear programming lower bounding solution itself.

Table 3.8: **Upper and lower bounds of LBLP**

Problem	# Gates n	# Flights m	LB	% of optimal	UB	% over optimal	Opt value (if known)
1	4	8	189250	99.45	190300	0.00	190300
2	4	12	253450	98.31	257800	0.00	257800
3	4	16	308975	96.25	322100	0.00	322100
4	4	20	425150	96.89	438800	0.00	438800
5	4	24	539500	93.31	578150	0.00	578150
6	4	32	710650	90.49	785350	0.00	785350
7	4	36	790500	89.82	880050	0.00	880050
8	5	10	264200	87.60	304750	1.03	301600
9	5	15	388550	83.74	501550	7.49	464000
10	5	20	508600	85.55	594500	0.00	594500
11	5	25	656600	84.79	818500	5.39	774400
12	6	12	376175	74.92	527800	4.87	502100
13	6	18	539525	70.62	813850	6.13	763950
14	7	14	508775	76.51	761250	12.64	665000

this bound begins to deteriorate somewhat significantly below the value $\nu(\overline{\text{MIP1}})$ as problem size increases. In contrast, $\overline{\text{MIP0}}$ and LBLP provide relatively weak lower bounds. However, as problem size increases, it eventually becomes computationally reasonable to solve just LBLP. Table 3.8 provides the lower and upper bounds obtained via LBLP for the same set of 14 test problems used in Table 3.7. Noting the foregoing comment, and the fact that LBLP often returns near optimal integer feasible (*upper bounding*) solutions, we now proceed to prescribe a viable heuristic procedure to solve Problem GAP in Chapter 4.

Chapter IV

Design of a Heuristic Procedure

Motivated by the observations made in Chapter 3, we now present a heuristic procedure to derive a good quality feasible solution to the gate assignment problem GAP. This procedure is essentially a truncated depth-first branch-and-bound scheme that selects a relaxation in the order

$$\{\ \overline{\text{MIP}}1,\ \ \overline{\text{MIP}}2,\ \ \overline{\text{MIP}}3,\ \ \text{LBLP}\ \} \tag{4.1}$$

depending on the first one in this string that might be computationally viable to solve, in order to compute lower and upper bounds. (Note that $\overline{\text{MIP0}}$ is not used here, because as seen from Table 3.7, it provides a bound far weaker than does $\overline{\text{MIP}}3$, and not any quicker.) In this process, the first time when a certain node problem solves a reduced relaxation $\overline{\text{MIP}}i$, for some $i \in \{1, 2, 3\}$ and obtains an integer optimal solution that therefore provides an optimal completion to the current partial solution, the procedure is terminated, and the best solution obtained thus far is taken as the prescribed heuristic solution.

<u>**Remark 6:**</u> Note that one could alternatively choose to continue the branch-and-bound scheme until completion while using a stepped-fathoming strategy in which a node is fathomed whenever its lower bound LB exceeds $(1-\epsilon)$UB for some tolerance $0 < \epsilon < 1$, where UB is the incumbent upper bound. However, if a careful, judicious choice is made in fixing variables, then it is likely that the solution obtained using the prescribed termination criterion will be near optimal. $\square$

4.1 Ranking the Linearized Reformulations

The decision as to which highest (earliest) ranked relaxation in Equation (4.1) is viable to solve is made based on an estimate of solution time predicted via regression equations derived for each problem and based on the size of the overall problem, using the results of Table 3.7. These regression equations are of the form

$$\log(\text{time}) = \beta_0 + \beta_1 \log\left[\left(\frac{\mathrm{M}}{100}\right)^2 \left(\frac{\mathrm{N}}{1000}\right) \rho\right] + \beta_2 n + \beta_3 \log(n), \tag{4.2}$$

where M and N are respectively the number of structural constraints and variables as given by Equation (3.19), n is the number of gates, and ρ is the density of the coefficient matrix of the problem. The values of the regression coefficients β_0, β_1, β_2, and β_3 are specified below, along with the R^2 statistic that measures the goodness of fit. (We remark here that the regression equation for LBLP has been derived only for comparison purposes; this relaxation is solved in any case, if the predicted effort of the preceding relaxations in Equation (4.1) are excessive.)

Besides using Equation (4.2) in selecting a relaxation, in order to ensure that the predicted times are reliable estimates of anticipated effort, a maximum row size is also enforced for each $\overline{\text{MIP}}i$, $i \in \{1,2,3\}$. Specifically, we compute the number of rows created by the free flights for each relaxation $\overline{\text{MIP}}i$, $i \in \{1,2,3\}$ and select that highest ranking relaxation from Equation (4.1) that has a predicted time lesser than a specified maximum (this maximum being taken as ∞ for LBLP) and has a row size lesser than or equal to the maximum specified size given in Equation (4.3).

Problem	β_0	β_1	β_2	β_3	R^2	Maximum Rows
$\overline{\text{MIP1}}$	-1.39	1.00	0.66	-2.80	0.97	3000
$\overline{\text{MIP2}}$	-6.11	1.07	-0.91	15.65	0.98	3500
$\overline{\text{MIP3}}$	-1.33	0.77	0.37	-0.99	0.85	7000
LBLP	-1.01	0.31	0.07	-1.17	0.90	∞

(4.3)

After a solution is constructed using the above truncated branch-and-bound search, this solution can be subjected to an improvement routine. Here, a sequence of partial subproblems of GAP can be investigated, each in turn, holding some of the prescribed flight-to-gate assignments as fixed, and attempting to improve the remaining assignments.

4.2 Heuristic Procedure (HP)

A detailed statement of this overall heuristic procedure is given below. Figure 4.1 provides a flow chart for this routine. Notationally, for any partial solution in which some flights are assigned to certain gates, we denote the corresponding gate in G_i to which flight i has been assigned, for $i \in \{1,\ldots,m\}$, as $j(i)$. Also, given a set of fixed flights $\subseteq \{1,\ldots,m\}$, and considering the remaining flights as "free", the sets G_i are revised to include only the feasible and free assignable gates for each of the free flights i, and the conflict set packing constraints (3.3) are derived based on the free problem as in Figure 3.5. This yields the constraints for the various relaxations in Equation (4.1). Their revised objective functions are given as follows.

