PHARMACY DUTY SCHEDULING PROBLEM WITH AN APPLICATION TO İZMİR

A THESIS

SUBMITTED TO

THE GRADUATE SCHOOL OF SOCIAL SCIENCES

OF

IZMIR UNIVERSITY OF ECONOMICS

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF ARTS

IN

THE GRADUATE SCHOOL OF SOCIAL SCIENCES

By

Ayşe Ebru Ağlamaz

July 2011

Approval of the Graduate School of Social Sciences:

Prof. Dr. Cengiz Erol
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Arts.

Asst. Prof. Dr. Muhittin Hakan Demir
Head of the Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Arts.

Asst. Prof. Dr. Özgür Özpeynirci
Supervisor

Examining Committee Members

Asst. Prof. Dr. Muhittin Hakan Demir _____

Asst. Prof. Dr. Özgür Özpeynirci _____

Asst. Prof. Dr. Zeynep Sargut _____

ABSTRACT

PHARMACY DUTY SCHEDULING PROBLEM WITH AN APPLICATION TO İZMİR

Ayşe Ebru Ağlamaz
M.A.in Logistics Management
Supervisor: Asst. Prof. Dr. Özgür Özpeynirci
July 2011, 125 pages

In this thesis, we define the pharmacy duty scheduling (PDS) problem, where a subset of pharmacies should be on duty on national holidays, at weekends and at nights in order to be able to satisfy the emergency drug needs of the society. We model the PDS problem as a multi-period facility location problem with special side constraints and analyze the computational complexity. We develop a Tabu Search algorithm to obtain feasible upper bounds for PDS problem and three lower bounds including Lagrangian Relaxation. We test the performance of mathematical models, Tabu Search and lower bounds on randomly generated instances. We analyze the current system in İzmir and obtain the real data with the help of a Geographic Information System specifically developed for this thesis. Our results show that proposed Tabu Search algorithm makes significant improvements over the current system.

Keywords: Facility location, Tabu search, Lagrangian relaxation, Duty scheduling.

ÖZET

ECZANE NÖBET ÇİZELGELEME PROBLEMİ: İZMİR UYGULAMASI

Ayşe Ebru Ağlamaz
Lojistik Yönetimi, M.A.
Tez Yöneticisi: Yard. Doç. Dr. Özgür Özpeynirci
Temmuz 2011, 125 sayfa

Bu tezde, eczanelerin bir alt kümesinin ulusal tatiller, hafta sonları ve hafta içi akşamları halkın acil ilaç ihtiyaçlarını karşılayabilmek için nöbetçi olması gerektiği, eczane nöbet çizelgeleme (ENÇ) problemini tanımladık. ENÇ problemini özel ek kısıtları olan çok dönemli bir tesis yerleşimi problemi olarak modelledik ve hesaplama karmaşıklığını inceledik. ENÇ problemine olurlu üst sınırlar elde etmek için tabu arama algoritması ve Lagrange gevşetmesi de dahil olmak üzere üç alt sınır geliştirdik. Matematik modellerin, tabu aramanın ve alt sınırların performanslarını rassal üretilmiş örneklerde test ettik. İzmir'deki mevcut sistemi inceledik ve bu tez için özel olarak geliştirilmiş coğrafi bilgi sistemi yardımı ile gerçek verileri elde ettik. Sonuçlarımız önerilen tabu arama algoritmasının mevcut sistem üzerinde önemli iyileştirmeler yaptığını göstermektedir.

Anahtar sözcükler: Tesis yerleşimi, Tabu arama, Lagrange gevşetme, Nöbet çizelgeleme.

To the most important four people in my life...

To my mom

To my dad

To my grandma

and

To my other half of the apple

Acknowledgement

I owe my deepest gratitude to my advisor Asst. Prof. Dr. Özgür Özpeynirci. This thesis would not be possible without his patience, support and guidance. It was a real honor working with him and it will never be enough how much I thank him for all those hours studying with me, trying to teach me everything we needed for this thesis and sometimes more than we needed. I have really learned a lot from him not only academically but also personally.

I would like to thank the committee members Asst. Prof. Dr. Muhittin Hakan Demir and Asst. Prof. Dr. Zeynep Sargut for their valuable guidance and comments. I also would like to thank all faculty members of Department of Logistics Management for their support and understanding throughout this thesis. I would like to specially thank Asst. Prof. Dr. Ahmet Camcı for helping me to get through LaTeX. I would like to thank Pharmacist Mehlika Çubukçu and members of the 3^{rd} Regional Chamber of Pharmacists' for their cooperation and the information they shared with us. I am indebted to Merve Alanyalı for her irreplaceable support while developing the GIS application. I would like to thank all my friends and colleagues for supporting me in many ways and for their patience even when I was too much crabby.

I am thankful to my dear friend Gökçen for being there for me whenever I needed and for taking my first steps into heuristics and algorithms together with me long before starting this thesis. I will never forget those days.

I would like to thank my best friend, my other half of the apple. You have inspired and encouraged me in many ways during our times together. I should thank you for believing in me more than I believe in myself.

Finally, I am grateful to my parents and my dear grandmother for bringing me up, being there for me at any time and condition, supporting me for everything I have decided to do. I would'nt be able to accomplish this much unless I knew you would stand by me.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

INTRODUCTION AND PROBLEM DEFINITION

In Turkey, drugs can only be obtained from pharmacies. Except for the pharmacies located in the government hospitals, each pharmacy is an individual enterprise that has to be owned and operated by a pharmacist. A pharmacist is allowed to operate only a single pharmacy. Also, pharmacy chains are not allowed in Turkey.

In a regular weekday, all pharmacies are open during the working hours. On national holidays, at weekends and at nights of the week days, only some of the pharmacies are allowed to be open for 24 hours. These pharmacies are called as on duty pharmacies. Based on a schedule, each pharmacy performs its duties multiple times in a year.

The duty schedules are prepared by the regional chambers of pharmacists.

In each chamber, there is a commission in charge of duty schedules. In İzmir, duties are being assigned to pharmacies manually by the On Duty Pharmacy Planning Commission of İzmir Regional Chamber of Pharmacists[1]. This manual system is causing some difficulties for both customers and the pharmacists. For the customers, main problem may be stated as traveling a long distance in order to find an on duty pharmacy. For the pharmacists side, sometimes two close pharmacies may be on duty at the same day. For this reason, we try to analyze this problem and develop analytical methods in order to solve the scheduling problem. We define this problem as the Pharmacy Duty Scheduling (PDS) Problem.

PDS problem deals with the assignment of duty days to different pharmacies for a specific length of time, which is called as the planning horizon. This problem is a multi-period facility location problem because it deals with the decision of opening and closing facilities on different days. PDS is a facility location problem because we try to decide on whether to open a pharmacy, in other words assign a duty to that pharmacy, or not. It is also a multi-period problem, since our planning horizon consists of multiple days and decisions given for a day affects the decisions for the other days.

In the scope of PDS problem, the pharmacies are the facilities. Opening a facility on a specific day means assigning a duty to that pharmacy on that day. The days in PDS problem represents different periods of the planning horizon. The demand for drugs is dispersed among many different households. In the

[1] www.izmireczaciodasi.org.tr

representation of the problem, it is almost impossible to locate each individual or household and to represent the corresponding demand one by one. Thus, in the PDS problem, we take the demand nodes to be the districts.

For a specific planning horizon, there may be a variety of assignments of duties to different pharmacies on different days of the planning horizon. This research aims to find the optimal assignments for a specific planning horizon. A schedule made by using the optimal assignments should lead to the least total cost of meeting the demand. Thus, the objective of PDS problem is to minimize the total demand weighted distance traveled by all customers during the planning horizon with respect to several operational constraints. The total demand weighted distance is calculated as the population of the district multiplied by the distance between the district center and the pharmacy that it gets service from.

For these purposes the following research questions are determined:

1. How should we measure the service quality of a duty schedule?
2. What is the quality of the current system?
3. How much can we improve the current system by the use of analytical methods to give the best service to the public?

In this thesis, we propose analytical methods in order to measure and increase the service quality. We find how much the current system may be improved by the use of proposed analytical methods.

The outline of this thesis is as follows. Chapter 1 presents the introduction, concepts used throughout the thesis and the definition of the problem. In Chapter 2, we provide a brief introduction to basic location decision problems. Then, we review related articles on facility location problems, Lagrangian Relaxation algorithms and Tabu Search heuristics. Chapter 3 describes the current system in detail and mentions the problems in the current system. In Chapter 4, we present the mathematical programming models developed. Chapter 5 introduces the algorithms developed in order to find lower and upper bounds for the problems. In Chapter 6, we discuss computational experiments and the results. Chapter 7 includes the İzmir application. Finally, we state conclusions and future research directions in Chapter 8.

Chapter 2

LITERATURE REVIEW

In this section, we review the related literature and present it in three main sections: i) general location problems, ii) emergency facility location problems, iii) tabu search.

2.1 General Location Problems

Hale and Moberg [1] define facility location problems as "*the problems investigating where to physically locate a set of facilities so as to minimize the cost of satisfying some set of demands subject to some set of constraints.*" These problems are covered under the title of location decision problems and have been an interesting research topic for a very long time.

Location decision problems have been mathematically formulated in different

forms in order to represent different situations or to satisfy different sets of constraints. In the following sections, some of these problems related to the research topic will be presented. These problems are:

- Covering Problems
- p-Median Problems
- Uncapacitated Facility Location Problems
- Multi-Period Capacitated Location Problems

2.1.1 Covering Problems

Covering problems are a class of location problems. There are different variations of covering problems; however, we will only focus on *set covering problems* in this thesis.

The set covering problem is a simple facility location problem. The objective is to select a minimum cost set of facilities from a set of candidate facilities so that every demand node is covered by at least one facility in the selected set. The following notation is used in the mathematical formulation of the set covering model by Daskin [2].

Notation

V_1: set of demand nodes, $i \in V_1$

V_2: set of candidate sites, $j \in V_2$

f_j = cost of locating a facility at candidate site j

$$a_{ij} = \begin{cases} 1, & \text{if candidate site } j \text{ can cover demands at node } i \\ 0, & \text{otherwise} \end{cases}$$

Decision Variables

$$X_j = \begin{cases} 1, & \text{if a facility is located at candidate site } j \\ 0, & \text{otherwise} \end{cases}$$

The mathematical formulation of the set covering problem is as follows:

Minimize

$$\sum_j f_j X_j \tag{2.1}$$

Subject to

$$\sum_j a_{ij} X_j \geq 1 \qquad \forall i \tag{2.2}$$

$$X_j \in \{0, 1\} \qquad \forall j \tag{2.3}$$

In the set covering model, the objective function (2.1) minimizes the total cost of locating facilities. Constraints (2.2) ensure that each demand node i is covered by at least one facility. Constraints (2.3) are the integrality constraints.

2.1.2 p-Median Problem

The p-median problem differs from the facility location problem, since the objective is to locate exactly p facilities so that the total cost of serving customers is minimized. In the general form of the p-median problems, the customer demand for a single commodity is met with a specified cost. The cost of meeting demand is calculated as the demand of node i multiplied by the distance between the demand node i and the facility j that it gets service from. The notation used in the mathematical formulation of the problem by Daskin [2] is as follows:

Notation

h_i: demand at node i

d_{ij}: distance between demand node i and candidate site j

p: number of facilities to locate

Decision Variables

$$X_j = \begin{cases} 1, & \text{if a facility is located at candidate site } j \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, & \text{if node } i \text{ is served by a facility at node } j \\ 0, & \text{otherwise} \end{cases}$$

The mathematical formulation of the p-median problem is as follows.

Minimize

$$\sum_i \sum_j h_i d_{ij} Y_{ij} \tag{2.4}$$

Subject to

$$\sum_j Y_{ij} = 1 \qquad \forall i \tag{2.5}$$

$$\sum_j X_j = P \tag{2.6}$$

$$Y_{ij} - X_j \leq 0 \qquad \forall i, j \tag{2.7}$$

$$X_j \in \{0, 1\} \qquad \forall j \tag{2.8}$$

$$Y_{ij} \in \{0, 1\} \qquad \forall i, j \tag{2.9}$$

In p-median problem, the objective function (2.4) minimizes the total demand weighted distance between each demand node and the nearest open facility. Constraints (2.5) state that each demand node i must be assigned to exactly one facility j while constraints (2.6) equalizes the number of facilities to be opened to exactly p. Constraints (2.7) ensure that the demand of node i can be assigned

to facility j only if a facility is opened at site j. Constraints (2.8) and (2.9) are the integrality constraints.

2.1.3 Uncapacitated Facility Location Problem

In the Uncapacitated Facility Location (UFL) problem, a number of facilities are chosen from a set of candidate facility locations in order to minimize the cost or maximize the profit while satisfying the customer demand for a single commodity. Generally, the associated costs are the transportation costs of distributing commodities between the facilities and the customers and the fixed costs for locating the facilities. If there is a capacity for each potential facility, then the problem is called as the *capacitated facility location problem*.

In the formulation of the UFL problem, there are a set of clients $I = \{1, ..., m\}$ with a given demand for a single commodity and a set of sites $J = \{1, ..., n\}$ where facilities can be located with a fixed cost f_j of opening facility j. The notation used to formulate the problem mathematically by Mirchandani and Francis [3] is as follows:

Notation

d_{ij}: cost of serving customer i from facility j

f_j: fixed cost of locating a facility at candidate facility site j

Decision Variables

$$X_j = \begin{cases} 1, & \text{if a facility is located at candidate site } j \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, & \text{if demands at node } i \text{ are served by a facility at node } j \\ 0, & \text{otherwise} \end{cases}$$

With this notation, the UFL problem is formulated as follows.

Minimize

$$Z = \sum_{i \in I} \sum_{j \in J} d_{ij} Y_{ij} + \sum_{j \in J} f_j X_j \tag{2.10}$$

Subject to

$$\sum_{j \in J} Y_{ij} = 1 \qquad \forall j \tag{2.11}$$

$$Y_{ij} \leq X_j \qquad \forall i, j \tag{2.12}$$

$$X_j \in \{0, 1\} \qquad \forall j \tag{2.13}$$

$$Y_{ij} \in \{0, 1\} \qquad \forall i, j \tag{2.14}$$

The objective function (2.10) minimizes the total cost of meeting customers by facilities. Constraints (2.11) require that each customer i is served by exactly

one facility. Constraints (2.12) state that in order to serve customer i from facility j, a facility must be opened at candidate site j. Finally, constraints (2.13) and (2.14) are the integrality constraints.

2.1.4 Multi-Period Capacitated Location Problems

Previously mentioned models are all single period models. The only question to be answered is where to locate the facilities. In the multi-period situation demand varies between different time periods. Thus, in multi-period problems, when to locate a facility is as important as where to locate it. Multi-Period Capacitated Location (MCL) problem consists of a set of periods $t = \{1, ..., T\}$ in which different amounts of customer demand should be met. The notation used in the formulation of MCL problem by Mirchandani and Francis [3] is as follows:

Notation

Z_{j0}: the initial capacity of facility j

w_{it}: demand of customer i at period t

$F_{jt}(z_{jt})$: the cost of a capacity expansion of size z_{jt} at the beginning of period t at location j

c_{jit} = the cost of producing one unit of flow at location j and transporting it to customer i in period t

Decision Variables

y_{jit}: the flow from location j to customer i in period t

z_{jt}: the capacity expansion at the beginning of period t at facility j

Z_{jt}: capacity available at period t at facility j

Using this notation, the MCL problem is formulated as follows:

Minimize

$$\sum_{t=1}^{T}\sum_{j=1}^{J}\left[F_{jt}(z_{jt}) + \sum_{i=1}^{I} c_{jit}y_{jit}\right] \tag{2.15}$$

Subject to

$$\sum_{j=1}^{J} y_{jit} = w_{it} \qquad \forall i,t \tag{2.16}$$

$$\sum_{i=1}^{I} y_{jit} \leq Z_{jt} \qquad \forall j,t \tag{2.17}$$

$$Z_{jt} = Z_{j(t-1)} + z_{jt} \qquad \forall j,t \tag{2.18}$$

$$Z_{jt} \geq 0 \qquad \forall j,t \tag{2.19}$$

$$z_{jt} \geq 0 \qquad \forall j,t \tag{2.20}$$

$$y_{jit} \geq 0 \qquad \forall i, j, t \tag{2.21}$$

The objective function, (2.15), minimizes the total cost of capacity expansions and serving the customers. Constraints (2.16) ensure that all demand is met while constraint (2.17) limits the flow from a facility to its capacity in each period. Constraints (2.19) state that the capacity of a facility in period t should be equal to the sum of its capacity in the previous period and capacity expansion made at the beginning of period t. Constraints (2.19), (2.20) and (2.21) define nonnegative decision variables.

