AN INTEGRATED ASSIGNMENT-ROUTING PROBLEM WITH TIME WINDOWS

AYBİKE (ÖZDEMİREL) AKICI

MAY 2014

AN INTEGRATED ASSIGNMENT-ROUTING PROBLEM WITH TIME WINDOWS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

IZMIR UNIVERSITY OF ECONOMICS

BY

AYBİKE (ÖZDEMİREL) AKICI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF SCIENCE

IN

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

MAY 2014

Approval of the Graduate School of Natural and Applied Sciences

_____
Prof. Dr. Cüneyt Güzeliş
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Intelligent Production Systems option of Intelligent Engineering Systems**.

_____
Assoc. Prof. Dr. M. Arslan Örnek
Head of 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 Science in Intelligent Production Systems option of Intelligent Engineering Systems**.

_____
Assoc. Prof. Dr. Deniz Türsel Eliiyi
Supervisor

Examining Committee Members:

Assoc. Prof. Dr. Deniz Türsel Eliiyi
Dept. of Industrial Engineering, IUE _____

Assoc. Prof. Dr. Yiğit Kazançoğlu
Dept. of Business Administration, IUE _____

Asst. Prof. Dr. Erdinç Öner
Dept. of Industrial Engineering, IUE _____

ABSTRACT

AN INTEGRATED ASSIGNMENT-ROUTING PROBLEM WITH TIME WINDOWS

(Özdemirel) Akıcı, Aybike

M.Sc. in Intelligent Engineering Systems
Graduate School of Natural and Applied Sciences

Advisor: Assoc. Prof. Dr. Deniz Türsel Eliiyi

May 2014, 62 pages

In this thesis, we consider a real-life public service problem at a donation center in Izmir, Turkey. The center is responsible for picking up incoming donated items from donors, assigning these items to incoming requests of the needy residents/clients in the district, and distributing the items. The unmatched items are stored at the single depot. A single vehicle is used for daily pickup and delivery, and the incoming requests have associated time windows for pickup or delivery. We propose novel utility-based assignment/routing integer programming models for this problem that assumes hard and soft time windows for service start times of the donors and clients. The details of the experiment design used for evaluating the performance of the developed model with soft time windows are presented, and computational results are discussed. With this thesis, we intend to contribute to the routing literature by introducing a novel integrated model, as well as providing optimal solutions to a practical and important humanitarian problem.

Keywords: Assignment and Routing Problem, Pickup and Delivery, Time Windows, Mixed Integer Programming.

ÖZ

ZAMAN ARALIĞI KISITLI BİR ATAMA-ROTALAMA PROBLEMİ

(Özdemirel) Akıcı, Aybike

Akıllı Mühendislik Sistemleri Yüksek Lisans Programı
Fen Bilimleri Enstitüsü

Tez Danışmanı: Doç. Dr. Deniz Türsel Eliiyi

Mayıs 2014, 62 sayfa

Bu tez çalışmasında İzmir Türkiye'de bulunan bir bağış merkezindeki bir gerçek hayat kamu hizmeti problemi ele alınmıştır. Merkez bağışlanan ürünlerin bağış sahiplerinden toplanılması, bu ürünlerin gelen istekler doğrultusunda ihtiyaç sahibi vatandaşlara atanması ve dağıtımından sorumludur. Atanamayan ürünler tek bir depoda geçici olarak bekletilmektedir. Günlük dağıtım ve toplama için tek araç kullanılmakta, gelen bağışçı ve ihtiyaç sahiplerinin ziyaret edilme saatlerine dair zaman aralıkları bulunmaktadır. Bu çalışmada problem için yeni ve özgün, fayda bazlı atama ve rotalamayı bir arada yapan tamsayılı programlama modelleri önerilmektedir. Modellerde bağışçı ve ihtiyaç sahiplerinin hizmet aldıkları zaman aralıkları bağlayıcı ve bağlayıcı olmayan kısıtlar olarak ele alınmıştır. Bağlayıcı olmayan kısıtlara sahip modelin performansının sayısal olarak değerlendirilmesi için geliştirilen deney tasarımı ve sonuçları ayrıntılarıyla sunulmuş ve tartışılmıştır. Bu tez çalışmasıyla hem rotalama literatürüne yeni ve özgün modellerle katkıda bulunmak, hem de pratik öneme sahip insani bir probleme optimal çözümler getirmek amaçlanmıştır.

Anahtar Kelimeler: Atama ve Rotalama Problemi, Dağıtım ve Toplama, Zaman Aralıkları, Karışık Tamsayılı Programlama.

ACKNOWLEDGMENTS

I would like to thank many people who helped me finish this graduate study, and will always be remarkable for me all my life.

First of all, I would like to present my sincere gratitude to my dear advisor Assoc. Prof. Dr. Deniz Türsel Eliiyi. Ever since the very beginning of this long journey, she has been there for me with her endless support and encouragement. She believed in me before I believed in myself, and made a great effort on me with endless patience and love. I will always owe her for not only the valuable contributions and help she provided for this thesis but also the unique vision, research mentality and self-esteem she gave to me that will remain as the most unforgettable gift I received.

I am also exceedingly grateful to Dr. Uğur Eliiyi for his invaluable help on coding our model on GAMS. Additionally, I would like to thank all my colleagues at İzmir University of Economics for being with me throughout this study.