LBLP Objective Function:

$$z = \sum_{i \text{ fixed}} c_{ij(i)} + \sum_{i \text{ fixed}} \sum_{\substack{k \text{ fixed} \\ > i \text{ fixed}}} q_{ij(i),kj(k)} + \text{minimum} \left\{ \sum_{i \text{ free}} \sum_{j \in G_i} \bar{c}_{ij} x_{ij} \right\} \tag{4.4}$$

$$\text{where} \quad \bar{c}_{ij} \equiv c_{ij} + \sum_{k \text{ fixed}} q_{ij,kj(k)} + \frac{1}{2} \sum_{\substack{k \text{ free} \\ k \neq i}} [\ \underset{\substack{l \in G_k \\ l \neq j \text{ if } \delta_{ik} = 1}}{\text{minimum}} \{q_{ijkl}\} \quad \forall\, i \text{ free},\ j \in G_i. \tag{4.5}$$

MIPi Objective Function, for $i = 1, 2, 3$:

$$z = \sum_{i \text{ fixed}} c_{ij(i)} + \sum_{i \text{ fixed}} \sum_{\substack{k \text{ fixed} \\ > i \text{ fixed}}} q_{ij(i),kj(k)} + \text{minimum} \left\{ \sum_{i \text{ free}} \sum_{j \in G_i} \hat{c}_{ij} x_{ij} + \sum_{i \text{ free}} \sum_{j \in G_i} \sum_{\substack{k \text{ free} \\ k > i}} \sum_{\substack{l \in G_k \\ l \neq j \text{ if } \delta_{ik} = 1}} q_{ijkl}\, y_{ijkl} \right\} \tag{4.6}$$

$$\text{where} \quad \hat{c}_{ij} \equiv c_{ij} + \sum_{k \text{ fixed}} q_{ij,kj(k)} \quad \forall\, i \text{ free},\ j \in G_i. \tag{4.7}$$

Heuristic Procedure (HP) (See Figure 4.1 for a flow chart):

Step 1 (Initialization):

Solve LBLP given by (3.20)-(3.25). If this problem is infeasible, then so is GAP; stop. Else, let $\bar{\mathbf{x}}$ denote an optimal solution of objective value $\mathrm{F}(\bar{\mathbf{x}})$. If $\bar{\mathbf{x}}$ is integral, then denote $\mathbf{x}^* = \bar{\mathbf{x}}$ as an incumbent solution to GAP with $z^* \equiv f(\mathbf{x}^*)$ as its objective value computed via (3.1). On the other hand, if $\bar{\mathbf{x}}$ is fractional, then attempt to construct a feasible solution by fixing at value 1 the variables for which $\bar{x}_{ij} = 1$ in this solution, and then taking a most fractional variable and making it 1. Accordingly, fix at value 0 any variables appearing with this variable in any of the constraints (3.21)-(3.23). Repeat this until all variables are integer valued. Check if the resulting solution is feasible to the equality constraints (3.21). (It is automatically feasible to the inequality constraints.) If so, then let $\mathbf{x}^*$ be this solution, and set $z^* = f(\mathbf{x}^*)$. If no incumbent solution is available thus far, then set $\mathbf{x}^* = \emptyset$ and $z^* = \infty$. If the lower bound $\mathrm{F}(\bar{\mathbf{x}}) \geq z^*(1-\epsilon)$ for some chosen tolerance $0 < \epsilon < 1$, then terminate with $\mathbf{x}^*$ of objective value z^* as an ϵ-optimal solution. Otherwise, initialize the partial solution list PS $= \emptyset$ (see Geoffrion, 1969), let the current linear program solved $\mathrm{LP}_{\mathbf{current}} \equiv$ LBLP, and set the indicator IND $= 1$. (IND $= 1$ indicates that only a subset of variables that were already 1 in a current linear programming relaxation solution have been fixed at this value, and so, the same relaxation need not be resolved at this point.) Proceed to Step 2 with $\bar{\mathbf{x}}$ being the optimal solution obtained for LBLP.

Step 2 (Selecting a Lower Bounding Relaxation):

Using the regression equations (4.2) and (4.3), select the highest ranked relaxation from (4.1) that has a predicted time $\leq \mathrm{T}_{\max}$ and a row size lesser than or equal to the maximum row size. (We used $\mathrm{T}_{\max} = 600$ seconds in our computations based on a SUN Sparcstation 1000 Workstation using CPLEX 3.0 as the LP solver.) Call this relaxation $\mathrm{LP}_{\mathrm{next}}$. If all predictions exceed $\mathrm{T}_{\max}$ and the maximum number of rows, let $\mathrm{LP}_{\mathrm{next}} \equiv$ LBLP. If IND $= 0$ or if $\mathrm{LP}_{\mathrm{next}} \neq \mathrm{LP}_{\mathrm{current}}$, go to Step 6. Otherwise, proceed to Step 3.

Step 3 (Augment Partial Solution List PS):

Case (i). $\mathrm{LP}_{\mathrm{current}}$ = LBLP: Fix those flights i to gates j for which $\bar{x}_{ij} = 1$ and $\bar{c}_{ij} = \min \{ \bar{c}_{it} : \ t \in G_i \}$, where $\bar{c}$ is defined by (3.25) or (4.5). If no such combinations exist, fix a free flight $\bar{\imath}$ to that gate $\bar{\jmath}$ for which

$$\bar{x}_{\bar{\imath}\bar{\jmath}} = \operatorname{arglexmin} \{ \mid G_i \mid, \bar{c}_{ij} : \ \bar{x}_{ij} = 1 \}. \tag{4.8}$$

(If no $\bar{x}_{ij}$ variable is 1, perform (4.8) over the set of largest valued $\bar{x}_{ij}$ variables.)

Case (ii). $\mathrm{LP}_{\mathrm{current}} = \overline{\mathrm{MIP}i}$, $i \in \{1,2,3\}$: Find the greatest value of $\bar{x}_{ij}$ over all i free, $j \in G_i$. If this value $\bar{x}_{\max}$, say, exceeds 0.5, perform part (a) below. Otherwise, perform part (b).

(a). Define $S = \{ \ i \text{ free: } \bar{x}_{ij} = \bar{x}_{\max} \text{ for some } j \in G_i \ \}$.

$$\text{Select} \quad \bar{\imath} = \operatorname{arglexmin} \{ \mid G_i \mid, \bar{c}_{(i \min)}, -\bar{c}_{i\Delta} : \ i \in S \}, \tag{4.9}$$

$$\text{where} \quad \bar{c}_{(i \min)} = \min \{ \bar{c}_{ij} : \ j \in G_i \} \text{ for } i \text{ free, and} \tag{4.10}$$

$$\bar{c}_{i\Delta} = \text{(absolute) difference between the minimum} \tag{4.11}$$

$$\text{and the next smallest } \bar{c}_{ij} \text{ value over } j \in G_i.$$

Fix flight $\bar{\imath}$ to gate $\bar{\jmath}$ for which $\bar{x}_{\bar{\imath}\bar{\jmath}} = \bar{x}_{\max}$.