The PDS problem is different from the location decision problems presented earlier. In location decision problems, the facilities remain open until the end of the planning horizon once they are opened. However, in PDS problem, the facilities are opened and closed multiple times during the planning horizon, since opening a facility in a period means assigning a duty for a pharmacy only in that period.

Wesolowsky and Truscott [4] work on dynamic facility location problem. They consider the location and relocation of facilities in a multi-period setting in response to demand changes. They use dynamic programming solution techniques in order to solve this problem. In each period, the existing facilities may be closed, relocated or remain open. However, in the PDS problem, each facility can be opened for only one day during the planning horizon and then it is closed.

The opening or closing of facilities may occur multiple times during the planning horizon.

After discussing these four models, their applications in the literature will be presented. Although these models are formulated in order to represent general situations, they can be used to model real life problems by introducing some side constraints as it is the case in the PDS problem.

2.2 Emergency Facility Location Problems

In general, emergency facility location problems try to locate emergency facilities so that the demands are met as soon as possible or within a given service threshold. Improving the system performance is very important for emergency services, thus, there are many examples of this type of problem in the literature.

Toregas et al. [5] modeled the *location of emergency service facilities* as a set covering problem. They aimed to open facilities that were within a limit of distance or time in order to reach the demand points since emergency facilities should be located very near to potential demand points. They introduced a mathematical program in order to find both optimal number and location of the facilities.

Çatay et al. [6] worked on planning the locations of the emergency medical service stations. Maximal covering model is used in order to solve both the single

period and multi-period emergency station location problems. They defined two different limits of time, t_1 and t_2, the former is the maximum time allowed to reach a demand node from the nearest station. The latter one is the time limit for the second closest station. The objective of the study is to maximize the population covered within these two limits of time by the stations located. Besides the exact solution methods proposed, they developed three different heuristic methods in order to solve the large size problems.

Carreras and Serra [7] worked on optimal location problem with threshold requirements. The model developed in the research is then applied to the pharmacy location problem in rural areas of Spain. The objective of the problem is determining the optimal locations for new pharmacies. In this problem, they tried to locate minimum number of facilities while keeping the service level above a defined threshold. They introduced a tabu search heuristic in order to solve this problem. This problem is different from the PDS problem. The aim of this problem is to locate new pharmacies, on the other hand, the aim of the PDS problem is to define duty schedules for the existing facilities.

Rajagopalan et al. [8] worked on dynamic redeployment of ambulances. They stated that the demand for ambulances change according to the day of the week or even time of the day. Thus, they aimed to increase the service performance by relocation and redeployment of ambulances. They first proposed a search algorithm in order to define the minimum number of ambulances to be located. Then,

they introduced a tabu search algorithm in order to locate the given ambulances.

2.3 Tabu Search Heuristics

Tabu Search is a meta-heuristic method developed to solve combinatorial optimization problems. It is based on searching the neighborhood of a given solution with the use of various neighborhood search techniques while not allowing to visit some solutions in order to search a broader space for better solutions. In Tabu Search heuristics, two types of decisions are made: tabu specific and problem specific decisions. For example, tabu tenures and aspiration criteria are tabu specific decisions . The problem specific decisions are related to problem type such as neighborhood structure, neighborhood search techniques etc.

Glover [9] introduced the Tabu Search algorithm and discussed general characteristics such as the aspiration criteria, tabu lists and memory functions. Tabu lists are kept in order to avoid getting stuck in local optima. By the use of tabu lists, recently visited solutions are prohibited and the search continues in different parts of the search space. Aspiration criteria are used when a very good solution is tabu. With the use of aspiration criteria, tabu solutions are accepted as admissible for instance, if they correspond to a solution that is better than the best known solution so far. Memory functions are divided into three levels as short-term, medium-term and long-term memories. These different memory functions are used throughout the Tabu Search algorithm.

Glover [10] introduced different neighborhood search methods for the Tabu Search algorithm. The neighborhood search techniques are used to search the different parts of the search space for better solutions. The fundamentals of the Tabu Search algorithm with different examples are introduced in Glover's tutorial on Tabu Search algorithm [11]. In addition to the previous researches, Glover and Laguna provide the fundamentals and variations of Tabu Search heuristics in detail in their book [12].

Tabu Search is a commonly used solution method in the literature. The application of Tabu Search algorithm varies from the facility location problems to scheduling problems.

Albareda-Sambola et al. [13], worked on multi-period incremental service facility location problem. This problem reduces to the p-median problem by limiting the number of facilities to p. In each period, at least a pre-defined number of customers' demand is met and the customers are served until the end of the planning horizon if they are serviced once at any time period. They have proposed a Lagrangian Relaxation algorithm by relaxing the constraint of assigning customers only to open facilities and obtained good solutions in reasonable computation times.

Wang et al. [14] worked on the budget constrained location problem in which some new facilities are opened and some existing facilities are closed. They applied their models on locating and relocating of bank branches in a large-size

town. They first developed a mathematical programming model and then they developed three heuristic approaches, namely greedy-interchange heuristic, tabu search approach and Lagrangian relaxation approximation for the problem. The greedy-interchange heuristic, consists of two phases. In the first phase, facilities are opened or closed to make the total number of open facilities equal to p. Phase two consists of an interchange procedure in which an open facility is swapped with an unopen facility in order to improve the objective function value. If there are no remaining swaps that improve the objective the algorithm stops and tabu search technique is applied to the solution in order to investigate local optima. Finally, they used the Lagrangian relaxation approximation in order to provide solutions within $\epsilon\%$ of the optimum. The solution methods are tested on real data for a specific bank and the performance of the algorithm was reported as good.

Al-Sultan et al. [15] worked on uncapacitated facility location problem using tabu search heuristics. They proposed a net benefit heuristic (NBH) in order to provide good quality initial solutions for the tabu search algortihm. In the initial phase of NBH, for each demand node, the facility with the least service cost is determined and it is opened unless it was opened before. The solution from the initial phase is then used as an upper bound to the problem. In the refinement phase of NBH, each facility that is opened in the initial phase is investigated. If closing an open facility provides benefit, than the demands are assigned to the next cheapest facility. After obtaining a good initial solution from the NBH, the tabu search technique is applied. The neighborhood solutions are searched

by generating random solutions and the one with the minimum cost is chosen. This process is repeated until a pre-defined maximum number of non-improving iterations are reached. The proposed tabu search algorithm was tested on some test problems and the performance of the algorithm was found to be better in comparison to the existing algorithms.

Bilgin and Azizoğlu [16] worked on a machine scheduling problem. The aim was to assign operations to different machines and allocate the capacity and the tools. For this problem, they first introduced a mathematical model for the problem. Then they proposed different heuristic methods and Lagrangean Relaxation algorithm in order to obtain lower bounds. Finally, they introduced a Tabu Search algorithm in order to solve this problem.

Chapter 3

CURRENT SYSTEM

In this chapter, we discuss the current pharmacy duty system in Turkey and provide the necessary information for İzmir.

In Turkey every pharmacy is registered to Pharmacists' Association[1]. This association has 53 regional chambers and İzmir is covered in the third region. In the metropolitan area of İzmir there are around 1300 pharmacies and the 3^{rd} Regional Chamber of Pharmacists, (Chamber from now on), is responsible for the duty scheduling of these pharmacies.

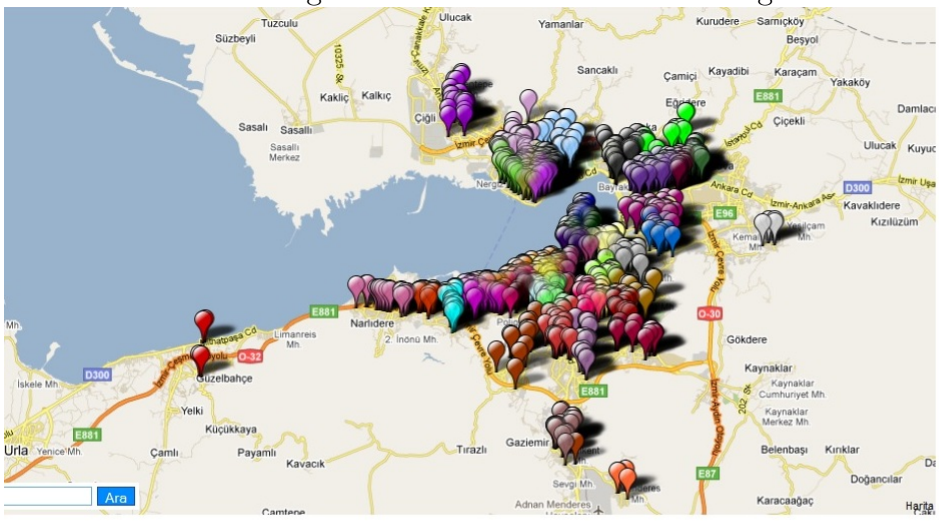
In the current planning system, there are three different types of duties. These are the duties on national holidays, duties on weekends and duties at nights on the week days. Saturday is considered as a week day between September and May due to high demand and the pharmacies are open during the working hours. In summer, when the demand is low, it is considered as weekend and only on

[1] www.teb.org.tr

duty pharmacies are open on Saturdays. For each type of duty, the sequence of pharmacies to be on duty differs although the same scheduling rule is applied. The duties on national holidays are the least preferred ones by the pharmacists and a special sequence is used for this type of duties.

There are 11 counties in the metropolitan area of İzmir. These are Balçova, Bayraklı, Bornova, Buca, Çiğli, Gaziemir, Güzelbahçe, Karabağlar, Karşıyaka, Konak and Narlıdere. These counties are divided into regions in order to simplify the planning and the control process of the on duty pharmacy schedules. There are a total of 45 regions in the metropolitan area of İzmir. Figure 3.1 shows the pharmacies in 45 regions on the map. In the figure, different colors represent different regions.

Figure 3.1: Pharmacies from 45 Regions



For each region of Izmir, a pharmacy is chosen in order to represent its region in the Chamber. During each period, the representative pharmacists propose a duty schedule for the next planning period. After the group of representatives finalize the schedule, the duty pharmacy commission of the Regional Chamber of Pharmacists controls the schedule and makes changes if needed. After that, the Regional Health Department of Izmir gives the last approval.

In Izmir, the length of each planning horizon is determined as 4 months and there are 3 periods in a year. Each Chamber determines the length of the planning horizon itself, in other words, there is no legal restriction regarding the length of the planning horizon. For example, in Istanbul, the 1^{st} Regional Chamber of Pharmacists has determined the length of the planning horizon as one year. In Ankara, the length of the planning horizon is also determined as 1 year by the 2^{nd} Regional Chamber of Pharmacists. In Izmir, they preferred a shorter period length. A shorter planning horizon is more flexible since accounting for changes a year ahead is hard. Many changes may occur within a year, for instance, new pharmacies may start business, some may go out of business, or some unpredictable events like illnesses may occur. Thus, a shorter planning horizon increases the control of the Chamber on the schedule. On the other hand, too short planning horizons are not preferred, either. The planning process of the on duty pharmacy schedules is a time consuming and difficult process for the chambers. Hence, they do not want to deal with it very frequently.

In each region, all the pharmacies which are able to take duties are listed according to their license numbers. Three copies of the same list are used seperately for three types of duties. The upcoming duties are assigned to pharmacies by order of this on duty pharmacy list. The first duty of a period is given to the pharmacy following the last on duty pharmacy of the previous period on the list. Then all the upcoming duties are assigned to pharmacies following the order in the list. When they reach the end of list, they continue to assign duties from the top of the list. This continues until all days in a planning period are assigned to one pharmacy from each region. The pharmacy following the-last-duty-assigned-pharmacy, again, takes the first duty in the next period. This is applied for all types of duties separately for each region.

The number of pharmacies in the metropolitan area of İzmir is different from the number of pharmacies taking duties. There is a difference because some parts of the regions are excluded from the on duty pharmacy regions. These parts are excluded because they are far away from both central residential areas and health centers even they are located in the metropolitan area. In addition, some pharmacies in the metropolitan area are excluded from the duty list if they have a reason for not taking duties. These reasons, such as illness of the pharmacist or the pharmacist being too old for a duty, should be accepted by the Chamber. These pharmacies may be allowed not to take duties temporarily or may permanently be exempt from duty with regard to their reasons. Because of the previously mentioned reasons, the number of pharmacies in the on duty

Table 3.1: A sample part of a duty schedule 2010

Licence #	Pharmacy	September	October	November	December
230	Güçlü	30		13	5W 31
253	Mavişehir Sağlık		1	15	12W
262	Bostanlı Pınar		2	20	19W
305	Damla		7	22	26W
310	Bostanlı Saadet		5	23	
⋮	⋮	⋮	⋮	⋮	⋮
507	Site		16 29H		4
End of List					
10	Çaylı	1	18		6

H: national holidays, W: weekends

pharmacy list may change from one period to another. One month prior to the start of each planning period, the pharmacies to be included in the on duty pharmacy list are determined. There are approximately 1300 pharmacies in the metropolitan area of İzmir. However, there were a total of 1046 pharmacies taking duties in the 3^{rd} period of 2010 due to the previously mentioned reasons. The number of pharmacies taking duties has increased to 1053 for the 1^{st} period of 2011.

Table 3.1 shows a sample from the on duty pharmacy schedule for the 3^{rd} period of 2010 from Bostanlı region. In the table, the numbers represent the days of the months. The days shown with letter W denote the duties on weekends and days shown with letter H denote the national holidays. The rest of all duties are night duties on week days. For the 3^{rd} period of 2010, only Sundays are considered as weekend since it is the high season. The last day of the 3^{rd} period

of 2010 is 31^{st} of December. The last duty is assigned to the Güçlü Pharmacy. This implies that the first night duty of the next period, which is the 1^{st} period of 2011, will be assigned to Mavişehir Sağlık Pharmacy. Likewise, the last Sunday of the period is the 26^{th} of December which is assigned to Damla Pharmacy. Thus, the first Sunday duty of the following period will be assigned to Bostanlı Saadet Pharmacy. The same rule applies with the national holiday duties. The last national holiday of the period is 29^{th} of October and it is assigned to Site Pharmacy. Thus, the first national holiday of the following period, which is 1^{st} of January, will be assigned to Çaylı Pharmacy .

As it can also be seen from the table, Damla and Bostanlı Saadet in October, sometimes the duty dates are not assigned in the given order. This is simply because some pharmacies exchange their duty dates by their own mutual will and it should also be approved by the Chamber.

The main objective of duty scheduling is to locate the pharmacies on duty in such a way that the customers requiring drugs may reach a pharmacy easily, in every day of the planning horizon. The most important problem of the current system is making assignments not centrally but within the specified regions. This assignment method sometimes causes two pharmacies that are very close to each other but from different regions to be on duty on the same day. Since there is only one on duty pharmacy for each region each day, such an assignment may cause some customers to travel a far distance in order to find an on duty pharmacy.

As a conclusion, the duty assignment decisions are made with a very simple method on a regional basis in the current system. There is no central planning system and the authorities are not using a decision support system. The central planning system is also mentioned to be more advantageous by the authorities of the Chamber, however, they also mentioned that it is a very complex system for them to apply. Aim of this research is to provide a centralized decision support system. Although central planning is a more complex system than the current one, it will help to solve the problems faced in the current system.

Chapter 4

MATHEMATICAL MODELS

In this chapter, we develop different mathematical programming models for the pharmacy duty scheduling problem and prove that the PDS problem is NP-Hard. In the first part of the chapter, simple models are introduced. In the second part, more realistic models are presented. All models included in this chapter are mixed integer programming models. While developing these models, we have made some assumptions regarding the demand points and the distances between demand points and the pharmacies.

In real life applications, it is very hard to determine the customer nodes and their demands precisely. Likewise, in our problem, representing each individual customer location with a customer node is not possible. So, we assume an aggregate customer node where a group of people living in the same neighborhood will be represented. We assume that the customer nodes, which are denoted by i, are located at the centers of the districts. There are a total of 350 districts

in the metropolitan area of İzmir. The demand in each customer node, which is represented as h_i, is determined as the population of these districts. Combining these two assumptions, our demand size and customer node representations become more accurate.

The distances between the customer nodes and the pharmacy locations are assumed to be rectilinear distances. These distances , which are represented as d_{ij}, are calculated as the rectilinear distances using the real coordinates of the pharmacy locations and the districts. In real life application, reasonable modifications are done in distance calculations. The total demand weighted distances in the models represented in the following sections are calculated using the distances and the demand sizes mentioned previously.