(b). Select $\bar{\imath} = \text{arglexmin}\ \{\ |\ G_i\ |, \bar{c}_{(i\ \min)}, -\bar{c}_{i\Delta}\ :\ \ i \text{ free}\ \}$ where $\bar{c}_{(i\ \min)}$ and $\bar{c}_{i\Delta}$ are defined in (4.10) and (4.11), and select

$$\bar{\jmath} = \quad \text{arglexmax}\ \{\ \bar{x}_{\bar{\imath}j}, -\bar{c}_{\bar{\imath}j}, -\ |\ F_j\ |:\ \ j \in G_i\ \}, \qquad (4.12)$$

$$\text{where} \quad |\ F_j\ | = \quad \text{number of flights that are assignable to gate } j. \qquad (4.13)$$

Fix flight $\bar{\imath}$ to gate $\bar{\jmath}$.

In each case above, increment the partial solution list PS by fixing the assignment $x_{\bar{\imath}\bar{\jmath}} = 1$, and set IND $= 0$ if any fractional variables in the solution $\bar{\mathbf{x}}$ have been set at 1. Proceed to Step 4.

Step 4 (Logical Reductions):

Given the current list PS, fix the variables as prescribed in PS and extract all the flights along with their assignable gates. Perform the preprocessing routines illustrated in Figures 3.3 and 3.5 to possibly fix additional variables at 0 or 1, and hence deduce the admissible set of gates G_i for each free flight i along with the resulting conflict constraints (3.3) involving these free flight-gate combinations. If any additional variables are fixed at 0 or 1 using these logical tests, augment PS by these designated fixings in an "underlined" mode as per Geoffrion (1969) (i.e., with an indication that the opposite side of the corresponding branch is fathomed). If these logical tests indicate infeasibility, then proceed to Step 5. Otherwise, return to Step 2 with this reduced gate assignment problem.

Step 5 (Fathom PS):

Fathom PS and backtrack as in Geoffrion's (1969) LIFO scheme. If PS $= \emptyset$, then stop; Problem GAP is infeasible if $\mathbf{x}^* = \emptyset$, and otherwise, $\mathbf{x}^*$ is an ϵ-optimal solution. Else, set IND $= 0$ and return to Step 4.

Step 6 (Solution of Selected Relaxation):

Set $\text{LP}_{\text{current}} = \text{LP}_{\text{next}}$. Formulate and solve this reduced relaxation, using (4.4) and (4.5), or (4.6) and (4.7) appropriately. Let $\bar{\mathbf{x}}$ be the solution thus obtained, with objective value $\bar{z}$ given by (4.4) or (4.6), as the case might be. Using $\bar{\mathbf{x}}$, perform the rounding procedure (if $\bar{\mathbf{x}}$ is nonintegral) as in Step 1 to possibly obtain a new incumbent solution $\mathbf{x}^*$ of objective value z^*.

- If $\text{LP}_{\text{current}} = \overline{\text{MIP}i}$ for some $i \in \{1,2,3\}$, and if $\bar{\mathbf{x}}$ is an all-integer solution, then proceed to Step 7.
- If $\bar{z} \geq z^*(1-\epsilon)$, then go to Step 5.
- Else, if $\text{LP}_{\text{current}} = \text{LBLP}$, then set IND $= 1$ and return to Step 2.
- Else, if $\text{LP}_{\text{current}} = \overline{\text{MIP1}}$, then return to Step 3.
- Else, we have $\text{LP}_{\text{current}} = \overline{\text{MIP}i}$ for some $i \in \{2,3\}$, and in this case, fix all free flights i to free gates j for which $\bar{x}_{ij} = 1$. Increment PS accordingly, set IND $=$ 1, and return to Step 4. If no such $\bar{x}_{ij} = 1$ variables exist, then return to Step 3.

Step 7 (Improvement Phase):

Given an incumbent solution $\mathbf{x}^*$ that assigns flights to gates over a certain time horizon, identify an initial time-window (dependent on the problem data), such that it is computationally feasible to solve $\overline{\text{MIP1}}$ by freeing the flight assignments over this time-window and holding the other assignments fixed. Solve the corresponding reduced problem $\overline{\text{MIP1}}$ defined via (4.6), (4.7), and Step 4 in order to possibly improve upon $\mathbf{x}^*$, using the rounding routine described at Step 1 for this purpose in case the solution to $\overline{\text{MIP1}}$ is fractional. Repeat this procedure for a suitable set of such overlapping time-windows, identified sequentially to cover the entire horizon.

4.3 Results of the Heuristic Procedure

We now present computational results using our heuristic procedure. The test problems used are based on data provided by *USAir*. All computations are performed on a SUN SparcStation 1000 Workstation using CPLEX 3.0 as the LP solver. Tables 4.1 and 4.2 provide a summary of the results. It includes the test problems from Table 3.7 as well as larger test problems.

Table 4.1 returned optimal answers as in Table 3.7 in less time. Due to the small size of these test problems, augmentation was not necessary in fixing flights. Problems 7 and 13 did require use of the heuristic procedure, the MIP3 formulation was run first followed by the MIP2 and MIP1 formulations, in that order. After MIP3 returned a solution, flights were fixed to gates corresponding to variables that returned a value of 1.0. The overall performance was greatly improved.