The length of the planning horizon which is denoted by T is an input for the mathematical models. In all of the models except Simple Model 2, it is a given parameter which is defined by the Chamber. In Simple Model 2, T is a decision variable and we try to maximize the length of the planning horizon.

4.1 Simple Models

There are two different models developed under the simple models section. These models are called as simple models since we ignore some real life constraints. These models can also be referred as the single duty models, since the pharmacies

are taking only one duty during the whole planning horizon. There are two approaches for the simple models. The first one is minimizing the total distance traveled. The second approach is to maximize the length of the planning horizon with respect to a defined service threshold.

4.1.1 Simple Model 1

The first simple model aims to minimize the total demand weighted distance traveled by the customers during the whole planning horizon. On each day t of the planning horizon, each customer i is serviced by at least one facility j. In the first model, the length of the planning horizon is determined as T. Thus, this model tries to minimize the total distance traveled in T days. In the formulation of the model, we make use of the following notation.

Notation

i: demand points (customer nodes), i={1, ..., I}

j: facility sites (pharmacies), j={1, ..., J}

t: time periods (days), t={1, ..., T}

h_i: demand at customer node i

d_{ij}: distance between customer node i and facility site j

Decision Variables

$$X_{ijt} = \begin{cases} 1, & \text{if customer } i \text{ is served by facility } j \text{ on day } t \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{jt} = \begin{cases} 1, & \text{if facility } j \text{ is opened on day } t \\ 0, & \text{otherwise} \end{cases}$$

Model 1 is formulated as follows with the use of the notation presented:

Minimize

$$F = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{t=1}^{T} h_i d_{ij} X_{ijt} \tag{4.1}$$

Subject to

$$X_{ijt} \leq Y_{jt} \qquad \forall i, j, t \tag{4.2}$$

$$\sum_{t=1}^{T} Y_{jt} \leq 1 \qquad \forall j \tag{4.3}$$

$$\sum_{j=1}^{J} X_{ijt} \geq 1 \qquad \forall i, t \tag{4.4}$$

$$Y_{jt} \in \{0, 1\} \qquad \forall j, t \tag{4.5}$$

$$X_{ijt} \in \{0,1\} \qquad \forall i,j,t \tag{4.6}$$

In simple model 1, the objective function (4.1) minimizes the total demand weighted distance traveled by all customers. Constraint set (4.2) ensures that no customer is assigned to an unopened facility at any time period. Constraint set (4.3) guarantees that no facility is opened more than once during the whole planning horizon. Constraint set (4.4) fulfills the necessity that each customer must be assigned to a facility in each day of the planning horizon. Constraint sets (4.5) and (4.6) define the binary decision variables.

Note that the integrality constraints on X_{ijt} variables can be relaxed since in an optimal solution, each customer will be assigned to the closest open facility on each day due to the minimization objective and the unlimited capacity of the facilities.

After solving this model to optimality, for any day t, we have the sets of pharmacies to be opened. In the optimal solution, exchanging days with each other does not make any difference to the objective value if the pharmacies in the same set are opened together. Thus, this yields to a $T!$ alternative optimal solutions. As a modification for this model, we define a new decision variable f_t that shows the total demand weighted distance traveled by all customers for day t.

$$f_t = \sum_{i=1}^{I} \sum_{j=1}^{J} h_i d_{ij} X_{ijt} \qquad \forall t$$

We can rewrite model 1 using f_t variables as follows:

Minimize

$$F = \sum_{t=1}^{T} f_t \tag{4.7}$$

Subject to

$$f_t \leq f_{t+1} \qquad \forall t \tag{4.8}$$

(4.2), (4.3), (4.4), (4.5), (4.6)

In this model, objective function (4.7) minimizes the total demand weighted distance traveled by all customers during the whole planning horizon. The only difference of this model is the additional constraint set (4.8) that state the total distance traveled by all customers on a given day must be less than the total distance traveled by all customers on the following day. In this problem, our aim is to find the optimal sets of pharmacies that should be opened together on the same day and we are not interested in finding which set of pharmacies is opened on which day of the planning horizon.

In the optimal solution of the modified model, day 1 provides the highest service quality and day T provides the lowest service quality. The order of days in the model does not necessarily represent the order of days in real life. One may

want to assign day 1 of the model to the day with the highest demand. Hence matching the model days with the calender days is another decision to be made.

As an observation on the problem, we may state that, in a one-day-problem, all potential pharmacies are opened on day 1. So, all customers are served by the facilities that are closest to them. When we increase the time horizon to two days, it means that the customers should be served for two days. Since a facility cannot be opened more than once during the whole planning horizon, we have to split the opened locations between two days. In this case, on some days, some customers are still served by the facilities that are closest to them. On the other hand, some of them may get service from the facilities that are not the closest ones.

We define F^1 and F^2 as the optimal objective values for the one-day-problem and two-days-problem, respectively. We also define F_1^2 and F_2^2 as the objective values of each day in a two-days-problem. By definition,

$$F^2 = F_1^2 + F_2^2$$

Since, all customers cannot be served by the closest facilities on both days of a two-days-problem, we can write the following equation:

$$F_1^2 \geq F_1 \text{ and } F_2^2 \geq F_1$$

Hence, if we sum F_1^2 and F_2^2 values, we get the following equation:

$$F^2 \geq 2 \times F^1$$

In order to generalize this observation, we define F^t as the optimal objective value of a t-days-problem.

$$F^t \geq t \times F^1$$

4.1.2 Simple Model 2

In the second simple model, we aim to maximize the length of the planning horizon while keeping the service quality above a desired level. For this purpose, we define a new decision variable g_t.

$$g_t = \begin{cases} 1 & \text{if service is provided to each customer on day } t \text{ within distance } DT \\ 0 & \text{otherwise} \end{cases}$$

The additional notation used in the model is as follows:

DT: distance threshold $\alpha_{ij} = \begin{cases} 1, & \text{if} \quad d_{ij} \leq DT \\ 0, & \text{otherwise} \end{cases}$

The mathematical model is as follows:

Maximize

$$G = \sum_{t=1}^{T} g_t \tag{4.9}$$

Subject to

$$g_{t+1} \leq g_t \qquad \forall t \tag{4.10}$$

$$g_t \leq \sum_{j=1}^{J} Y_{jt} \qquad \forall t \tag{4.11}$$

$$\sum_{j=1}^{J} \alpha_{ij} X_{ijt} \geq g_t \qquad \forall i, t \tag{4.12}$$

$$g_t \in \{0, 1\} \forall t \tag{4.13}$$

(4.2), (4.3), (4.5), (4.6)

The objective function (4.9) maximizes the number of days customers are getting service in a lexicographic manner. Constraint set (4.10) states that in order to give service on day $t+1$, service must be given on day t. Constraint set (4.11) shows that in order to give service in a specific day, at least one facility should be opened on that day. Constraint set (4.12) ensures that each customer is served by a facility within the given distance limits if service is given on day t. Finally, constraint set 4.13 states that decision variable g_t is binary variable. In addition to these, constraints (4.2), (4.3), (4.5), (4.6) from Model 1 also hold for Model 2.

As a conclusion, models in this section are useful for understanding the basics of the problem and to develop alternative models and solution techniques (see Ch5). However, these models assume that each facility should be opened once during the planning horizon. In real life, this is a strong assumption and should

be relaxed. In section 4.2, we develop more realistic models by relaxing this assumption.

4.2 Realistic Models

The realistic models are more complex models than the models developed in the previous section. They also take constraints mentioned by the authorities of the Chamber into account. The realistic models may also be referred as the multiple duty models, since the basic difference of these ones are the multiple duties assigned to the pharmacies.

The first model in this section, considers the regional setting applied in the current system and assigns multiple duties for the pharmacies. The second model developed does not consider the regions of the current system but still distributes the duties for close pharmacies evenly.

4.2.1 Realistic Model 1

The realistic model 1 considers the regions in the current system. There are multiple duties taken by the pharmacies and the number of the duties are determined according to the region of the pharmacy. Thus, pharmacies in different regions may be assigned different numbers of duties.

In the multiple duty problem, the duty numbers of pharmacies vary according

to the number of pharmacies in their region. If there are many pharmacies within the region of a pharmacy, then the duties for that pharmacy is less frequent. On the other hand, if there are very few pharmacies in a region, then those pharmacies take duties more often.

The average number of duties to be assigned to any pharmacy within a region may be found by dividing the total number of days in a planning period by the total number of pharmacies in that region. If we round this number down, we find the duty number lower bound for a given pharmacy and likewise, duty number upper bound when the number is rounded up. These upper and lower bounds on the duty numbers are calculated for each region and used in order to assign duties evenly to pharmacies within the same region.

The following notation introduced in the previous section is also used in this section.

i: demand points (customer nodes), i={1, ..., I}

j, q: potential facility sites, j={1, ..., J}, q={1, ..., J}

t: time periods (days), t={1, ..., T}

h_i: demand at customer node i

d_{ij}: distance between customer node i and potential facility site j

Decision Variables

The same decision variables from Simple Models section are used in this model.

$$X_{ijt} = \begin{cases} 1, & \text{if customer } i \text{ is served by potential pharmacy } j \text{ on day } t \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{jt} = \begin{cases} 1, & \text{if potential pharmacy } j \text{ is opened on day } t \\ 0, & \text{otherwise} \end{cases}$$

The additional notation introduced is as follows:

k: regions, k={1, ..., K}

J_k: set of pharmacies in region k

$$J = \bigcup_{k=1}^{K} J_k$$

Minimize

$$F = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{t=1}^{T} h_i d_{ij} X_{ijt} \tag{4.14}$$

Subject to

$$\sum_{t=1, j \in J_k}^{T} Y_{jt} \leq \left\lceil \frac{T}{|J_k|} \right\rceil \qquad \forall k \tag{4.15}$$

$$\sum_{t=1, j \in J_k}^{T} Y_{jt} \geq \left\lfloor \frac{T}{|J_k|} \right\rfloor \qquad \forall k \tag{4.16}$$

$$X_{ijt} \leq Y_{jt} \qquad \forall i, j, t \tag{4.17}$$

$$\sum_{j=1}^{J} X_{ijt} \geq 1 \qquad \forall i, t \tag{4.18}$$

$$\sum_{j:q(j,k)=1} Y_{jt} = 1 \qquad \forall k, t \tag{4.19}$$

In the model, objective function (4.14) minimizes the total demand weighted distance traveled by all customers. In constraint sets (4.15) and (4.16) the value of $\frac{T}{|J_k|}$ shows the exact number of duties to be assigned to pharmacy j in region k. Thus, these constraint sets, make the total number of duties assigned to pharmacy j equal to the the number of duties assigned to other pharmacies within the same region. The constraint set (4.17) states that in order to assign customer i to pharmacy j on day t, the pharmacy should be open on day t. Constraint set (4.18) requires that each customer is assigned to at least one pharmacy on each day of the planning horizon. Finally, constraint set (4.21) states that only one pharmacy can be opened from each region on each day of the planning horizon.

4.2.2 Realistic Model 2

In this model, pharmacies again may take duties more than once during the planning horizon. The authorities from the Chamber define the most important characteristic of the planning system as the fair distribution of duties to the pharmacies. The Chamber tries to assign same number of duties to the pharmacies in the same region. Although this approach is reasonable within a region, two close

pharmacies may take extremely different number of duties.

In realistic model 2, we aim to assign similar number of duties to pharmacies close to each other. This model does not use regions defined by the Chamber.

Notation

The following notation are the additional notation introduced for this model.

α: distance limit defined to set the neighborhoods

β: error term which represents the allowable difference in number of duties of a pharmacy and the average number of duties of the pharmacies within the pharmacies in α neighborhood

S_j: the set of pharmacies within the α neighborhood of pharmacy j

$$S_j = \bigcup_{k:d_{jk} \leq \alpha, k \neq j} \{k\}$$

The mathematical formulation of the problem is as follows.

Minimize

$$F = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{t=1}^{T} h_i d_{ij} X_{ijt} \tag{4.20}$$

Subject to

$$\sum_{t=1}^{T} Y_{jt} \leq \beta + \frac{1}{|S_j|} \sum_{k \in S_j}^{K} \sum_{t=1}^{T} Y_{kt} \qquad \forall j \tag{4.21}$$

$$\sum_{t=1}^{T} Y_{jt} \geq -\beta + \frac{1}{|S_j|} \sum_{k \in S_j}^{K} \sum_{t=1}^{T} Y_{kt} \qquad \forall j \tag{4.22}$$

(4.17), (4.18)

In this model, the objective function (4.20) is the same with the objective function of realistic model 1. Constraint sets (4.21) and (4.22) state that the number of duties taken by pharmacy j should be within a β error limit of the average number of duties taken by the pharmacies within the α-neighborhood of pharmacy j.

Both models developed in this section are memoryless and do not consider the schedules of previous planning periods. Some pharmacies may be assigned more duties than the others. In the long run, the total number of duties assigned to different pharmacies may not be the same since some pharmacies are assigned duties on the duty number lower bound and some on duty number upper bound. Thus, this may lead to an unfair schedule. To be fair, a new parameter,that keeps the number of duties in the previous planning horizon for each pharmacy may be added to the models. Hence, duties may be fairly distributed among the pharmacies in the long run.

In the upcoming chapters, we will focus on two models. These are Simple

Model 1 that assigns one duty to each pharmacy and Realistic Model 1 that assigns multiple duties to pharmacies during the planning horizon.

We omit Simple Model 2, because maximizing the length of the planning horizon is not appropriate for application. The aim of the Chamber is to distribute the duties fairly for a specific time period. We also omit the Realistic Model 2, since, the α and β values are not easy to define and would be hard to apply by the authorities.

4.3 Computational Complexity Analysis for the PDS Problem

In this section, we prove that the Pharmacy Duty Scheduling problem with single and multiple duties are NP-Hard.

Theorem 1. *PDS-1 problem is NP-Hard.*

Proof. Assume that the planning horizon length is T is 2 days, there are I customers and J pharmacies. Since the pharmacies in this case can only take one duty during the planning horizon, the aim is to separate J pharmacies into 2 partitions in order to minimize the total demand weighted distance. Thus, a group of pharmacies out of J will be selected to take duties on the first day and the remaining will be on duty on the second day.

In general, this problem aims to partition the pharmacies into T partitions

so that the total cost is minimized. PDS-1 problem is equal to set-partitioning problem and NP-Hard for variable T. □

Theorem 2. *PDS-M problem is NP-Hard.*

Proof. Assume that planning horizon T is 1 day, number of regions is p, there are n pharmacies ($n >> p$) in each region and hence $J = np$. Pharmacies $(k-1)n+1, \ldots, kn$ are in region k. There are I customers.

Let $d_{i,j} > 0$ for $i = 1, \ldots, I$, $j = 1, \ldots, J$ and $d_{i,j} = d_{i,j+kn}$ for $k = 1, \ldots, p-1$, $j = 1, \ldots, n$, $i = 1, \ldots, I$. We may simply consider pharmacies j, $j+n$, $j+2n$,..., $j+(p-1)n$ are located at the same place. Therefore there are n distinct pharmacy locations; each includes p pharmacies, one from each region. By construction, all pharmacies at the same location are equally distant to all customers. Since $n > p$ and $d_{ij} > 0$ for $i = 1, \ldots, I$, $j = 1, \ldots, n$, it is obvious that in the optimal solution, at most one pharmacy can be selected for the duty from a pharmacy location. Note that $T = 1$ and only one pharmacy is selected from each region. This special case of PDS-M problem aims to minimize the demand weighted distance by selecting p locations out of n candidates. PDS-M problem is equal to p-median problem and NP-hard for variable p. □

4.4 Concluding Remarks

In this chapter, we propose 4 mathematical models. In the following chapters we will use Simple Model 1 and Realistic Model 1. We will not use Simple Model 2 and Realistic Model 2, since these models are hard to apply for the parties involved in real life. The models applied to the real case should be accepted by the parties involved and they should be simple tools for application. Thus, in real application we omit these two models.

We evaluate the PDS problem only from the side of customers. The problem may also be evaluated from the pharmacists' side. In our models, it is not likely that two pharmacies that are very close to each other to be opened on the same day. As a by product, our models are also beneficial for the pharmacies. However, evaluating the problem from the pharmacists' side and proposing direct benefits for them may be evaluated as a future research.

Throughout the thesis, simple model and single duty models will be used interchangeably in order to represent the problem type in which pharmacies are assigned only one duty during the planning horizon. Realistic model and multiple duty models will be used interchangeably in order to represent the problem type in which pharmacies are assigned multiple duties during the planning horizon.

Chapter 5

UPPER AND LOWER BOUNDS

In Chapter 4, we proved that PDS problem with both single and multiple duties are NP-hard. In addition to this, during the preliminary test runs, we have realized that as the problem size increases, it gets harder to obtain results in reasonable computation times. We tested our models with one hour runs for different instances and optimal solutions cannot be found within one hour limit for the large-size problems. Since, the real problem to be solved is a very large-size problem, we decided to use heuristic based methods. Thus, we developed some heuristic and meta-heuristic algorithms in order to obtain lower and upper bounds on the objective values of the problems.

In this chapter, we introduce different algorithms used to obtain bounds for large-size problems. In the first section, we discuss algorithms developed to find

upper bounds for the problems. First, we introduce the initial solution heuristics for single duty (simple model) and multiple duty (realistic) problems. Then, we discuss the Tabu Search algorithm.

In the second section of this chapter, we introduce the algorithms developed to obtain lower bounds for the problems. We first discuss the Lagrangean Relaxation algorithm and then AO_S, AO_M1 and AO_M2 lower bound algorithms.

5.1 Upper Bounds

In the first part of this section, we introduce two initial solution heuristics for single duty and multiple duty models. In the second part, we introduce the Tabu Search algorithm.

We compute the cost of given duty assignments with the use of an algorithm (Algorithm 5.1). When the optimal duty dates are known for the pharmacies, then the customers may be matched with the pharmacies optimally. Thus, this algorithm simply assigns the customers to the closest open pharmacies for each day of the planning period. The cost of each assignment is the demand weighted distance between the customer node and the pharmacy. After assigning each customer to an open pharmacy on each day of the planning horizon, these costs are summed for all customers and all days. Thus, the total cost of assigning customers to pharmacies are calculated for a planning horizon.

Algorithm 5.1 Cost Computation Algorithm

```
obj_val = 0
for t = 1 → T do
  for i = 1 → I do
    min_d = M
    for j = 1 → J do
      if Y_jt = 1 and d_ij < min_d then
        min_d = d_ij
      end if
    end for
    obj_val = obj_val + h_i × min_d
  end for
end for
```

5.1.1 Initial Solution Heuristics

We developed simple heuristic algorithms in order to provide initial solutions for the Tabu Search algorithms. These algorithms also provide upper bounds on the objective value of the problem. In the following sections initial solution heuristics for single and multiple duty models are introduced.

5.1.1.1 Initial Solution Heuristic for Single Duty Problem

In the single duty problem, at most one duty is assigned to each pharmacy during the planning period. In this algorithm, we assign duties to pharmacies in a simple way without taking the objective value into account. In other words, one duty is assigned to each pharmacy in an ascending order for the days of the planning period. The pseudocode for the initial solution heuristic for single duty problem is given as Algorithm 5.2.

Algorithm 5.2 Initial Solution Heuristic for Single Duty Problem

```
for j = 1 → J do
  if t ≥ T then
    t = 0
  end if
  t = t + 1
  Y_jt = 1
end for
Compute Cost
```

After applying the initial solution heuristic algorithm, we obtain a feasible solution to the problem. Then, we compute the cost of the given feasible duty assignments and use these assignments as the starting solution for the Tabu Search algorithm. In addition, this solution itself is an upper bound on the objective value of the problem, since the optimal solution may not be worse than any feasible solution.

5.1.1.2 Initial Solution Heuristic for Multiple Duty Problem

In the multiple duty problem, each pharmacy may be assigned more than one duty. However, on each day of the planning period, there should only be one on duty pharmacy from each region. Thus, the initial solution heuristic algorithm is changed accordingly. This algorithm is also a simple one and assigns duties to pharmacies in the pharmacy index order without paying attention to the cost of the assignments. The only difference of this algorithm from Algorithm 5.2 is that only one pharmacy is opened from each region on each day of the planning horizon. The set of pharmacies within region k is defined by J_k. The pseudocode

for this algorithm is given as Algorithm 5.3.

Algorithm 5.3 Initial Solution Heuristic for Multiple Duty Problem

```
for k = 1 → K do
  j = 1
  for t = 1 → T do
    assign = 0
    while assign = 0 do
      if j ∈ J_k then
        Y_jt = 1
        assign = 1
      end if
      j = j + 1
      if j > J then
        j = 1
      end if
    end while
  end for
end for
Compute cost
```

5.1.2 Tabu Search Algorithm

In Chapter 4, we proved that our problem is NP-hard. Due to this property and the large-size of the real problem, it cannot be solved to optimality in reasonable time. Thus, in this section, we introduce a meta-heuristic method in order to solve large-size instances like the real life problem.

Tabu Search Heuristics is a strategy developed for solving combinatorial optimization problems and it has been widely used for various problems [9]. The algorithm depends on searching the neighborhood of a given solution by modifying the solution with different moves by the use of predefined neighborhood search techniques. At any stage, the aim is to find better solutions than the best

known so far. Sometimes moves to some solutions, which are called as the tabu solutions, are restricted in order to escape from stucking at local optima. Tabu search is found to be a very efficient algorithm and it is used in many facility location problems in the literature.

In the tabu search algorithm, two types of decisions are made. Firstly, we decide on the pharmacies to be opened on each period of the planning horizon. Secondly, we decide on the assignment of customers to open facilities while minimizing the total demand weighted distance traveled by all customers throughout the planning horizon.

We first present the Tabu Search algorithm for single duty problem. Then, we discuss the algorithm for multiple duty problem.

5.1.2.1 Tabu Search Algorithm for Single Duty Problem

In order to solve the single duty problem, we have developed a tabu search algorithm. The pseudocode of the Tabu Search algorithm for the single duty problem is stated as Algorithm 5.4.

In the tabu search algorithm for the single duty problem, *swap* and *move* methods are used in order to search the neighborhood of a given solution. In the swap algorithm, (Algorithm 5.5), the duty days of two pharmacies are changed with each other. A pharmacy is selected and the possible swaps with all other pharmacies that are on duty on a different day are determined. The costs of these

Algorithm 5.4 Tabu Search Algorithm for Single Duty Problem

```
while outer_iteration < maximum_outer_iteration do
  if outer_iteration = 1 then
    Generate initial solution
  else
    Generate new initial solution
  end if
  while inner_iteration < maximum_inner_iteration do
    Search best swap
    Search best move
    if F_best_cand_swap < F_best_cand_move then
      Apply best swap
    else
      Apply best move
    end if
    Update frequencies
    Update tabu status
    if F_current < F_best then
      Update best solution
    end if
  end while
end while
```

swaps are calculated. If swapping the duties of two pharmacies improves the objective value, this solution becomes the candidate solution. After all possible swaps are checked the best one among them is determined and chosen as the best candidate solution.

Algorithm 5.5 Swap Heuristic for Single Duty Problem

```
for j1 = 1 → J do
  for t1 = 1 → T do
    if Y_{j1 t1} = 1 then
      for j2 = j1 + 1 → J do
        for t2 = 1 → T do
          if Y_{j2 t2} = 1 & t1 ≠ t2 then
            Y_{j1 t1} = 0
            Y_{j1 t2} = 1
            Y_{j2 t1} = 1
            Y_{j2 t2} = 0
            Compute candidate solution cost
            if F_{cand_swap} < F_{best_cand_swap} then
              if j1, j2, t1, t2 not tabu then
                Update best candidate swap
              else
                if F_{cand_swap} < F_{best} then
                  Update best candidate swap
                end if
              end if
            end if
            Y_{j1 t1} = 1
            Y_{j1 t2} = 0
            Y_{j2 t1} = 0
            Y_{j2 t2} = 1
          end if
        end for
      end for
    end if
  end for
end for
```

Likewise the swap algorithm, the *move algorithm* (Algorithm 5.6) is used to

search the neighborhood of a given solution. In this search method, an on duty pharmacy is selected and it is moved to another day of the planning period. In other words, we check whether the solution quality improves if the selected pharmacy was opened on any other day than the day it is actually opened or not. After all possible moves are checked, the best candidate move resulting in the highest improvement is determined.

Algorithm 5.6 Move Heuristic for Single Duty Problem

```
for j = 1 → J do
  for t1 = 1 → T do
    if Y_{jt1} = 1 then
      for t2 = 1 → T do
        if Y_{jt2} = 0 then
          Y_{jt1} = 0
          Y_{jt2} = 1
          if F_{cand_move} < F_{best_cand_move} then
            if j, t1, t2 not tabu then
              Update best candidate move
            else
              if F_{cand_move} < F_{best} then
                Update best candidate move
              end if
            end if
          end if
          Y_{jt1} = 1
          Y_{jt2} = 0
        end if
      end for
    end if
  end for
end for
```

During the search process of better solutions, we make use of tabu tenures in order to escape local optima for both search methods. We assign a tabu tenure to recently visited solutions and prevent the algorithm to search for solutions in a

small portion of the search space. In the single duty problem, we keep tabu lists for the days and the pharmacies. For each inner iteration and for each move type, the tabu status of the candidate solutions are checked. If the candidate solutions are in the tabu list, they are not denoted as the best candidate solutions unless they meet the aspiration criteria. The aspiration criteria are used in order to allow the algorithm to move to the solutions with a very good objective value even if they are tabu. In our Tabu Search Algorithm, we allow a tabu solution to be indicated as the best candidate solution if the solution gives an objective value better than the best known solution so far.

Following the previous steps, the best candidate solutions for each type of search methods are determined with the help of neighborhood search algorithms. After that, these two best alternatives from each type of search methods are compared and the better one among them is chosen as the best solution for that inner iteration. The selected best solution becomes the initial solution for the next inner iteration and the neighborhood searches are made for this solution. This process continues until the inner iteration number reaches to its maximum.

After each inner iteration, we check if the newly found solution is better than the best known solution so far. If so, the best solution found so far is updated and the new solution is used as the best solution for making comparisons in the upcoming iterations.

In the Tabu Search Algorithm for single duty problem, we make use of intensification and diversification strategies [10, 11, 12]. We first create an initial solution and intensify the search by *swap* and *move* search methods. After a number of iterations, that is the maximum_inner_iteration, a new initial solution is generated in order to diversify the search space. This technique is used in order not to get stuck in a local optima and search a larger portion of the search space. However, starting from a totally new or random initial solution is not preferred either. Thus, we keep a record of the frequencies of the pharmacy pairs that are opened on the same day. While generating the new initial solution, the pharmacy pairs with high frequencies in the evaluated solutions so far are assigned to different days of the planning horizon. The pseudocode for the new initial solution is given as Algorithm 5.7. In the first part of the algorithm, the pharmacies with the highest frequency of opening on the same day is found. In the second part, these pharmacies are assigned to different days. After all pharmacies with high frequencies are assigned to different days, the assignments of the remaining pharmacies are made.

When the number of inner iterations reach to its maximum level, the new solution heuristic runs and generates a new solution. This solution makes use of the frequencies collected during the inner iterations. If two pharmacies have a tendency to be opened on the same day, they are assigned duties on different days using Algorithm 5.7. Thus, a new and diverse solution is obtained. This new solution enables us to search different parts of the search space.

Algorithm 5.7 New Initial Solution Heuristic for Single Duty Problem

```
maximum_frequency = 1
t1 = 1
t2 = T
while maximum_frequency > 0 do
  maximum_frequency = 0
  for j1 = 1 → J do
    if z_{j1} < 0 then
      for j2 = j1 + 1 → J do
        if z_{j2} < 0 then
          if maximum_frequency < frequency_{j1 j2} then
            maximum_frequency = frequency_{j1 j2}
          end if
        end if
      end for
    end if
  end for
  if maximum_frequency > 0 then
    z_{j1} = t1
    Y_{j1 t1} = 1
    t1 = t1 + 1
    if t1 > T/2 then
      t1 = 1
    end if
    z_{j2} = t2
    Y_{j2 t2} = 1
    t2 = t2 − 1
    if t1 ≤ T/2 then
      t2 = T
    end if
  end if
end while
```

We use maximum iteration numbers and the number of iterations in which no improvements were made as the stopping criteria. The maximum iteration numbers limit both the inner and the outer iterations. Thus, the algorithm stops after a predefined number of iterations are made. Secondly, the algorithm stops if no improvements on the objective value are made in a predefined number of iterations. We use this type of stopping condition for the inner iterations, so that, the algorithm may skip non-improving directions and move to other parts of the search space. After the stopping conditions are met, the Tabu Search algorithm stops and reports the best solution found.

5.1.2.2 Tabu Search Algorithm for Multiple Duty Problem

For the multiple duty problem, the logic of the Tabu Search algorithm uses the same structure with the single duty algorithm. However, some changes need to be done in order to reflect the regions and the multiple duties for the pharmacies. The pseudocode of the Tabu Search algorithm for multiple duty problem is stated as Algorithm 5.8.

In the tabu search algorithm for the multiple duty problem, we make use of *swap* and *add-drop* neighborhood search techniques. The swap algorithm, (Algorithm 5.9), again looks for possible swaps for a pharmacy but the allowable swaps are only with the pharmacies within the same region. Since, there is only one on duty pharmacy in each region on each day of the planning horizon,

Algorithm 5.8 Tabu Search Heuristic for Multiple Duty Problem

```
while outer_iteration < maximum_outer_iteration do
  Reset tabu status
  if outer_iteration = 1 then
    Generate initial solution
  else
    Generate new initial solution
  end if
  while inner_iteration < maximum_inner_iteration do
    Search best swap
    Search best add-drop
    if F_best_cand_swap < F_best_cand_add−drop then
      Apply best swap
    else
      Apply best add-drop
    end if
    Update frequencies
    Update tabu status
    if F_current < F_best then
      Update best solution
    end if
  end while
end while
```

the solution would become infeasible if a pharmacy is swapped with another pharmacy from a different region. Thus, the algorithm only looks for swaps between the pharmacies within the same region.

Algorithm 5.9 Swap Heuristic for Multiple Duty Problem

```
for j1 = 1 → J do
  for t1 = 1 → T do
    if Y_{j1 t1} = 1 then
      for j2 = j1 + 1 → J do
        if region_{j1} = region_{j2} then
          for t2 = 1 → T do
            if Y_{j2 t2} = 1 & t1 ≠ t2 then
              Y_{j1 t1} = 0
              Y_{j1 t2} = 1
              Y_{j2 t1} = 1
              Y_{j2 t2} = 0
              Compute candidate solution cost
              if F_{cand_swap} < F_{best_cand_swap} then
                if j1, j2, t1, t2 not tabu then
                  Update best candidate swap
                else
                  if F_{cand_swap} < F_{best} then
                    Update best candidate swap
                  end if
                end if
              end if
              Y_{j1 t1} = 1
              Y_{j1 t2} = 0
              Y_{j2 t1} = 0
              Y_{j2 t2} = 1
            end if
          end for
        end if
      end for
    end if
  end for
end for
```

The same logic is valid for the add-drop algorithm. In the simple duty problem, we introduced the move search method which is inapplicable for the multiple duty problem. In the multiple duty problem, if we move a pharmacy from one day to another then there will be no on duty pharmacy on the day the pharmacy was initially assigned and two on duty pharmacies on the moved day. Thus, this causes infeasibility of the solutions. For this reason, we move one step ahead and introduce the add-drop algorithm (Algorithm 5.10).

For the add-drop algorithm, we calculate the upper and lower bounds for the duty numbers for each region. The average number of duties for pharmacies during the planning horizon within a region is simply calculated by dividing the number of days in a planning horizon by the number of pharmacies in the region. This number gives the average number of duties to be assigned to each pharmacy in the region. However, sometimes this number may not be an integer. Thus, the duty numbers of pharmacies within a region may be different from each other. The upper and lower bounds of duty numbers are calculated by rounding the average duty number up and down unless it is an integer value.

The duty number upper bounds and lower bounds of the pharmacies must be different from each other in order to apply the add-drop algorithm. If these values are equal, then, it means that the selected pharmacy is assigned exactly the number of duties that it should take. In order to add a duty for a pharmacy, it should be assigned duties on its duty number lower bound. Otherwise, the

maximum number of duties to be assigned will be exceeded for the selected pharmacy and the problem will become infeasible. Likewise, in order to drop a duty from a pharmacy, it should be assigned duties on its duty number lower bound. Otherwise, the pharmacy will be assigned less duties than the minimum number allowed and the problem will again become infeasible.

In the add-drop algorithm, a pharmacy is chosen. Then, the algorithm tests if the solution quality improves, when another duty is assigned to the selected pharmacy on any other day of the planning horizon. The new duty is assigned to the pharmacy unless the maximum number of duties that can be assigned to a pharmacy is exceeded. If this number is not exceeded, the new duty is assigned to the pharmacy and all the existing duties remains the same. If the pharmacy is assigned a duty on e.g. day t_1, then the duty of the pharmacy from its region on day t_1 is canceled. In this way, a duty is added for the selected pharmacy and the duty of another one is dropped within the same region.