Table 4.2 reports on experience using larger test problems. For problems 16, 17,

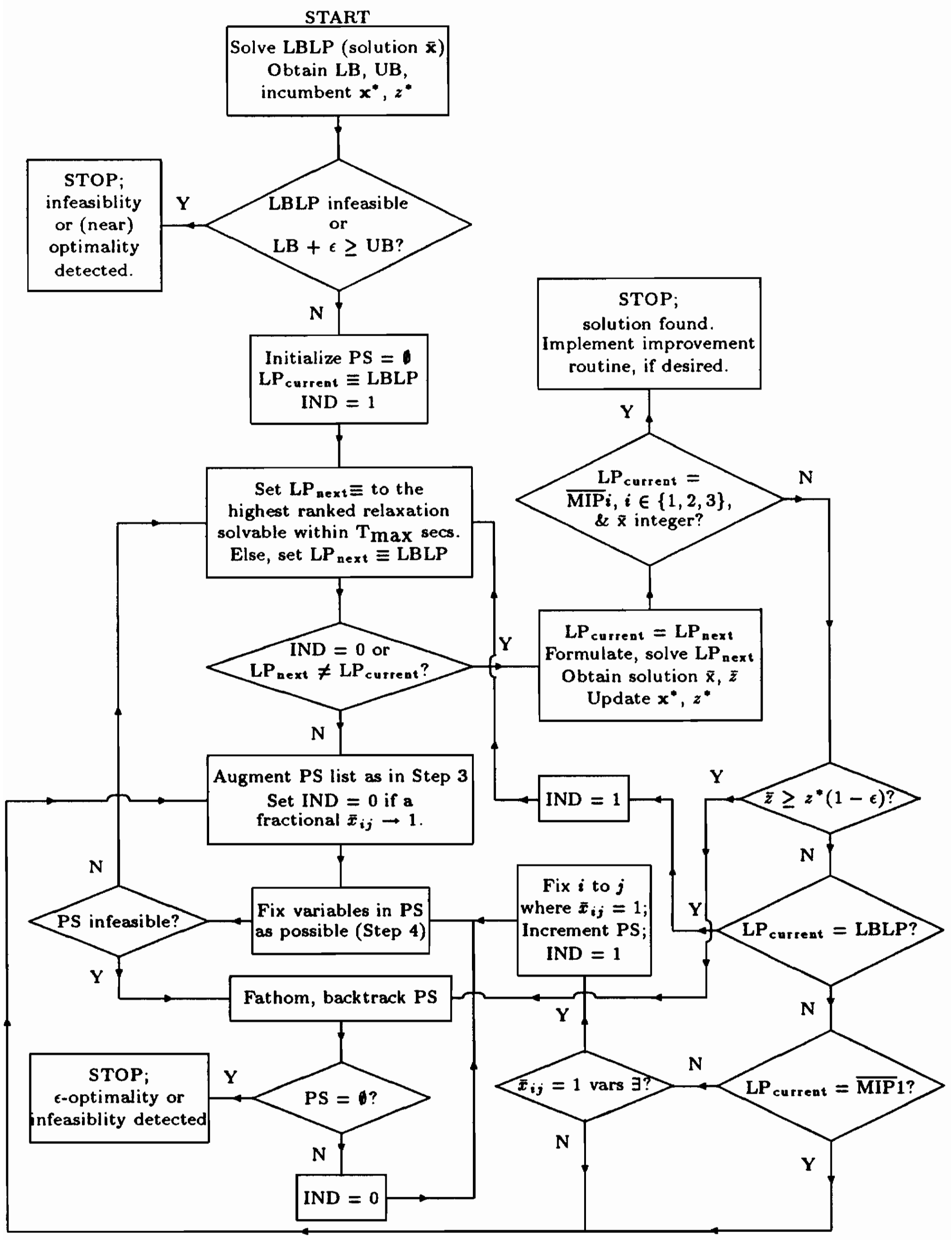

START
Solve LBLP (solution x̄) Obtain LB, UB, incumbent x*, z*
LBLP infeasible or LB + ε ≥ UB?
Y
STOP; infeasiblity or (near) optimality detected.
N
Initialize PS = ∅ LPcurrent ≡ LBLP IND = 1
Set LPnext ≡ to the highest ranked relaxation solvable within Tmax secs. Else, set LPnext ≡ LBLP
IND = 0 or LPnext ≠ LPcurrent?
Y
N
Augment PS list as in Step 3 Set IND = 0 if a fractional x̄ij → 1.
PS infeasible?
N
Y
Fix variables in PS as possible (Step 4)
Fathom, backtrack PS
PS = ∅?
Y
STOP; ε-optimality or infeasiblity detected
N
IND = 0
STOP; solution found. Implement improvement routine, if desired.
Y
LPcurrent = MIPi, i ∈ {1,2,3}, & x̄ integer?
N
LPcurrent = LPnext Formulate, solve LPnext Obtain solution x̄, z̄ Update x*, z*
IND = 1
z̄ ≥ z*(1 − ε)?
Y
N
LPcurrent = LBLP?
Y
N
Fix i to j where x̄ij = 1; Increment PS; IND = 1
Y
x̄ij = 1 vars ∃?
N
N
LPcurrent = MIP1?
Y

Figure 4.1: **Heuristic Procedure Routine**

and 20-24, augmentation was required initially as described in the heuristic to select flights to fix at gates in order to reduce the problem size. Once the problem size was adequate, the problems were solved using various combinations of formulations invoked in the order of Equation (4.1). The improvement routine returned better solutions. In the case of problems 17 and 19 where the improvement routine required more than 1400 CPU seconds, one can reduce the size of the time horizon to speed up this process. Backtracking was not necessary in any of the problems.

It is our observation that the heuristic procedure returns very good answers in a reasonable amount of time. When the improvement routine is used, it usually returns even better results.

One final comment on our computational results. Note that all are test problems were feasible. In case GAP is infeasible, on can attempt to decrease the time buffer added on to the arrival or departure times. A second alternative might be to examine the flights that have an unusually long dwell time at the terminal. These flights can be scheduled to leave the terminal (i.e. pushed away and parked at on off-gate location and then returned to the terminal in enough time for passengers to board). Such considerations need to be incorporated into an operational scheme in practice.

Table 4.1: **Results of the Heuristic Procedure for the Test Problems from Table 3.7**

Problem	1	2	3	4	5	6	7
Gates	4	4	4	4	4	4	4
Flights	8	12	16	20	24	32	36
Initialize: CPU	0.02	0.03	0.02	0.05	0.06	0.07	0.09
LB	189250	253450	308975	425150	539500	710650	790500
UB	190300	257800	322100	438800	578150	785350	880050
	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU
LBLP							
MIP3					1 - 2.73	1 - 13.35	1 - 20.30
MIP2							
MIP1	1 - 0.03	1 - 0.90	1 - 2.81	1 - 8.87			2 - 0.04
Solution	190300^*	257800^*	322100^*	438800^*	578150^*	785350^*	880050^*
Improve: CPU							
Solution							
Final Solution	190300^*	257800^*	322100^*	438800^*	578150^*	785350^*	880050^*
# Flight Fix: LBLP							
MIP3							23
MIP2							
MIP1							
Program CPU	0.11	1.07	2.99	9.25	2.99	13.77	20.94

Problem	8	9	10	11	12	13	14
Gates	5	5	5	5	6	6	7
Flights	10	15	20	25	12	18	14
Initialize: CPU	0.05	0.05	0.08	0.07	0.01	0.08	0.05
LB	264200	388550	508600	656600	376175	539525	508775
UB	304750	501550	594500	818500	527800	813850	761250
	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU
LBLP							
MIP3			1 - 3.06	1 - 13.12		1 - 11.78	
MIP2		1 - 25.71				2 - 326.01	
MIP1	1 - 0.39				1 - 17.68		1 - 404.73
Solution	301600^*	464000^*	594500^*	774400^*	502100^*	763950^*	665000^*
Improve: CPU							
Solution							
Final Solution	301600^*	464000^*	594500^*	774400^*	502100^*	763950^*	665000^*
# Flight Fix: LBLP							
MIP3						2	
MIP2							
MIP1							
Program CPU	0.53	26.01	3.35	13.53	17.85	338.40	405.05

Note:
1. CPU time is in seconds.
2. A blank represents a zero entry or a routine that was not performed.
3. A "$*$" represents the known optimal solution value.
4. Improvement routine not implemented if variables were not augmented.

Table 4.2: **Results of the Heuristic Procedure Using Larger Size Test Problems**

Problem	15	16	17	18	19
Gates	4	5	5	6	6
Flights	33	45	49	54	64
Initialize: CPU	0.07	0.23	0.16	0.71	0.46
LB	434625	1185000	702625	1617800	1155925
UB	494750	1470000	898800	2403800	1612900
	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU
LBLP					
MIP3	1 - 52.03	1 - 435.32	1 - 10.70	1 - 102.27	1 - 7.71
				2 - 58.41	
				3 - 19.47	
				4 - 11.38	
MIP2	2 - 238.67	2 - 31.25		5 - 0.78	
MIP1			2 - 100.48		
Solution	488150*	1470000	874600	2385700	1601450
Improve: CPU			2201.59	102.36	1491.30
Solution			849200	2318600	1562550
Final Solution	488150*	1470000	849200	2318600	1562550
# Flight Fix: LBLP			20	13	30
MIP3	14	22	10	18	
MIP2					
MIP1					
Program CPU	291.47	468.17	2314.42	298.91	1502.43

Problem	20	21	22	23	24
Gates	7	7	8	12	15
Flights	63	79	92	170	187
Initialize: CPU	1.48	1.46	5.26	4.72	4.86
LB	2169250	1615925	781513	2935225	5315075
UB	3572800	2697760	1014925	4002750	6438125
	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU	Run#-CPU
LBLP					
MIP3	1 - 8.44	1 - 181.23	1 - 4.16		
		2 - 126.30			
		3 - 101.69			
		4 - 73.38			
		5 - 53.33			
		6 - 23.33			
		7 - 9.21			
MIP2					
MIP1		8 - 14.04		1 - 2.13	1 - 0.49
Solution	3538700	2664950	998225	3984625	6426175
Improve: CPU	85.46	735.38	64.74	29.93	91.35
Solution	3435550	2646350	982600	3949675	6375525
Final Solution	3435550	2646350	982600	3949675	6375525
# Flight Fix: LBLP	29	36	45	104	127
MIP3		26			
MIP2					
MIP1					
Program CPU	98.78	1330.59	80.79	66.20	126.48

Note:
1. CPU time is in seconds.
2. A blank represents a zero entry or a routine that was not performed.
3. A "*" represents the known optimal solution value.
4. Improvement routine not implemented if variables were not augmented.

Chapter V

Conclusions and Recommendations

In this thesis we have formulated a quadratic partial assignment and packing model for the airline gate assignment problem. Several classes of linearizations aimed at producing tight linear programming relaxations have been designed and tested for this problem. Based on this study, a heuristic procedure has been developed, illustrated, and successfully tested with test problems provided by *USAir*.

This chapter concludes our thesis and presents ideas to improve the performance of the solution approach, discusses alternative applications of the model, and suggests recommendations for future research. The first section will be a revised reduction routine taking advantage of further variable reductions. The second section proposes how to group a series of gates together. Then, applications of the model within other areas of the transportation industry are presented. Finally, we present our recommendations for future research.

5.1 Revised Reduction Routine

Using the feasible assignment reduction routine of Figure 3.3, we present two enhancements. (Other similar enhancements might be possible). The first enhancement takes advantage of two gates that share a push-out tug. The second enhancement takes advantage of a fixed number of overlapping flights, say $a, b, c \in F$, feasible to the same gates, where $| G_a | = | G_b | = | G_c | = | F |$, and the members of G_a, G_b, and G_c are the same gates.

Two Gates Sharing a Push-Out Tug:

Push-out tugs are the vehicles that push back a departing flight from the gate. Once it's pushed back, the tug releases itself from the aircraft and returns to the gate. This process takes about five minutes. Therefore, if a flight is assigned to one of two gates that share a push-out tug, then a flight can be assigned to the other gate only if its departure time is five minutes before or after the first flight's departure time.

Assume flight a has $| G_a |= 2$, and the two gates, $j_1, j_2 \in G_a$ share the same push-out tug. If any flight b overlaps flight a, has its departure time within five minutes of flight a, and is feasible to j_1 and/or j_2, then $G_b \leftarrow G_b - \{j_l\}, \;\; l = 1,2$ if $j_l \in G_b$.

We note here that this enhancement was used in problems 22, 23, and 24 of Table 4.2.

Overlapping Flights Feasible to the Same Number of Gates:

Consider a situation which a given set of flights overlap with each other, and suppose that these flights are feasible to the exact same gates, and number of feasible gates being equal to the number of flights in this set. If there exists another flight that overlaps these flights, then a variable reduction strategy can occur.

For example, assume we have three flights, a, b, and c, feasible to the same three gates, j_1, j_2, and j_3, and these three flights all overlap each other. Clearly, one can see that a, b, and c will be assigned to j_1, j_2, and j_3. The exact assignment will depend on the objective function. For any flight d that overlaps flights a, b, and c and is feasible to j_1, j_2, and/or j_3, we can set $G_d \leftarrow G_d - \{j_l\}, \;\; l = 1,2,3 \;$ if $\; j_l \in G_d$. This process will work for any number of flights, as long as the number of flights is less than the total number of gates at the airport.

5.2 Grouping a Series of Gates Into One Gate

A third enhancement would be to group a series of gates into one fixed gate, such as gates that handle commuter and shuttle flights. As an example, most airports have commuter and shuttle flights designated to a series of gates. These series of gates are doors that are a few feet apart. As the gate agent announces that the flight is boarding, passengers line up at one of the available doors. Then passengers walk or take a bus to the aircraft. Therefore, the walking distance to these gates from other areas of the terminal are the same and the only consideration is how close to assign the regular flights to the commuter or shuttle flights. By grouping a series of gates into one gate, the walking distances can be treated just like check-in and baggage claim. This reduces the number of quadratic variables in MIP1, MIP2, and MIP3 as well as the number of rows in the constraints, thus reducing the overall problem size.