For the single duty problem, we keep tabu list for days and the pharmacies. In the multiple duty problem, we keep a tabu list for the regions, additionally. If a change is made within a region during an iteration, then no changes are allowed for that region for a number of iterations. The number of iteration in which no changes are allowed is equal to the tabu tenure for regions. The same aspiration criterion with the single duty problem is used for the multiple duty problem, too. A solution is accepted as admissible if it is better than the best known solution

Algorithm 5.10 Add-Drop Heuristic for Multiple Duty Problem

```
for j_1 = 1 → J do
  k = region_{j_1}
  if duty_number_LB_{j_1} < duty_number_UB_{j_1} and duty_numberj_1 =
  duty_number_UBj_1 then
    for t = 1 → T do
      if Y_{j_1 t} = 1 then
        for j_2 = 1 → J do
          if region_{j_2} = k and duty_numberj_2 < duty_number_UBj_2 then
            Y_{j_1 t} = 0
            Y_{j_2 t} = 1
            Compute candidate solution cost
            if F_{cand_add−drop} < F_{best_cand_add−drop} then
              if j_1, j_2, t not tabu then
                Update best candidate add-drop
              end if
              if j_1, j_2 or t is tabu and aspiration criteria is met then
                Update best candidate add-drop
              end if
            end if
          end if
        end for
      end if
    end for
  end if
end for
```

even if the selected solution is tabu.

During each inner iteration of the algorithm, the best candidate swap and add-drop solutions are determined. Then, these two solutions are compared and the best one among them is chosen and it is applied. This solution becomes the initial solution for the next inner iteration and the search process continues within the neighborhood of this solution.

After the maximum number of inner iterations is reached for each outer iteration a new initial solution is formed. During the inner iterations, the frequencies of the pharmacies that are opened on the same day are kept. The new initial solution algorithm for the multiple duty problem, (Algorithm 5.11), makes use of these frequencies and assign the pharmacies that tend to be opened on the same day frequently to different days of the planning horizon.

In this algorithm, we partition the planning horizon into two parts. For a period of length T, these two partitions are $[1, T/2)$ and $(T/2, T]$. We make use of these partitions while assigning duty dates to the pharmacies with high frequencies. By the use of partitioning, the pharmacies with high frequencies are assigned to different parts of the planning horizon and it is not possible that they are assigned duties on the same day.

Algorithm 5.11 New Initial Solution Heuristic for Multiple Duty Problem

$t_1 = 1$
$t_2 = T$
while $maximum\_frequency > 0$ **do**
  $maximum\_frequency = 0$
  for $j_1 = 1 \rightarrow J$ **do**
    if $z_{j_1} < 0$ **then**
      for $j_2 = j_1 + 1 \rightarrow J$ **do**
        if $z_{j_2} < 0$ **then**
          if $maximum\_frequency < frequency_{j_1 j_2}$ **then**
            $maximum\_frequency = frequency_{j_1 j_2}$
            $maxj_1 = j_1$
            $maxj_2 = j_2$
          end if
        end if
      end for
    end if
  end for
  if $maximum\_frequency > 0$ **then**
    $z_{maxj_1} = t_1$
    $Y_{j_1 t_1} = 1$
    $t_1 = t_1 + 1$
    if $t_1 > T/2$ **then**
      $t_1 = 1$
    end if
    $z_{j_2} = t_2$
    $Y_{j_2 t_2} = 1$
    $t_2 = t_2 - 1$
    if $t_1 \leq T/2$ **then**
      $t_2 = T$
    end if
  end if
end while

5.2 Lower Bounds

We need lower bounds in order to test the performance of the Tabu Search algorithms. With the help of lower bounds, the gap between the lower bound and the upper bound may be found and that could give an idea about how far the solution found is from the optimal solution. In this section, the lower bound algorithms are introduced. In the first part, Lagrangean Relaxation algorithm is introduced. Then, in the following section different lower bound algorithms are proposed for the single duty and the multiple duty problems.

5.2.1 Lagrangian Relaxation Algorithm

In general, the complexity of the problems are caused by the constraints. By use of Lagrangian relaxation, some of these constraints may be relaxed. The relaxed constraints are reflected in the objective function with a penalty term. Thus, the amount of violation is penalized with an increase (in minimization problems) or decrease (in maximization problems) in the objective value. We use Lagrangian relaxation in order to obtain lower bounds for the models developed.

5.2.1.1 Application of Lagrangian Relaxation to Model 1

Model 1 is developed for single duty problem in Section 4.1.1. The model is as follows:

Minimize

$$F = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{t=1}^{T} h_i d_{ij} X_{ijt}$$

Subject to

$$X_{ijt} \leq Y_{jt} \qquad \forall i, j, t \tag{5.1}$$

$$\sum_{t=1}^{T} Y_{jt} \leq 1 \qquad \forall j \tag{5.2}$$

$$\sum_{j=1}^{J} X_{ijt} \geq 1 \qquad \forall i, t \tag{5.3}$$

$$Y_{jt} \in \{0, 1\} \qquad \forall j, t$$

$$X_{ijt} \in \{0, 1\} \qquad \forall i, j, t$$

In this model, there are three possible relaxations. These are relaxing constraint sets (5.1), (5.2) or (5.3), respectively. When constraint set (5.1) is relaxed, customers may be assigned to unopened facilities. Relaxing constraint set (5.2) may cause some facilities to be opened more than once during the planning horizon. Finally, when constraint set (5.3) is relaxed, not all customers are necessarily assigned to a facility each day of the planning horizon. For each of these

relaxations, penalty terms are defined in order to penalize the violations in the objective function.

Sections 5.2.1.1.1, 5.2.1.1.2 and 5.2.1.1.3, represent relaxations of constraint sets (5.1), (5.2) and (5.3), respectively.

5.2.1.1.1 Relaxation of Constraint Set (5.1)**:** Relaxation 1 is obtained by relaxing the constraint set that ensures no customer is assigned to an unopened facility.

$$X_{ijt} \leq Y_{jt} \qquad \forall i, j, t$$

When this constraint set is relaxed, the following model is obtained:

Minimize

$$F = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T} h_i d_{ij} X_{ijt} + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T} \lambda_{ijt}(X_{ijt} - Y_{jt}) \tag{5.4}$$

Subject to

$$\sum_{j=1}^{J} X_{ijt} \geq 1 \qquad \forall i, t \tag{5.5}$$

$$\sum_{t=1}^{T} Y_{jt} \leq 1 \qquad \forall i, t \tag{5.6}$$

$$Y_{jt} \in \{0, 1\} \qquad \forall j, t \tag{5.7}$$

$$X_{ijt} \geq 0 \qquad \forall i, j, t \tag{5.8}$$

By relaxing this constraint, assignment of customers to unopened facilities is allowed. By arranging the terms in (5.4), the revised objective function (5.9) is obtained.

$$F = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T}(h_i d_{ij} + \lambda_{ijt})X_{ijt} - \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T}\lambda_{ijt}Y_{jt} \tag{5.9}$$

This revised version of the objective function (5.4), has two separate components ,n the objective function and the constraints that are related with the decision variables X_{ijt} and Y_{jt}, respectively. Thus, this model decomposes into two subproblems which can be solved separately.

Subproblem 1: P1

Minimize

$$F_1 = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T}(h_i d_{ij} + \lambda_{ijt})X_{ijt} \tag{5.10}$$

Subject to

$$\sum_{j=1}^{J} X_{ijt} \geq 1 \qquad \forall i, t \tag{5.11}$$

$$X_{ijt} \geq 0 \qquad \forall i, j, t \tag{5.12}$$

Subproblem 2: P2

Maximize

$$F_2 = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{t=1}^{T} \lambda_{ijt} Y_{jt} \tag{5.13}$$

Subject to

$$\sum_{t=1}^{T} Y_{jt} \leq 1 \qquad \forall i, t \tag{5.14}$$

$$Y_{jt} \in \{0, 1\} \qquad \forall j, t \tag{5.15}$$

After this decomposition, each subproblem may further be decomposed into sub-subproblems. Subproblem 1 may be decomposed into (I) sub-subproblems each one representing an (i, t) pair. Similarly, subproblem 2 may be decomposed into J sub-subproblems.

The related decomposition of subproblem 1 is as follows:

Sub-subproblem **P1**(i, t)

Minimize

$$F_{1it} = \sum_{j=1}^{J} (h_i d_{ij} + \lambda_{ijt}) X_{ijt} \tag{5.16}$$

Subject to

$$\sum_{j=1}^{J} X_{ijt} \geq 1 \tag{5.17}$$

$$X_{ijt} \geq 0 \tag{5.18}$$

Let

$$c_j = \sum_{i=1}^{I} \sum_{t=1}^{T} h_i d_{ij} + \lambda_{ijt}$$

Then subproblem 1 can be rewritten as follows for a given (i, t) pair:

Minimize

$$F_{1it} = \sum_{j=1}^{J} c_j X_j$$

Subject to

$$\sum_{j=1}^{J} X_j \geq 1$$

$$X_j \geq 0$$

The decision here is to set $X_{j^*} = 1$ such that $c_{j^*} = min\{c_j\}$ and all other $X_j = 0$.

The further decomposition of subproblem 2 is as follows:

Sub-subproblem **P2(j)**

Maximize

$$F_{2j} = \sum_{t=1}^{T} \left\{ \sum_{i=1}^{I} \lambda_{ijt} \right\} Y_{jt} \tag{5.19}$$

Subject to

$$\sum_{t=1}^{T} Y_{jt} \leq 1 \qquad \forall i, t \tag{5.20}$$

$$Y_{jt} \in \{0, 1\} \qquad \forall j, t \tag{5.21}$$

Let

$$c_t = \sum_{i=1}^{I} \lambda_{ijt}$$

Then the model is revised as

Maximize

$$F_{2j} = \sum_{t=1}^{T} c_t Y_{jt} \tag{5.22}$$

Subject to

$$\sum_{t=1}^{T} Y_t \leq 1$$

$$Y_t \in \{0, 1\}$$

The solution is trivial, set $Y_{t^*} = 1$ such that $c_{t^*} = min\{c_t\}$ and all other $Y_t = 0$.

5.2.1.1.2 Relaxation of Constraint Set (5.2)**:** Relaxation 2 is obtained by relaxing constraint set (5.2) in Model 1. These constraints state that each facility must be opened only once during the planning horizon. Relaxation of this constraint set causes some facilities to be opened more than once.

$$\sum_{t=1}^{T} Y_{jt} \leq 1 \qquad \forall j$$

If the constraint set above is relaxed, the following model is obtained:

Minimize

$$F = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T} h_i d_{ij} X_{ijt} + \sum_{j} \lambda_j (1 - \sum_{t=1}^{T} Y_{jt}) \tag{5.23}$$

Subject to

$$X_{ijt} \leq Y_{jt} \qquad \forall i, j, t \tag{5.24}$$

(5.17), (5.21) and (5.18)

5.2.1.1.3 Relaxation of Constraint Set (5.3)**:** Relaxation 3 is obtained by relaxing constraint set (5.3) from Model 1. This constraint set makes sure that each customer is assigned to at least one facility each day of the planning horizon. Some customers will not be assigned to any facility in different days by the relaxation of this constraint set. The model is as follows:

Minimize

$$F = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T} h_i d_{ij} X_{ijt} + \sum_{i=1}^{I}\sum_{t=1}^{T} \lambda_{it} (1 - \sum_{j=1}^{J} X_{ijt}) \tag{5.25}$$

Subject to (5.20), (5.21), (5.18) and (5.24)

We have proved that relaxation of constraint sets 1 and 3 are unimodular. Thus, these relaxations will produce lower bounds equal to linear relaxation at best. The relaxation of constraint set 2 is not unimodular and with the optimal λ_j values it may produce lower bounds better than that of LP relaxation. However, during the preliminary experiments, it was not possible to get promising

results within reasonable computation times for relaxation 2. For this reason, Lagrangean Relaxation is not used for large-size problems and not included in the computational results.

5.2.2 AO_S Lower Bound Algorithm for the Single Duty Problem

In this section, the lower bound algorithm for the single duty problem is introduced (Algorithm 5.12). In this algorithm, a customer is chosen at first. Then, the closest pharmacy is found for the customer and the customer is assigned to the closest pharmacy for each day of the planning horizon. Since, each pharmacy may take only one duty, each customer may be assigned to the pharmacy that is closest to them only once. For the following days of the planning period, customers are assigned to the second closest pharmacy, to the third closest pharmacy and so on.

Algorithm 5.12 AO_S Lower Bound Algorithm for Single Duty Problem

$LB = 0$
for $i = 1 \rightarrow I$ **do**
  for $t = 1 \rightarrow T$ **do**
    $minimum\_distance = M$
    for $j = 1 \rightarrow J$ **do**
      if $d_{ij} < minimum\_distance$ **then**
        $minimum\_distance = d_{ij}$
        $min\_j = j$
      end if
    end for
    $LB = LB + minimum\_distance \times h_i$
    $d_{i,min\_j} = M$
  end for
end for

In the scope of this algorithm, it is not required that a pharmacy is opened in order a customer to be assigned to it. Since, this algorithm just tries to find a lower bound, the aim is to find a solution that the real solution cannot be better off. Likewise, we could have relaxed the constraint that each pharmacy can only be opened once and assign each customer to their closest facility for all days of the planning horizon. When the cost of these assignments were computed, we would again obtain a lower bound value, however, that lower bound would not be tight enough. During the test runs, we observed that the AO_S Lower Bound Algorithm gives exactly the same lower bound value with the LP relaxation within much less computation time.

5.2.3 AO_M1 Lower Bound Algorithm for the Multiple Duty Problem

The AO_M1 lower bound algorithm, (Algorithm 5.13), is introduced for the multiple duty problem. This algorithm works with a similar logic with Algorithm 5.12. The difference is that the pharmacies may take more than one duty during the planning period in the multiple duty problem. In this case, the customers may be assigned to their closest pharmacies more than once. The number of this assignment is actually equal to the number of duty days assigned to that pharmacy. Thus, the minimum and maximum duty numbers for each pharmacy are calculated with regard to their regions. Thus, when the closest pharmacy to a customer is found, the customer is assigned to the closest pharmacy for the

maximum number of days the pharmacy can take duties. Similarly, the cost of assignments is calculated by taking the maximum number of duties into account.

Algorithm 5.13 AO_M1 Lower Bound Algorithm for Multiple Duty Problem

```
LB = 0
for i = 1 → I do
  t = 0
  while t ≤ T do
    minimum_distance = M
    for j = 1 → J do
      if d_ij < minimum_distance then
        minimum_distance = d_ij
        min_j = j
      end if
    end for
    duty_number = duty_number_UB_min_j
    if T − t ≤ duty_number then
      duty_number = T − t
    end if
    LB = LB + minimum_distance × h_i × duty_number
    d_ij = M
    t = t + duty − number
  end while
end for
```

However, this lower bound algorithm is not producing tight lower bounds, thus, we move a step ahead and introduce the AO_M2 Lower Bound Algorithm for the multiple duty problem.

5.2.4 AO_M2 Lower Bound Algorithm for the Multiple Duty Problem

In the real case, not all pharmacies take same number of duties even if they are within the same region. Some of them are assigned minimum number of duties

for their regions while the others are assigned the maximum number of duties. So, a better lower bound algorithm needed to reflect this characteristic of the real problem. Thus, we introduce AO_M2 Lower Bound Algorithm for the multiple duty problem (Algorithm 5.14).

Algorithm 5.14 AO_M2 Lower Bound Algorithm for Multiple Duty Problem

```
for i = 1 → I do
  t = 0
  while t ≤ T do
    minimum_distance = M
    for j = 1 → J do
      if d_ij < minimum_distance then
        minimum_distance = d_ij
        min_j = j
      end if
    end for
    duty_number = duty_number_LB_min_j
    if T − t ≤ duty_number then
      duty_number = T − t
    end if
    LB = LB + minimum_distance × h_i × duty_number
    d_ij = M
    t = t + duty_number
  end while
end for
```

In this algorithm, for each of the regions, the number of pharmacies that should take the maximum and minimum number of duties are calculated. Then the maximum number of duties are assigned to the pharmacies that are closest to each customer unless the number of pharmacies that should take maximum number of duties is exceeded. If the number is exceeded, then, the pharmacies are assigned duties on the lower bound. During the test runs, as expected, it is observed that the AO_M2 Lower Bound Algorithm gives better results than

the AO_M1 Lower Bound Algorithm. In other words, it is a more tight bound. However, the LP relaxation in multiple duty case provides better bounds for the problems than the AO_M2 Lower Bound Algorithm. This is simply due to the fact that in the AO_M2 Lower Bound Algorithm, we assign duties to the pharmacies on the upper bound until the limit is reached. On the other hand, in the LP relaxation, the assignments are made while taking costs into account.

Chapter 6

COMPUTATIONAL EXPERIMENTS

In this chapter, we present the computational experiments made for both single duty and multiple duty problems. In each of the sections, first preliminary experiments designed for the problems are explained. Then, the results of different instances generated for the related problems are discussed.

We conducted the computational experiments on a HP Laptop with AMD Triple Core Processor 1.80 GHz, 4GB RAM and Windows 7 Professional. We solve the mathemtatical models using GAMS 22.5 and CPLEX 11.2. We coded Tabu Search and lower bound algorithms using C++ and Codeblocks.

6.1 Single Duty Problem

The single duty problem is the problem type in which the pharmacies are assigned one duty during the planning horizon and similar to the Simple Model 1 from Chapter 4. For the single duty problem, different instances were generated in order to test the performance of the Tabu Search algorithm. Before starting the test runs, we conducted some preliminary experiments in order to find the most effective values for the tabu tenures and the maximum iteration numbers.

During the preliminary experiments, we used 6 different problem sizes. Each of these problem sizes were scaled to the real problem size, so that the test results would give an insight about the large-size real life problem. Two types of parameters were tested during the preliminary experiments. We first tested 3 different levels of tabu tenures while taking the iteration numbers constant. After choosing the most suitable tabu tenure, we tested 3 different levels of iteration numbers using the selected tabu tenure.

In Tabu Search algorithm, tabu tenures are used to represent the number of iterations in which a recently changed solution cannot be visited. For the tabu tenures, we tested three different tenures as 5, 8 and 10 that are commonly used values in the literature [11]. We used constant iteration numbers during these tests. We chose maximum outer iteration number as 10 and maximum inner iteration number as 30. Table 6.1 shows the test results for tabu tenures. In the table, I, J and T denote the numbers of customers, pharmacies and days,

Table 6.1: Tabu Tenure Test Results for Single Duty Problem

I	J	T	In	Out	Day	Node	AO LB	LB CPU	Tabu Search	TS CPU (seconds)	Gap (%)
20	10	5	30	10	5	5	362,789	0.001	363,922	0.205*	0.3*
20	10	5	30	10	8	8	362,789	0.002	363,922	0.213	0.3*
20	10	5	30	10	10	10	362,789	0.001	363,922	0.214	0.3*
20	30	15	30	10	5	5	701,217	0.001	708,271	3.150*	1.0
20	30	15	30	10	8	8	701,217	0.002	709,083	4.218	1.1
20	30	15	30	10	10	10	701,217	0.002	705,964	4.464	0.7*
40	40	10	30	10	5	5	690,491	0.002	693,924	16.992*	0.5*
40	40	10	30	10	8	8	690,491	0.002	693,924	18.035	0.5*
40	40	10	30	10	10	10	690,491	0.002	693,924	18.428	0.5*
40	60	15	30	10	5	5	1,125,694	0.001	1,133,049	30.734*	0.7*
40	60	15	30	10	8	8	1,125,694	0.002	1,134,184	45.184	0.8
40	60	15	30	10	10	10	1,125,694	0.002	1,136,737	47.016	1.0
60	60	10	30	10	5	5	867,355	0.002	874,606	75.324*	0.8*
60	60	10	30	10	8	8	867,355	0.002	874,606	79.111	0.8*
60	60	10	30	10	10	10	867,355	0.002	874,606	81.071	0.8*
60	90	15	30	10	5	5	1,316,550	0.002	1,340,447	139.762*	1.8*
60	90	15	30	10	8	8	1,316,550	0.002	1,342,397	191.316	2.0
60	90	15	30	10	10	10	1,316,550	0.002	1,342,454	208.443	2.0

respectively. These parameters define the problem size. Then the iteration numbers are given under in and out iterations which are constant. The day and node tenures are the tested parameters. *AO_LB* column shows the lower bound values for the instances where *Tabu Search* column denotes the objective value of the Tabu Search algorithm. *LB CPU* and *TS CPU* denote the computation times in seconds for the lower bound algorithm and the Tabu Search algorithm, respectively. Finally, *Gap* denotes the relative gap between the objective value of tabu search algorithm and the lower bound value of the instance. The best results in terms of computation time and objective value can be found in Table 6.1.

After conducting the preliminary experiments on tabu tenures, the results for each tabu tenure were compared according to the computation time and the

Table 6.2: Summary Table for Tabu Tenures

Tenure	**CPU**	**Solution Quality**
5	81%	92%
8	96%	97%
10	100%	94%

gap between the lower bound and the objective value. For each problem size, the minimum TS CPU and Gap values are indicated with a (*). We compute average performance of each tabu tenure with respect to the best results obtained in each problem size. Table 6.2 shows the average CPU times and the gap values of all test instances for each tenure level. The 100% represents the largest values and the other percentages show the values with respect to the largest one. As it can be seen from Table 6.2, tabu tenure 5 was the best among all both for the computation time and the objective value. Thus, we decided to use 5 as the tabu tenure of days and nodes for the single duty problem.

After determining the tabu tenure, the iteration numbers were tested. Again, there were 3 levels of iteration numbers for both inner and outer iterations while taking the day and node tabu tenures as 5. Table 6.3 shows the results of six different instances.

Table 6.4, shows a summary of the preliminary experiments for the iteration numbers. In this table, the largest CPU is represented with 100% and the other ones show the ratio of CPU usage with regard to the CPU usage of 40 and 30

Table 6.3: Iteration Number Test Results for Single Duty Problem

I	J	T	In	Out	Day	Node	AO_LB	LB CPU	Tabu AO	TS CPU (seconds)	Gap (%)
20	10	5	20	10	5	5	362,789	0.001	363,922	0.138*	0.3
20	10	5	30	15	5	5	362,789	0.002	362,789	0.307	0.0*
20	10	5	40	20	5	5	362,789	0.002	363,922	0.563	0.3
20	30	15	20	10	5	5	701,217	0.001	706,261	2.088*	0.7
20	30	15	30	15	5	5	701,217	0.002	708,271	4.783	0.1*
20	30	15	40	20	5	5	701,217	0.002	706,131	8.747	0.7
40	40	10	20	10	5	5	690,491	0.002	694,419	10.426*	0.6
40	40	10	30	15	5	5	690,491	0.002	693,924	25.627	0.5*
40	40	10	40	20	5	5	690,491	0.004	693,924	47.832	0.5*
40	60	15	20	10	5	5	1,125,694	0.002	1,133,049	46.428	0.7
40	60	15	30	15	5	5	1,125,694	0.002	1,333,049	46.376*	18.4
40	60	15	40	20	5	5	1,125,694	0.002	1,132,838	86.573	0.6*
60	60	10	20	10	5	5	867,355	0.002	879,647	47.08*	1.4
60	60	10	30	15	5	5	867,355	0.001	874,606	110.722	0.8*
60	60	10	40	20	5	5	867,355	0.002	874,460	205.192	0.8*
60	90	15	20	10	5	5	1,316,550	0.003	1,371,646	75.364*	4.2
60	90	15	30	15	5	5	1,316,550	0.002	1,340,447	205.685	1.8
60	90	15	40	20	5	5	1,316,550	0.002	1,327,967	328.593	0.9*

iterations for inner and outer loops. As the CPU time increases the solution quality increases since more iterations in general mean more solution space searched. However, for a 4% increase in the solution quality, the CPU times increases more than four times. Thus, we decided to give up some quality in return for decreased CPU time. According to these results, the maximum number of inner iterations is determined as 30 and the maximum number of outer iterations is determined as 15.

After conducting the preliminary experiments, the performance of the tabu search algorithm is tested. In these test runs, ten different instances were generated for each of six different problem sizes. Table 6.5, shows the arithmetic

Table 6.4: Summary of Iteration Number Test Results for Single Duty Problem

In	Out	CPU	Solution Quality
20	10	28%	7%
30	15	55%	4%
40	20	100%	3%

average values of 10 instances for each problem size. In the table, problem size column shows the I, J and T values that refer to customers, pharmacies and days. LB_AO column refers to the average lower bound values for each problem size. Tabu_AO column represents the average results found for each problem size by Tabu Search algorithm. The average lower and upper bound values found with the use mathematical models are given under GAMS LB and GAMS UB columns, respectively. # column shows the number of problem instances solved to optimality out of 10 within one hour limit. Real Gap column shows the relative gap between the GAMS LB and the Tabu_AO. Perceived gap shows the relative gap between the lower bound found by the lower bound algorithm and the Tabu Search algorithm. On the other hand, real gap corresponds to the relative gap between the best known lower bound value and the objective value of the Tabu Search algorithm. LB_AO, Tabu_AO and GAMS CPU columns represent the computation times for the mentioned methods. As expected, the CPU times increase a lot as the problem size gets larger. For the mathematical models, we determined one hour limit for the CPU times. Thus, a CPU time of 3600 seconds states that the problem could not be solved optimally within one hour.

Table 6.5: Average Test Results for Single Duty Problem

I	J	T	LB_AO	LB CPU (seconds)	Tabu AO	Tabu Search CPU (seconds)	GAMS LB	GAMS UB	GAMS CPU (seconds)	#	Real Gap (%)	Perceived Gap (%)
20	10	5	284,841	0.001	285,926	0.299	285,926	285,926	0.740	10	0.0	0.4
20	20	10	552,603	0.001	555,947	1.911	554,673	554,673	22.591	10	0.2	0.6
20	30	15	796,253	0.001	801,372	4.533	799,701	799,982	1856.153	6	0.2	0.7
40	20	5	363,767	0.001	365,558	3.355	365,248	365,248	15.589	10	0.1	0.5
40	40	10	696,906	0.002	701,943	24.097	698,711	699,532	3600.000	0	0.5	0.7
40	60	15	1,005,932	0.001	1,018,604	45.130	1,006,088	1,009,538	3600.000	0	1.2	1.3
60	30	5	411,919	0.001	415,108	13.883	413,440	413,440	183.641	10	0.4	0.8
60	60	10	818,676	0.001	829,189	103.670	819,118	822,345	3600.000	0	1.2	1.3
60	90	15	1,216,310	0.002	1,247,472	165.894	1,216,311	1,237,099	3600.000	0	2.6	2.6

Table 6.6: Average Results for Single Duty Problem

	Real	**Perceived**
Gap	0.717	0.974

	Tabu	**GAMS**
CPU Time	40.308	1830.968

Table 6.6 shows a summary of the gaps and the CPU times. The Tabu Search algorithm mostly produces results within 1% of the optimal solutions as the gaps state. If the average CPU times for the Tabu Search algorithm and the GAMS mathematical models are compared, it is obvious that the Tabu Search algorithm finds results within much less CPU time and these results are not far from the results of the mathematical models.

Figures 6.1, 6.2 and 6.3 show the computation time change with respect to number of customers, length of the planning horizon and problem size, respectively. As the figures show, the length of the planning horizon has serious effect on computation time. Figure 6.3 shows that even with higher number of customers and pharmacies, results are found within shorter computation times when the planning horizon length is shorter.

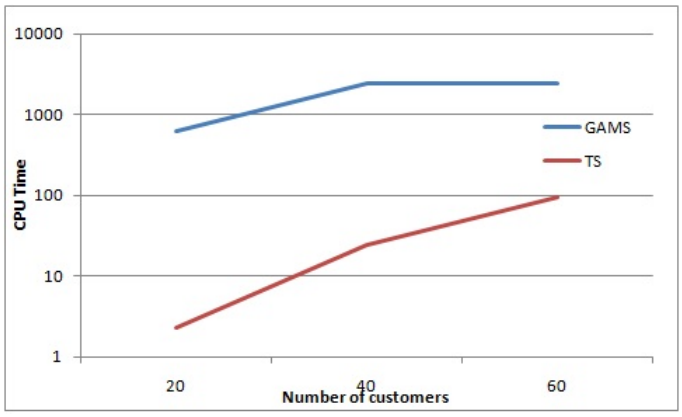


Figure 6.1: CPU Time vs. Number of Customers

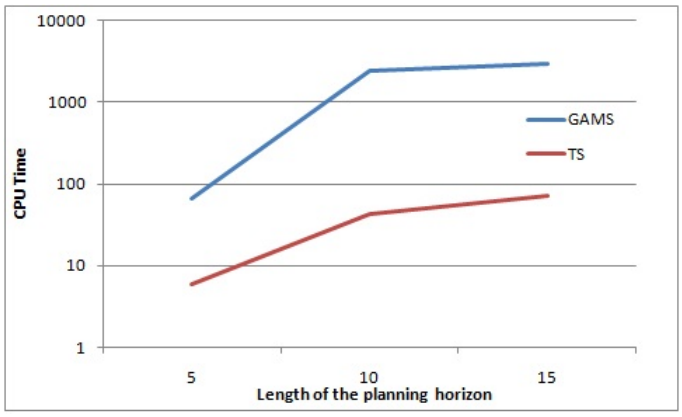


Figure 6.2: CPU Time vs. Length of the Planning Horizon

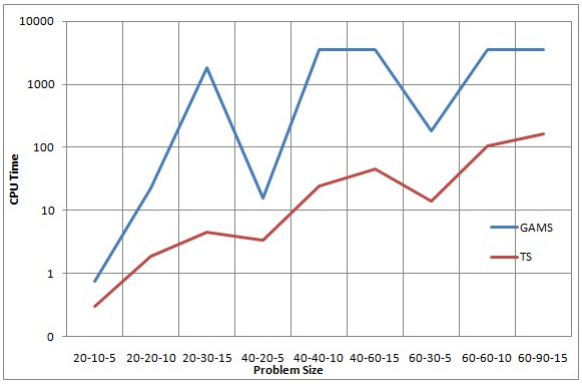


Figure 6.3: CPU Time vs. Problem Size

6.2 Multiple Duty Problem

For the preliminary experiments of the multiple duty problem, we generated 6 different problem instances with different numbers of customers, pharmacies, days and regions. Before the test runs, we conducted preliminary experiments on these instances in order to determine the tabu tenures and the iteration numbers.

For the day, pharmacy and region tabu tenures, we tested four different levels. For these experiments inner and outer iteration numbers are considered constant as 30 and 10, respectively. Table 6.7, shows the results of the preliminary experiments for tabu tenures.

Table 6.8, shows the average values of CPU times and the gaps for the instances used in preliminary experiments. As it can be seen from the table, day tenure 5, node tenure 5 and region tenure 3 combination gives the best solution quality within a tolerable CPU time increase.

Table 6.7: Tabu Tenure Experiments for Multiple Duty Problems

I	J	T	K	In	Out	Day	Node	Region	LB AO	LB CPU (Seconds)	Tabu AO	TS CPU (Seconds)	Gap (%)
20	10	5	4	30	10	5	5	1	265,362	0.001	269,717	1.577	1.6*
20	10	5	4	30	10	5	5	3	265,362	0.001	269,717	1.627	1.6*
20	10	5	4	30	10	8	8	3	265,362	0.001	269,717	1.557*	1.6*
20	10	5	4	30	10	10	10	4	265,362	0.001	269,717	1.606	1.6
40	20	5	4	30	10	5	5	1	309,088	0.001	329,693	2.263	6.7*
40	20	5	4	30	10	5	5	3	309,088	0.001	329,693	2.171	6.7*
40	20	5	4	30	10	8	8	3	309,088	0.001	329,693	2.092*	6.7*
40	20	5	4	30	10	10	10	4	309,088	0.001	332,396	2.166	7.5
40	60	15	9	30	10	5	5	1	655,103	0.001	705,162	36.134	7.6*
40	60	15	9	30	10	5	5	3	655,103	0.001	705,236	36.257	7.7
40	60	15	9	30	10	8	8	3	655,103	0.001	705,236	37.370	7.7
40	60	15	9	30	10	10	10	4	655,103	0.001	705,236	34.332*	7.7
60	30	5	9	30	10	5	5	1	323,115	0.001	342,853	5.701	6.1*
60	30	5	9	30	10	5	5	3	323,115	0.001	342,969	5.584	6.1*
60	30	5	9	30	10	8	8	3	323,115	0.001	343,830	5.256	6.4
60	30	5	9	30	10	10	10	4	323,115	0.001	345,186	5.213*	6.8
60	60	10	9	30	10	5	5	1	658,016	0.001	742,222	26.665	12.8*
60	60	10	9	30	10	5	5	3	658,016	0.001	742,222	26.371	12.8*
60	60	10	9	30	10	8	8	3	658,016	0.001	742,222	25.900*	12.8*
60	60	10	9	30	10	10	10	4	658,016	0.001	742,222	26.571	12.8
60	90	15	9	30	10	5	5	1	934,230	0.002	1,038,933	79.854	11.2
60	90	15	9	30	10	5	5	3	934,230	0.001	1,035,723	79.919	10.9*
60	90	15	9	30	10	8	8	3	934,230	0.002	1,039,818	78.851*	11.3
60	90	15	9	30	10	10	10	4	934,230	0.002	1,039,818	81.569	11.3

Table 6.8: Summary of Tabu Tenure Preliminary Experiments for Multiple Duty Problem

Day	**Node**	**Region**	**CPU**	**Solution Quality**
5	5	1	98.6%	96.1%
5	5	3	98.0%	95.7%
8	8	3	95.7%	97.0%
10	10	4	95.7%	100.0%

After analyzing the tabu tenures, the iteration numbers are tested with the selected tabu tenures. The same instances are also used for iteration number tests. Three levels of inner and outer iteration number combinations are tested. These combinations are (20,10), (30,15) and (40,20). Table 6.9 shows the results for iteration number tests.