5.3 Applications

There are many applications that mirror the model studied in this thesis. Two direct applications that come to mind are within the transportation industry. The first is to use a similar model to schedule gate agents at an airport in order to minimize their walking distances from gate to gate. Once the flights have been assigned to their gates, agents need to be scheduled to help with the passenger demand. Once the flight departs and a long time period exists before the next flight arrives, these agents could perform their duties at another gate. By minimizing total walking distances, this can improve productivity/convenience, and perhaps, reduce the total number of gate agents needed. A second application might be to use a model of this type to assign trailers to docking ramps for a Less-than-Truckload carrier. As trailers are docked, they are unloaded, the cargo is sorted, then loaded onto another trailer. A model could be formulated to schedule the movements of trailers so as to minimize the total distance that the cargo must travel.

5.4 Recommendations for Future Research

As with any model formulation, there is always room for further enhancements. We have recommended two of them in Section 5.1 above. Another would be to consider Remark 2 from Chapter 3 and test the strategy of assigning flights to be closer together that have a smaller time interval between the arrival of one flight and the departure of the next flight. *USAir* naturally has a policy of leaving adequate time for passengers to transfer between flights. This time duration varies depending on the type of flight transfers (i.e. regular to commuter), but the minimum time is usually 25 minutes. By applying the aforementioned strategy in conjunction with current policies, passengers having a small time duration to transfer between flights would not have to walk large distances. This would also enable an improved contingency in case of late arrivals. Finally, one can study the above mentioned applications (Section 5.2), reformulate the model to represent the particular problems, and design and test similar solution procedures for these cases.

Chapter VI

References

ADAMS, W.P., AND H.D. SHERALI. 1986. "A Tight Linearization and an Algorithm for Zero-One Quadratic Programming Problems," *Management Science* **32**, 1274-1290.

ADAMS, W.P., AND H.D. SHERALI. 1990. "Linearization Strategies for a Class of Zero-One Mixed Integer Programming Problems," *Operations Research* **38**, 217-225.

ASSAD, A., AND W. XU. 1992. "The Quadratic Minimum Spanning Tree Problem," *Naval Research Logistics Quarterly* **39**, 339-417.

BABIĆ, O., D. TEODOROVIĆ, AND V. TOŠIĆ, A. M. ASCE. 1984. "Aircraft Stand Assignment to Minimize Walking," *Journal of Transportation Engineering* **110**, 55-66.

BAZARAA, M.S., J.J. JARVIS, AND H.D. SHERALI. 1990. *Linear Programming and Network Flows, Second Edition.* John-Wiley and Sons.

BAZARAA, M.S., AND A.N. ELSHAFEI. 1979. "An Exact Branch-and-Bound Procedure for the Quadratic Assignment Problem," *Naval Research Logistics Quarterly* **26**, 109-121.

BAZARAA, M.S., AND O. KIRCA. 1983. "A Branch-and-Bound Heuristic for Solving the Quadratic Assignment Problem," *Naval Research Logistics Quarterly* **30**, 287-304.

BAZARAA, M.S., AND H.D. SHERALI. 1980. "Benders' Partitioning Scheme Applied to a New Formulation of the Quadratic Assignment Problem," *Naval Research Logistics Quarterly* **27**, 29-41.

BAZARAA, M.S., AND H.D. SHERALI. 1982. "On the Use of Exact and Heuristic Cutting Plane Methods for the Quadratic Assignment Problem," *Journal of the Operational Research Society* **33**, 991-1003.

BEALE, E.M.L., AND J.A. TOMLIN. 1972. "An Integer Programming Approach to a Class of Combinatorial Problems," *Mathematical Programming* **3**, 339-344.

BIHR, R.A. 1990. "A Conceptual Solution to the Aircraft Gate Assignment Problem Using 0,1 Linear Programming," *Proceedings of the 12th Annual Conference on Computers and Industrial Engineering* **19**, 280-284.

BONOMI, E., AND J. LUTTON. 1986. "The Asymptotic Behaviour of a Quadratic Sum Assignment Problems: A Statistical Mechanics Approach," *European Journal of Operational Research* **26**, 295-300.

BRAAKSMA, J.P. 1977. "Reducing Walking Distance at Existing Airports," *Airport Forum* **4**, 135-142.

BRUJIS, P.A. 1984. "On the Quality of Heuristic to a 19x19 Quadratic Assignment Problem," *European Journal of Operational Research* **17**, 21-30.

BURKARD, R.E. 1984. "Quadratic Assignment Problems," *European Journal of Operational Research* **15**, 283-289.

BURKARD, R.E. 1990. "Locations wit Spatial Interactions: The Quadratic Assignment Problem," in P.B. Mirchandani and R.L. Francis (eds.), *Discrete Location Theroy*, John-Wiley and Sons, 387-437.

BURKARD, R.E., AND T. BÖNNIGER. 1983. "A Heuristic for Quadratic Boolean Programs with Applications to Quadratic Assignment Problems," *European Journal of Operational Research* **13**, 374-386.

BURKARD, R.E., AND U. DERIGS. 1980. "Assignment and Matching Problems: Solution Methods with FORTRAN-Programs," *Lecture Notes in Economics and*

Mathematical Systems 184, M. Beckman and H.P. Kunzi (ed.), Springer-Veriag Publishing Company, Berlin, 99-148.

BURKARD, R.E., AND U. FINCKE. 1983. "The Asympototic Probabilistic Behaviour of Quadratic Sum Assignment Problems," *Zeitschrift für Operations Research* **27**, 73-81.

BURKARD, R.E., AND F. RENDL. 1984. "A Thermodynamically Motivated Simulation Procedure for Combinatorial Optimization Problems," *European Journal of Operational Research* **17**, 169-174.

BURKARD, R.E., AND K. STRATMANN. 1978. "Numerical Investigations on Quadratic Problems," *Naval Research Logistics Quarterly* **25**, 129-148.

CARRARESI, P., AND F. MALUCELLI. 1992a. "A New Lower Bound for the Quadratic Assignment Problem," *Operations Research* **40**, S22-S27.

CARRARESI, P., AND F. MALUCELLI. 1992b. "A Reformulation Scheme and New Lower Bounds for the Quadratic Assignment Problem," Technical Report TR-34/92, Dipartimento Di Informatica, Università Degli Studi Di Pisa.

CHAKRAPANI, J., AND J. SKORIN-KAPOV. 1992a. "A Connectionist Approach to the Quadratic Assignment Problem," *Computers and Operations Research* **19**, 287-295.

CHAKRAPANI, J., AND J. SKORIN-KAPOV. 1992b. "Massively Parallel Tabu Search for the Quadratic Assignment Problem," Harriman School Working Paper (HAR-91-06), State University of New York, Stony Brook, New York. (To appear in *Annals of Operations Research*).

CHAKRAPANI, J., AND J. SKORIN-KAPOV. 1993. "A Constructive Method Improve Lower Bounds for the Quadratic Assignment Problem," Harriman School Working Paper, State University of New York, Stony Brook, New York.