Table 6.10 shows the total CPU times and the gaps for the tested instances. Iteration number set 40 and 20 gives the best result quality, however, the CPU times are four times higher than the iteration number set 20 and 10. When the solution qualities of these two iteration number sets are compared, it is seen that iteration number set 20 and 10 gives solutions nearly as good as the iteration number set 40 and 20 within much less CPU time. Thus, iteration number set 20 and 10 is chosen for multiple duty problems.

We separated the multiple duty problem into two parts as the small-size and the large-size problems. In the following two sections, we discuss the computational results for the small-size problems. In the second one, we introduce the

Table 6.9: Iteration Number Experiments for Multiple Duty Problems

I	J	T	K	In	Out	Day	Node	Region	LB_AO	LB CPU (seconds)	Tabu AO	TS CPU (seconds)	Gap (%)
20	10	5	4	20	10	8	8	3	265,362	0.001	269,717	1.218*	1.6*
20	10	5	4	30	15	8	8	3	265,362	0.001	269,717	2.359	1.6*
20	10	5	4	40	20	8	8	3	265,362	0.001	269,717	4.918	1.6*
40	20	5	4	20	10	8	8	3	309,088	0.001	329,693	1.499*	6.7*
40	20	5	4	30	15	8	8	3	309,088	0.001	329,693	3.305	6.7*
40	20	5	4	40	20	8	8	3	309,088	0.001	329,693	5.731	6.7*
40	60	15	9	20	10	8	8	3	655,103	0.001	710,377	24.839*	8.4
40	60	15	9	30	15	8	8	3	655,103	0.001	705,236	55.986	7.7
40	60	15	9	40	20	8	8	3	655,103	0.001	703,507	98.308	7.4*
60	30	5	9	20	10	8	8	3	323,115	0.001	345,186	3.714*	6.8
60	30	5	9	30	15	8	8	3	323,115	0.001	343,830	8.299	6.4*
60	30	5	9	40	20	8	8	3	323,115	0.001	344,561	14.663	6.6
60	60	10	9	20	10	8	8	3	658,016	0.001	748,930	17.837*	13.8
60	60	10	9	30	15	8	8	3	658,016	0.001	742,222	39.498	12.8
60	60	10	9	40	20	8	8	3	658,016	0.001	741,404	70.461	12.7*
60	90	15	9	20	10	8	8	3	934,230	0.002	1,046,746	53.724*	12.0
60	90	15	9	30	15	8	8	3	934,230	0.002	1,039,818	119.049	11.3
60	90	15	9	40	20	8	8	3	934,230	0.002	1,033,323	211.500	10.6*

Table 6.10: Summary of Iteration Number Preliminary Experiments for Multiple Duty Problem

In	Out	CPU	Solution Quality
20	10	25.37%	100.00%
30	15	55.25%	95.17%
40	20	100.00%	94.09%

modifications made in the Tabu Search algorithm in order to solve the large-size problems and discuss the computational results.

6.2.1 Small-Size Problems

For the small-size problems, we defined 9 different problem sizes. For each problem size, we generated 10 instances and compared the results in terms of computation times and the gap between the lower bound and the optimal values if available or with the upper bounds if the optimal value could not be obtained. Table 6.11 shows the average of the results and the CPU times of ten instances for each problem size. In the table, lower bound algorithm values, GAMS lower bound and the LP relaxation values are presented. Alsa Tabu Search algorithm objective value and the GAMS upper bound value are given. The # column shows the number of problem instances solved to optimality within 3600 seconds of computation limit. The LB, TS, OPT and LP CPUs refer to the computation times of the lower bound algorithm, Tabu Search algorithm, mathematical model and LP relaxation, respectively. As the problem size gets larger, the CPU

times increase. However, even for larger problem sizes, the proposed Tabu Search algorithm finds results within much less computation times.

Table 6.12 shows a summary of average gap and CPU time values. The Tabu Search algorithm is capable of finding acceptable results in very short computation times.

Figures 6.4, 6.5 and 6.6 show the computation time change with respect to number of customers, number of pharmacies and number of days in the planning horizon. The length of the planning horizon has a major effect on difficulty of the problem for the multiple duty problem, too.

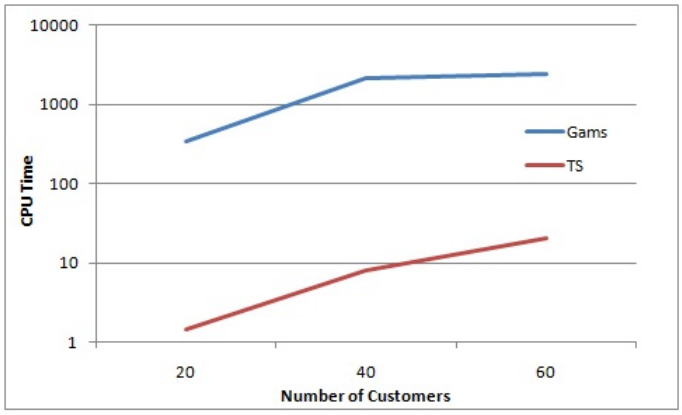


Figure 6.4: CPU Time vs. Number of Customers

6.2.2 Large-Size Problems

For the large-size problems, we made some modifications on the Tabu Search algorithm. In the actual algorithm, the neighborhood searches are being done for

Table 6.11: Average Results for Small-Size Multiple Duty Problems

I	J	T	K	LB_AO	LB CPU (seconds)	Tabu AO	Tabu CPU (seconds)	GAMS LB	GAMS UB	OPT CPU (seconds)		GAMS LP	LP CPU (seconds)	Real Gap (%)	Perceived Gap (%)
20	10	5	4	192,658	0.001	200,439	0.066	200,439	200,439	0.387	10	200,176	0.249	0.0	4.3
20	20	10	4	360,541	0.001	385,194	0.651	384,808	384,808	226.578	10	382,415	0.789	0.1	7.2
20	30	15	9	350,721	0.001	356,219	3.663	355,317	355,348	806.803	8	354,760	1.241	0.2	1.5
40	20	5	4	324,884	0.001	352,368	0.376	351,624	351,624	7.385	10	349,400	0.663	0.2	8.6
40	40	10	9	419,670	0.001	443,064	5.222	439,344	439,576	2758.768	4	437,857	4.850	0.9	5.7
40	60	15	9	632,962	0.001	676,561	18.444	663,621	666,049	3601.574	0	663,149	27.124	2.0	7.0
60	30	5	9	314,801	0.001	337,406	1.514	337,240	337,240	23.863	10	336,252	1.155	0.0	7.3
60	60	10	9	606,741	0.001	683,935	13.486	671,132	674,575	3601.745	0	669,585	38.010	1.9	12.8
60	90	15	9	903,807	0.002	1,033,785	46.058	1,001,159	1,025,069	3603.627	0	1,001,081	252.658	3.3	14.6

Table 6.12: Average Results for Small-Size Multiple Duty Problem

	Real	**Perceived**
Gap	0.961	7.667

	Tabu	**GAMS**
CPU Time	9.942	1625.637

Figure 6.5: CPU Time vs. Length of the Planning Horizon

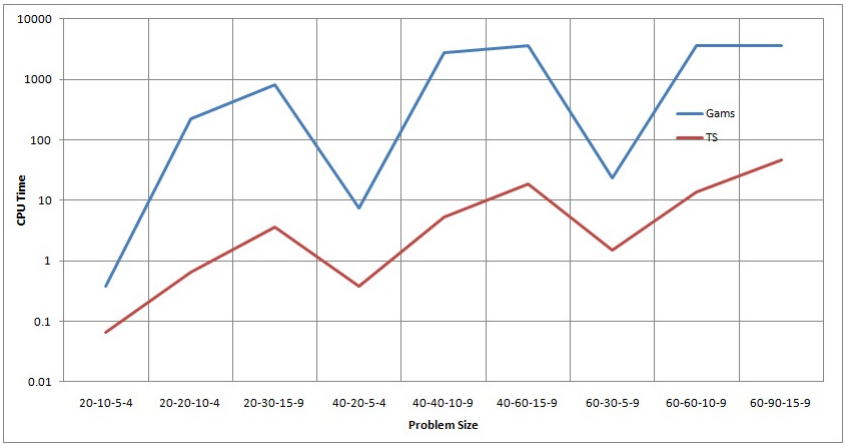


Figure 6.6: CPU Time vs. Problem Size

each pharmacy in all neighborhood search techniques. For large-size problems, since the number of pharmacies are huge, it becomes very hard to obtain results in reasonable computation times.

As the first modification, we started to count the number of iterations in which an improvement on the best candidate solution is made. When this number reaches to the predefined level, we stop searching better results for the selected pharmacy and continue with the next one. In this way, we limit the search done for a pharmacy and do not try to find the best candidate solution. We just find a solution that is good enough and look for different solutions by searching other pharmacies.

As another modification, we make use of the regional setting of the multiple duty problem. In the modified tabu search algorithm, we relate the regions to be searched for with the outer iteration numbers. In this way, every outer iteration starts from a different set of pharmacies, not from the beginning every time. Thus, we can search a broader space for better solutions. For this reason, we change the maximum inner iteration number to 45, so that, in each iteration a region can be searched.

The test results for large-size problems are given in Table 6.13. The optimal values and the LP relaxation values are not available for the large-size problems, since these are hard to obtain in reasonable times. Thus, in this part of the study, we compare the lower bound values from Algorithm AO_M2 and the upper bound

Table 6.13: Results for Large-Size Multiple Duty Problems

I	J	T	K	LB_AO	LB CPU (second)	Tabu AO	TS CPU (second)	Perceived Gap (%)
350	250	25	49	4,185,525	0.008	4,604,968	17,180	10.0
350	250	25	49	4,281,480	0.008	4,727,774	17,194	10.4
350	250	25	49	3,888,865	0.008	4,339,607	17,326	11.6
350	250	25	49	3,946,011	0.008	4,355,554	16,939	10.4
350	250	25	49	4,026,870	0.009	4,411,435	16,632	9.5
350	500	50	49	8,115,930	0.021	9,555,180	120,159	17.7
350	500	50	49	7,471,040	0.020	9,012,111	134,734	20.6
350	500	50	49	7,808,867	0.017	9,123,935	76,620	16.8
350	500	50	49	7,496,463	0.016	8,649,854	75,793	15.4
350	500	50	49	7,471,051	0.016	8,900,553	82,272	19.1
350	1000	100	49	11,780,551	0.032	17,626,749	540,240	49.6
350	1000	100	49	11,500,085	0.016	17,357,072	587,043	50.9
350	1000	100	49	12,711,395	0.031	17,868,820	537,777	40.6
350	1000	100	49	11,373,180	0.031	17,275,659	540,238	51.9
350	1000	100	49	12,263,164	0.016	18,124,038	576,574	47.8

values from the Tabu Search algorithm and report the relative gap between these two results.

As the table shows, the relative gap between the lower bound and the tabu search objectives are very high even with long computation time. We believe that this is due to using randomly generated data.

Chapter 7

İZMİR APPLICATION

We chose İzmir to implement the proposed method and compare the current and the proposed solutions. Thus, in this chapter, we introduce the related information about İzmir.

7.1 Data Gathering and Visualization

After making test runs using the randomly generated data sets, we collected the real data of pharmacies in the metropolitan area of İzmir, the regions of the pharmacies and the duty dates of these pharmacies from the Chamber. We obtained the pharmacy and duty lists for two planning periods, namely, the third period of 2010 and the first period of 2011.

For the given pharmacies, we obtained the coordinates from the geographic

information system[1] of İzmir Municipality[2] in order to locate the pharmacies.

We located the demand nodes as the centers of the districts and the demand sizes as the population of the districts. We collected the coordinates of the districts from the geographical information system of the Municipality and we bought the population data from Turkish Statistical Institute[3].

After collecting the coordinates for the pharmacies, we updated the collected data by adding the pharmacies that are necessary but not included in GIS database of the Municipality, taking out the pharmacies that are not taking duties and the districts that are not in the metropolitan area of İzmir. We created a full list of potential on duty pharmacies and their coordinates. We also made a list of all districts and their coordinates. Also, we store the number of pharmacies in each region and calculate the related upper and lower bounds on the number of duties for the pharmacies using these numbers.

Then, we transfered the printed data, such as the duty lists for different periods, to the computer in order to use in calculations of the current duty plan's objective function value so that we can make comparisons between the current and the proposed solutions.

In order to calculate the costs of the assignments, we needed the locations of the pharmacies. Thus, we had a geographical information system developed

[1] http://cbs.izmir.bel.tr/CbsWebServisleri/CbsLoadStone/download/download.aspx?Id=111
[2] www.izmir.bel.tr/
[3] www.tuik.gov.tr

for this study. New pharmacies can be added and the unnecessary ones can be deleted from the database using the interface of the GIS. We used the GIS in order to obtain the distances between the pharmacies and the districts. Thus, for each of the two periods, two distance matrices are formed. First, the district-pharmacy distance matrix is formed in order to calculate the costs of the assignments made. Secondly, the pharmacy-pharmacy distance matrix is formed to use in new region generation.

In addition to data collection, we make use of the GIS in visualization of the results. The system is capable of showing the locations of the pharmacies and the districts on a map. Also, we can see a given solution on the map with the help of the GIS. We can also use these visualizations while comparing the current and proposed solutions.

7.2 Comparison of Current and Proposed Solutions

In this section of the thesis, we present the comparison of the current and the proposed solutions for two periods, namely, the third period of 2010 and the first period of 2011.

Table 7.1: Comparison of Results for the 3^{rd} Period of 2010

	Cost (demand-meter)	CPU (seconds)	Gap*
Current	309,011,788,792	—	9.4%
Tabu Search	305,873,089,207	36,624.911	8.2%
AO_M2 LB	282,589,154,945	0.085	—
AO_M1 LB	274,449,079,588	0.080	—

*: Relative gap using AO_M2 LB

7.2.1 Real Instance 1: 3^{rd} Period of 2010

For the third period of 2010, there were 1046 pharmacies taking duties. We calculated the cost of the current assignments using our cost algorithm taking the cost of an assignment as the demand weighted distance between the customer and the pharmacy it is assigned to. We compare the current cost with the assignments we make with the use of the Tabu Search algorithm. Table 7.1 shows the two results with the lower bound value and computation time of Tabu Search Algorithm.

We make an improvement of $3,151,573,307$ meters approximately with a 10-hour run. If we assume that every customer traveling to a pharmacy needs to turn back to their home, then, we need to double this number to $6,303,146,614$ meters. Assume, traveling one kilometer with a car costs around 0.25 TL, then the monetary cost of this improvement will be around 1.6 million Turkish Liras for only the third period of 2010.

The gaps between the AO_M2 lower bound, Tabu Search algorithm and the current assignments are presented under the gap column in Table 7.1. The relative

Table 7.2: Comparison of Results for the 1^{st} Period of 2011

	Cost (demand-meter)	CPU (seconds)	Gap*
Current	410,889,602,394	—	40%
Tabu Search	317,121,539,733	55,428.838	8%
AO_M2 LB	293,916,086,606	0.087	—
AO_M1 LB	288,041,632,701	0.335	—

*: Relative gap using AO_M2 LB

gap between the lower bound and the Tabu Search solution is still huge. If we increase the iteration numbers, we may get closer to the optimal value and find better results. In this case, the computation times will be longer. Since this is not a every day problem, it is solved only three times a year, longer computation times are bearable in exchange for better results. The important point is to decide whether the improved solution quality or the computation time is more crucial.

7.2.2 Real Instance 2: 1^{st} Period of 2011

The number of pharmacies taking duties in the first period of 2011 has increased to 1053. For this period, again the current cost is calculated and it is compared with the proposed one. Table 7.2 shows the results in meters.

The improvement made for this period is $93,768,062,661$ meters. If we double this number for the round trip, the actual gain becomes $187,536,125,322$ meters. A tour around the world is approximately around $40,000$ kilometers.

For both periods, we make an assumption that all people will have a demand

for pharmacies each day of the planning horizon while calculating the improvements. This assumption is not valid in real life applications. However, the calculations for the current and the proposed solutions are made in the same way. Thus, we still have a fair comparison even though the calculated benefit may be less.

Another point is, when the working hours are over, each pharmacist has to put a sign on the pharmacy window which shows the on duty pharmacies of the current day. Currently, the pharmacies within the same region, direct customers only to the on duty pharmacies within their region. Sometimes the on duty pharmacy of that region is farther away in distance for that customer in comparison to the on duty pharmacy of another region. In the current planning system, the customer has to travel to the far away on duty pharmacy for the given reasons. When we calculate the cost of a given solution, we assign the customers to the closest open facility. On the other hand, in real life not all customers are traveling to the closest pharmacies due to the regional policy in use. Thus, the real cost of the current assignments are higher than our calculations. If the real cost of assignments could be calculated then, the improvement obtained will definitely be more than the calculated one.

Chapter 8

CONCLUSIONS AND FURTHER RESEARCH

In this study, we defined a new real life problem as the Pharmacy Duty Scheduling (PDS) Problem. We analyzed the complexity of this problem and we proved that the problem is NP-hard. We proposed solution methods for this problem using exact, heuristics and meta-heuristics algorithms.

We defined our problem in two different levels as the simple and the realistic problems. For both problems, we proposed mathematical programming models. We modeled the problems using integer programming models. We solved the small-size problems exactly, using *Cplex* solver on *GAMS*. For large-size problems, we proposed different algorithms using heuristics and meta-heuristics approaches. In order to test the solution quality of the algorithms, we developed different lower bound algorithms.

We gathered data for the current real life situation in İzmir. We tested our algorithms on two different duty periods that are the third period of 2010 and the first period of 2011. We calculated the objective value of the current assignments and compared our solutions with the assignments made for both periods. We showed that our algorithm makes significant improvements for both periods.

In the scope of future researches, exact algorithms can be further investigated and improved in order to solve the large-size problems. Different solution methods may be developed in order to solve the real life problem. Simple model 2 and realistic model 2 may be improved and be used in the solutions. The problem may also be evaluated with the scope of pharmacists not only the public.

Bibliography

[1] Hale, T. S., Moberg, C. R. (2003). "Location Science Research: A Review". *Annals of Operations Research*. 123, 21-35

[2] Daskin, Mark S. (1995). *Network and Discrete Location: Models, Algorithms, and Applications*. New York: John Wiley & Sons, Inc.

[3] Mirchandani, Pitu S. and Richard L. Francis. 1990. *Discrete Location Theory*. New York: John Wiley & Sons, Inc.

[4] Wesolowsky, George O., Truscott, William G. (1975). "The Multiperiod Location-Allocation Problem With Relocation of Facilities". *Management Science*. 22(1), 57-65

[5] Toregas, C., Swain, R., ReVelle, C., Bergman, L. (1971). "The Location of Emergency Service Facilities". *Operations Research*. 19(6), 1363-1373

[6] Çatay, B., Başar, A.,Ünlüyurt, T. (2008). "İstanbul'da Acil Yardım İstasyonlarının Planlanması". *Endüstri Mühendisliği Dergisi*. 19(4),20-35

[7] Carreras, M., Serra, D. (1999). On Optimal Location with Threshold Requirements". *Socio-Economic Planning Sciences*. 33(2), 91-103

[8] Rajagopalan, H. K., Saydam, C., Xiao, J. (2008). "A Multiperiod Set Covering Location Model for Dynamic Redeployment of Ambulances".

[9] Glover, F. (1989). "Tabu Search - Part I". *ORSA Journal on Computing*. 1(3), 190-206

[10] Glover, F. (1990). " Tabu Search - Part II". *ORSA Journal on Computing*. 2(1), 4-32

[11] Glover, F. (1990). " Tabu Search: A Tutorial". *INTERFACES*. 20(4), 74-94

[12] Glover, F., Laguna, M. (1997). " Tabu Search". *Kluwer Academic Publishers*

[13] Albareda-Sambola, M., Fernandez, E., Hinojosa, Y., Puerto, J. (2009). "The Multi-Period Incremental Service Facility Location Problem". *Computer and Operations Research*. 36, 1356-1375

[14] Wang, Q., Batta, R., Joyendu, B., Rump, C. M. (2003). "Budget Constrained Location Problem with Opening and Closing of Facilities". *Computers and Operation Research*. 30, 2047-2069 *Computers and Operations Research*. 35, 814-826

[15] Al-Sultan, K. S., Al-Fawzan, M. A. (1999). "A Tabu Search Approach to the Uncapacitated Facility Location Problem". *Annals of Operations Research*. 86, 91-103

[16] Bilgin, S., Azizoğlu, M. (2006). "Capacity and Tool Allocation Problem in Flexible Manufacturing Systems". *Journal of the Operational Research Society*. 57, 670-681

[17] Garey, M. R., David, S. J. (1979). " Computers and intractability: a guide to the theory of NP-completeness". *W. H. Freeman*

[18] Özpeynirci, Ö., Köksalan, M. (2009). "Multiobjective Traveling Salesperson Problem on Halin Graphs". *European Journal of Operational Research*. 196, 155-161

[19] Fisher, M. L. (1985). "An Applications Oriented Guide to Lagrangian Relaxation". *INTERFACES*. 15(2), 10-21

[20] Fisher, M. L. (2004). "The Lagrangian Relaxation Method for Solving Integer Programmimg Problems". *Management Science*. 50(12), 1861-1871

Appendix A

Geographic Information System

Figure A.1: Home Page of GIS Application

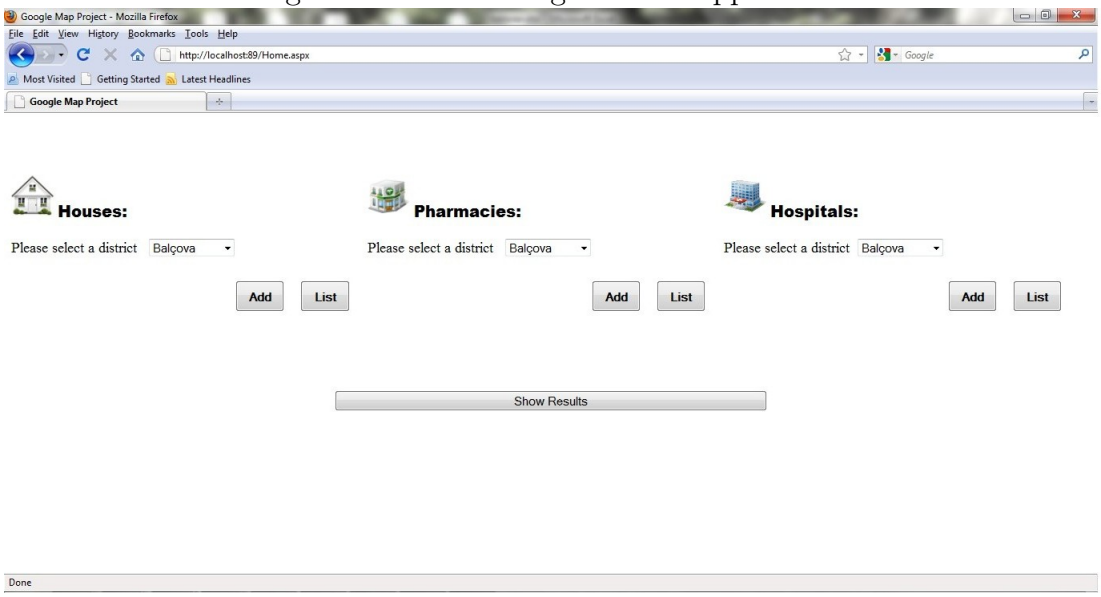


Figure A.2: Sample House List

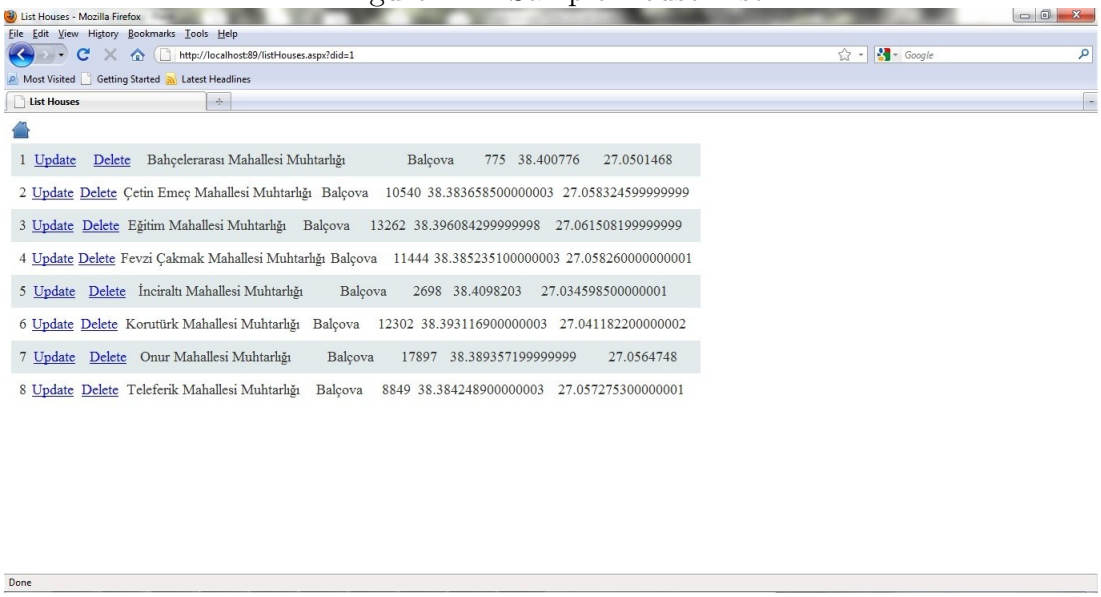


Figure A.3: Sample Pharmacy List

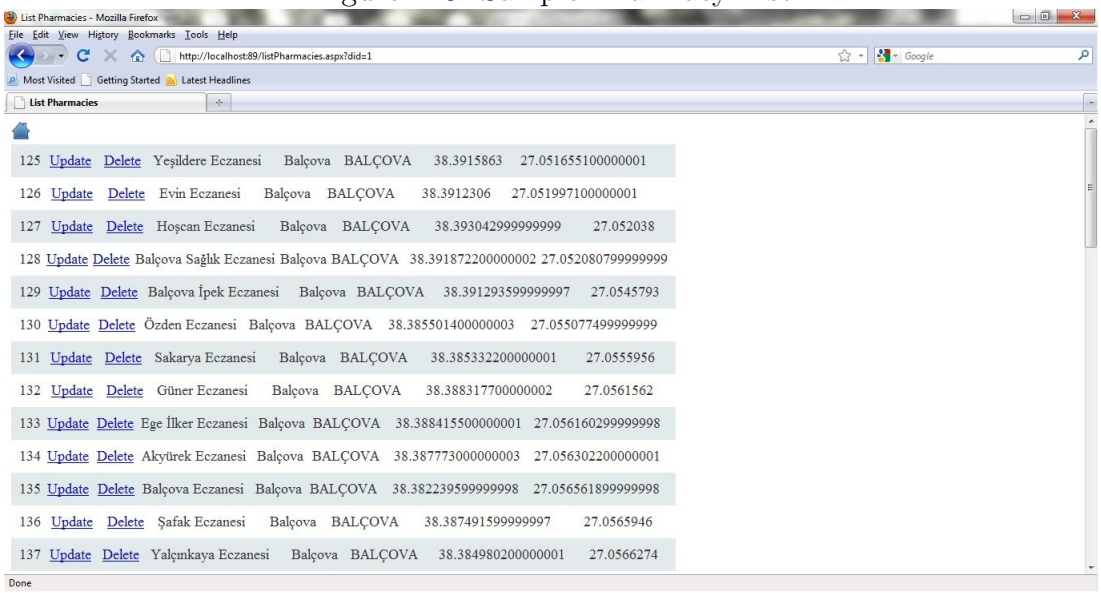


Figure A.4: Insert Pharmacy Screen

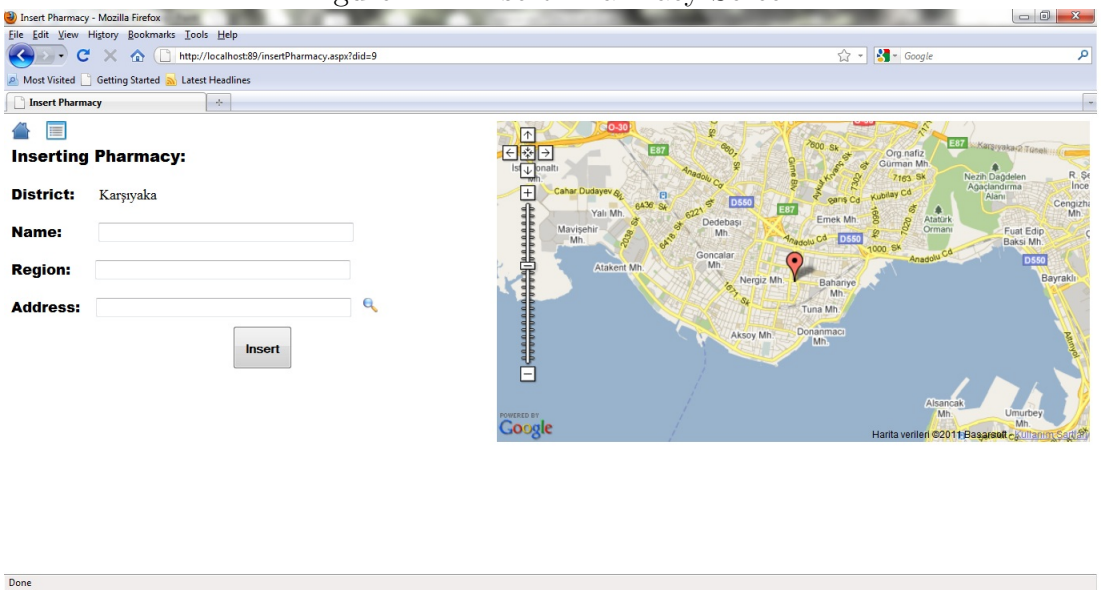


Figure A.5: Results Page

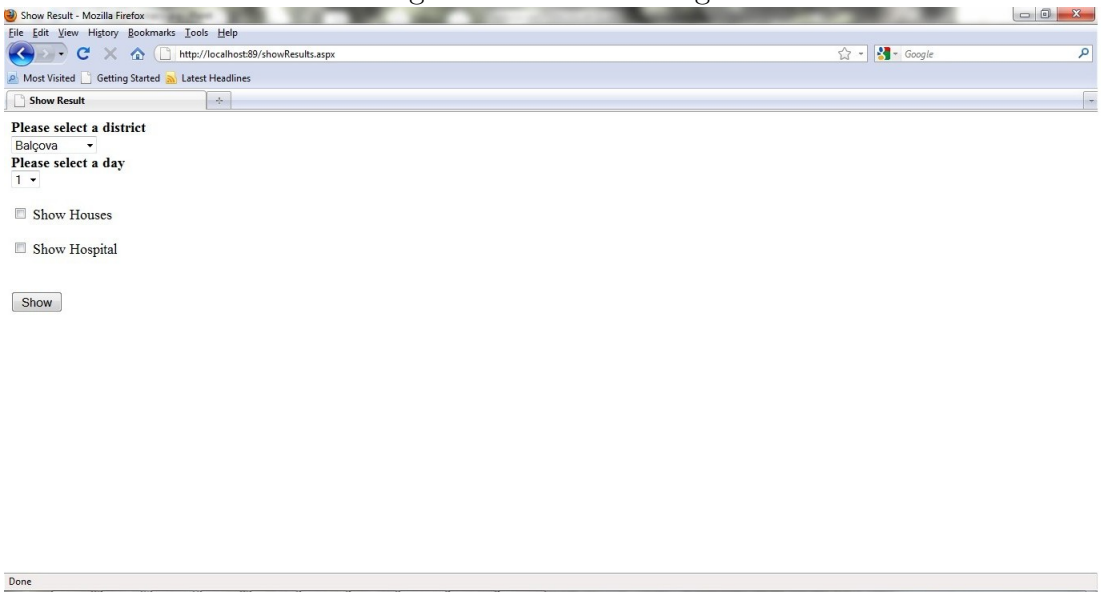


Figure A.6: Sample On-Duty Pharmacies