CHRISTOFIDES, N., AND E. BENAVENT. 1988. "An Exact Algorithm for the Quadratic Assignment Problem on a Tree," *Operations Research* **37**, 760-768.

CHRISTOFIDES, N., AND M. GERRARD. 1981. "A Graph Theoretic Analysis of Bounds for the Quadratic Assignment Problem," in P. Hansen (ed), *Studies on*

Graphs and Discrete Programming, Annals of Discrete Mathematics **11**, North-Holland, 61-68.

CHRISTOFIDES, N., A. MINGOZZI, AND P. TOTH. 1980. "Contributions to the Quadratic Assignment Problem," *European Journal of Operational Research* **4**, 243-247.

COHOON, J.P., S.U. HEGDE, W.N. WORTHY, AND D.S. RICHARDS. 1991. "Distributed Genetic Algorithms for the Floorplan Design Problem," *IEEE Transactions on Computer-Aided Design* **10** 483-491.

CONNOLLY, D.T. 1990. "An Improved Annealing Scheme for the QAP," *European Journal of Operational Research* **46**, 93-100.

EDWARDS, C.S. 1980. "A Branch-and-Bound Algorithm for the Koopmans-Beckmann Quadratic Assignment Problem," *Mathematical Programming Study* **13**, 35-52.

ELSHAFEI, A.N. 1977. "Hospital Layout as a Quadratic Assignment Problem," *Operational Research Quarterly* **28**, 167-179.

FINKE, G., R.E. BURKARD, AND F. RENDL. 1987. "Quadratic Assignment Problems," in S. Martello, G. LaPorte, M. Minoux, and C. Ribeiro (eds.), *Surveys in Combinatorial Optimization*, Annals of Discrete Mathematics **31**, North-Holland, 61-82.

FLOOD, M.M. 1990. "Exact and Heuristic Algorithms for the Weighted Feedback Arc Set Problem: A Special Case of the Skew-Symmetric Quadratic Assignment Problem," *Networks* **20**, 1-23.

FOULDS, L.R., AND D.F. ROBINSON. 1976. "A Strategy for Solving the Plant Layout Problem," *Operational Research Quarterly* **27**, 845-855.

FRANCIS, R.L., L.F. MCGINNIS, JR., AND J.A. WHITE. 1992. *Facility Layout and Location: An Analytical Approach, Second Edition.* Prentice Hall.

FRIEZE, A.M., AND J. YADEGAR. 1983. "On the Quadratic Assignment Problem," *Discrete Applied Mathematics* **5**, 89-98.

GASCHÜTZ, G.K., AND J.H. AHRENS. 1968. "Suboptimal Algorithms for the Quadratic Assignment Problem," *Naval Research Logistics Quarterly* **15**, 49-62.

GAVETT, J.W., AND N.V. PLYTER. 1966. "The Optimal Assignment of Facilities to Locations by Branch-and-Bound," *Operations Research* **14**, 210-232.

GEOFFRION, A.M. 1969. "An Improved Enumeration Approach for Integer Programming pproach," *Operations Research* **17**, 437- 454.

GEOFFRION, A.M. AND G.W. GRAVES. 1976. "Scheduling Parallel Production Lines with Changeover Costs: Practical Application of a Quadratic Assignment/LP Approach," *Operations Research* **24**, 595-610.

GILMORE, P.C. 1962. "Optimal and Suboptimal Algorithms for the Quadratic Assignment Problem," *Journal of the Society of Industrial and Applied Mathematics* **2**, 305-313.

GLOVER, F. 1989. "Tabu Search - Part I," *ORSA Journal on Computing* **1**, 190-206.

GRAVES, G.W., AND A.B. WHINSTON. 1970. "An Algorithm for the Quadratic Assignment Problem," *Management Science* **17**, 453-471.

HADLEY, S.W., F. RENDL, AND H. WOLKOWICZ 1992a. "A New Lower Bound Via Projection for the Quadratic Assignment Problems." *Mathematics of Operations Research* **17** 727-739.

HADLEY, S.W., F. RENDL, AND H. WOLKOWICZ 1992b. "Symmetrization of Nonsymmetric Quadratic Assignment Problems and the Hoffman-Wielandt Inequality." *Linear Algebra and its Applications* **167** 53-64.

HANAN, M., AND J.M. KURTZBERG 1972. "A Review of the Placement and Quadratic Assignment Problems." *SIAM Review* **14** 324-342.

HEIDER, C.H. 1973. "An N-Step, 2-Variable search Algorithm for the Component Placement Problem," *Naval Research Quarterly* **20**, 699-724.

HERAGU S.S., AND A. KUSIAK. 1988. "Machine Layout Problem In Flexible Manufacturing Systems," *Operations Research* **36**, 258-268.

HILLIER, F.S., AND M.M. CONNORS. 1966. "Quadratic Assignment Problem Algorithms and the Location of Indivisible Facilities," *Management Science* **13**, 42-57.

HUNTLEY, C.L., AND D.E. BROWN. 1991. "A Parallel Heuristic for Quadratic Assignment Problems," *Computers and Operations Research* **18**, 275-289.

JOHNSTON, T. 1992. "New Linear Programming-Based Solution Procedures for Quadratic Assignment Problem," *Doctoral Dissertation, Clemson University, Clemson, South Carolina.*

KAUKU, B.K., AND G.L. THOMPSON. 1986. "An Exact Algorithm for the General Quadratic Assignment Problem," *European Journal of Operational Research* **23**,
382-390.

KAUKU, B.K., G.L. THOMPSON, AND T.E. MORTON. 1991. "A Hybrid Heuristic for the Facilities Layout Problem," *Computers and Operations Research* **18**, 241-253.

KAUFMAN, L., AND F. BROECKX. 1978. "An Algorithm for the Quadratic Assignment Using Problem Bender's Decomposition," *European Journal of Operational Research* **2**, 207-211.

KOOPMANS, T.C., AND M. BECKMAN. 1957, "Assignment Problems and the Location of Economic Activities," *Econometrica* **25**, 53-76.

KUSIAK, A., AND S. HERAGU. 1987, "The Facility Layout Problem," *European Journal of Operational Research* **29**, 229-251.

LAPORTE, G, AND H. MERCURE. 1988, "Balancing Hydraulic Turbine Runners: A Quadratic Assignment Problem," *European Journal of Operational Research* **35**, 378-381.

LASHKARI, R.S., AND S.C. JAISINGH. 1980, "A Heuristic Approach to Quadratic Assignment Problem," *Journal of the Operational Research Society* **31**, 845-850.

LAWLER, E.L. 1963. "The Quadratic Assignment Problem," *Management Science* **19**, 586-599.

LIGGETT, R.S. 1981. “The Quadratic Assignment Problem: An Experimental Evaluation of Solution Strategies,” *Management Science* **27**, 442-458.

LOVE, R.F. AND J.Y. WONG. 1976. “Solving Quadratic Assignment Problems with Rectangular Distances and Integer Programming,” *Naval Research Logistics Quarterly* **23**, 623-627.

MANGOUBI, R.S., AND D.F.X. MATHAISEL. 1985. “Optimizing Gate Assignments at Airport Terminals,” *Transportation Science* **19**, 173-188.

METROPOLIS, N., A. ROSENBLUTH, M. ROSENBLUTH, A. TELLER, AND E. TELLER. 1953. *Journal of Chemical Physics* **21**, 1087-1092.

MURTAGH, B.A., T.R. JEFFERSON, AND V. SORNPRASIT. 1982. “A Heuristic Procedure for Solving the Quadratic Assignment Problem,” *European Journal of*
Operational Research **9**, 71-76.

NUGENT, C.E., T.E. VOLLMANN, AND J. RUML. 1968. “An Experimental Comparison of Techniques for the Assignment of Facilities to Locations,” *Operations Research* **16**, 150-173.

PARKER, C.S. 1976. “An Experimental Investigation of Some Heuristic Strategies for Component Placement,” *Operational Research Quarterly* **27**, 71-81.

PICONE, C.J., AND W.E. WILHELM. 1984. “A Perturbation Scheme To Improve Hillier’s Solution to the Facilities Layout Problem,” *Management Science* **30**, 1238-1249.

PIERCE, J.F., AND W.B. CROWSTON. 1971. “Tree-Search Algorithms for Quadratic Assignment Problems,” *Naval Research Logistics Quarterly* **18**, 1-36.

PIERSKALLA, W.P. 1968. “The Multidimensional Assignment Problem,” *Operations*
Research **16**, 422-430.

RAJGOPAL, P. 1985. ”A Flexible and Improvement Heuristic for the Quadratic Assignment Problem,” *Masters Thesis, Virginia Polytechnic Institute and State University.*

RENDL, F. 1985. "Ranking Scalar Products to Improve Bounds for the Quadratic Assignment Problem," *European Journal of Operational Research* **20**, 363-372.

RENDL, F. AND H. WOLKOWICZ 1992. "Applications of Parametric Programming and Eigenvalue Maximization to the Quadratic Assignment Problem," *Mathematical Programming* **53**, 63-78.

ROUCAIROL, C. 1987. "A Parallel Branch-and-Bound Algorithm for the Quadratic Assignment Problem," *Discrete Applied Mathematics* **18**, 211-225.

SAHNI, S. AND T. GONZALEZ. 1976. "Journal of the Association for Computing Machinery," *Journal of the Association for Computing Machinery* **23**, 555-565.

SCHRÖDER, H. 1972. "Assignment of Aircraft to Gate Positions," *AGIFORS Symposium* **12**, 301-318.

SHARPE, R., AND J. BROTCHIE. 1972. "An Urban Systems Study," *Royal Australian Planning Institute Journal* **10**, 3-12.

SHERALI, H.D. 1979. "The Quadratic Assignment Problem: Exact and Heuristic Methods," *Doctoral Dissertation, Georgia Institute of Technology, Atlanta, Georgia.*

SHERALI, H.D. AND W.P. ADAMS. 1994. "A Hierarchy of Relaxations and Convex Hull Characterizations for Mixed-Integer Zero-One Programming Problems," *Discrete Applied Mathematics* **52**, 83- 106.

SHERALI, H.D. AND W.P. ADAMS. 1990. "A Hierarchy of Relaxations Between the Continuous and Convex Hull Representations for Zero-One Programming Problems," *SIAM Journal on Discrete Mathematics* **3**, 411-430.

SHERALI, H.D. AND Y. LEE. 1993. "Tighter Representation for Set Partitioning Problems Via a Reformulation-Linearization Approach," Research Report, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061-0118. (To appear in *Discrete Applied Mathematice*).

SHERALI, H.D. AND P. RAJGOPAL. 1986. "A Flexible, Polynomial-Time Construction and Improvement Heuristic for the Quadratic Assignment Problem," *Computers and Operations Research* **13**, 587-600.

SKORIN-KAPOV, J. 1990. "Tabu Search Applied to the Quadratic Assignment Problem," *ORSA Journal on Computing* **2**, 33-45.

SKORIN-KAPOV, J. 1991. "Extensions of a Tabu Search Adaptation to the Quadratic Assignment Problem," Harriman School Working Paper (HAR-90-006), State University of New York, Stony Brook, New York. (To appear in *Computers and Operations Research*).

STEINBERG, L. (1961). "The Backboard Wiring Problem: A Placement Algorithm," *SIAM Review* **3**, 37-50.

TAILLARD, E. (1991). "Robust Taboo Search for the Quadratic Assignment Problem," *Parallel Computing* **17**, 443-455.

TATE, D.M., AND A.E. SMITH. 1992. "A Genetic Approach to the Quadratic Assignment Problem," (To appear in *Computers and Operations Research*).

VANDERSTRAETEN, G., AND M. BERGERON. 1988. "Automatic Assignment of Aircraft to Gates at a Terminal," *Computers and Industrial Engineering* **14 (1)**, 15-25.

WEST, D.H. 1983. "Algorithm 608: Approximate Solution of the Quadratic Assignment Problem," *ACM Transactions on Mathematical Software* **9**, 461-466.

WILHELM, M.R., AND T.L. WARD. 1987. "Solving Quadratic Assignment Problems by 'Simulated Annealing'," *IIE Transactions* **19**, 107-119.

Vita

Eric was born on February 1, 1967, in Las Vegas, Nevada. As an Air Force brat, he lived in Hawaii, the Philippines, and Virginia.

Eric received his B.S. in Industrial Engineering and Operations Research, May, 1989, from Virginia Polytechnic Institute and State University. After receiving his degree, he spent two years working for a retail company in Washington, D.C., as an Industrial Engineer in their distribution center. Fueled with the desire to build decision support models in the transportation industry, he returned to VPI & SU in August, 1991, to earn an M.S. in Industrial and Systems Engineering.

In July, 1993, Eric accepted a job at Burlington Northern Railroad, Ft. Worth, Texas, designing and building decision support models. He has spent the last two years completing his thesis and in the process, he and his advisor published a paper on this thesis.

Eric enjoys being with his family, camping, biking, traveling, history, studying the Bible, and plays the piano and violin. He has been married for four years to his wife, Natalie. They recently had a daughter, Lauren Grace, this past January 3.