I cannot end this acknowledgment without thanking my parents Sultan and Meriç Özdemirel, whose everlasting love I have relied on, for their endless support, encouragements and belief in me. They were my first teachers and educated me with many priceless values with compassion and never-ending love, to such an extent that no school can teach. I also want to thank to my sister, Ayça and my brother, Tuğşad for being my backbone when I couldn't stand up for myself. The warm memories of our childhood have a special place in my heart and I continue advocating them through all my life.

Last, but not the least, I am most grateful to my lovely husband Fatih Akıcı for his infinite support, guidance and love. Although he has been very far away from me physically during my Master's study, he was always the closest one to my mind and heart.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Transport and logistics decisions are important components of the cost of doing business, and are one of the main determinants of efficiency in today's competitive environment. The companies should take several tactical and operational logistics decisions in order to optimize their supply chain processes. These decisions are also essential for the efficiency of the public sector, which in turn affects public welfare.

In under-developed or developing countries, there is a considerable amount of needy population. Basic needs of this population require immediate attention of the governmental/municipal authorities. One of the ways of meeting the incoming requirements is via donated goods and funds collected from many donor residents. The municipal authorities, and specifically donation centers, are usually responsible for the timely collection and distribution of donated items to the needy in a fair manner. The operations at such a donation center constitute a logistics problem involving two interrelated decisions. The first is to determine a proper and fair matching of donated items and indigent residents, and the second is to efficiently collect and distribute these items in a timely manner. A fair matching is required for effectively meeting the needs, whereas efficient and timely distribution optimizes the utilization of public resources.

In this thesis, we take such a real life problem into consideration. The donation center in question is in Izmir, Turkey. People donate their used/discarded items, such as furniture, textile, white goods etc., to the center. During a day, the items that are declared by the donors via phone calls are collected by a single vehicle. Donations should be matched to the incoming requests by the indigent residents. The collected items are either directly distributed to their assigned

recipients during the day, or transported back to the depot and stored until need occurs. There are time windows in which each donor/receiver prefers to be served. Based on their needs and economic status, the donation center aims to distribute the items to needy residents in the fairest and most-gratifying manner, while keeping transportation costs at minimum. Currently, the decision makers at the center carry out these tasks manually, which leads to unsatisfactory/mismatched assignment reports and inefficient utilization of the vehicle, as well as high transportation costs.

We propose a multi-criteria assignment/routing integer programming model for the problem at hand. Numerous criteria affecting assignment decisions are merged under a utility definition. The objective of the developed model is to maximize the total utility of the assignments minus the total penalty resulting from time window preferences of the donors and residents, and the tour time. While the utility-maximizing assignment problem is somewhat straightforward to solve on its own, the integrated assignment/routing problem defined in this study is rather complex.

To the best of our knowledge, ours is the first study to consider an integrated assignment/routing problem. In this respect, as well as applying operations research techniques to a real-life problem to obtain optimal solutions, we intend to contribute to the routing literature by proposing this novel problem. We believe that the proposed problem can find its applications in many different problem environments including assignment and routing. The problem can also be extended for usage in diverse contexts involving public service or disaster relief situations.

In the next chapter, we discuss the literature regarding our problem. In Chapter 3 we define the problem and present the proposed mathematical model. We explain the details of our experiment design in Chapter 4.1 while presenting our computational experiment results in Chapter 4.2. We conclude with future research directions in Chapter 5.

CHAPTER 2

LITERATURE REVIEW

The routing component of our problem includes pickups and deliveries with a single vehicle and a single depot. The Pickup and Delivery Problem (PDP) formulates constructing a set of routes in order to satisfy transportation requests by a fleet of vehicles (Savelsberg and Sol, 1995). Every transportation request has an origin where the load is to be picked up, and a destination it is to be delivered. All vehicles depart from and return to the depot. In general, PDP is analyzed in terms of its different problem characteristics; dynamic vs. static, and single vs. multi-vehicle. In the dynamic case, the requests are gradually obtained through the day, whereas all requests are known in advance in the static case. According to the number of vehicles in the fleet, the problem has been named as single or multi-vehicle PDP.

Parragh et al. (2008a, 2008b) conduct a two-part comprehensive survey on pickup and delivery problems and distinguish two basic categories for the classification of the problems. The first class deals with the transportation of the goods from the depot to linehaul customers and from backhaul customers to the depot, whereas the second class refers to the problems where goods are transported between multiple pickup and delivery locations. Figure 1 shows the classification of the pickup and delivery problems presented by Parragh et al (2008a, 2008b). In the second class, there are two subclasses that refer to the situations where pickup and delivery points are unpaired or paired. In the unpaired case where each unit picked up can fulfill the demand of any delivery point, the problem is denoted as Vehicle Routing Problem with Pickup and Delivery (PDVRP). Other subclass comprises the PDP problems resulting in paired pickup and delivery locations. Based on the characteristic of our problem environment, the daily problem at the donation center is a single vehicle static problem with time windows, as all donations and requests are known at the start of the working day, and the pickup and delivery activities take

place between customer pairs as well as the depot. In addition, any incoming request/demand during the day is recorded to be planned for the upcoming days. Hence, our problem has similarities with the problems in the second subclass of the second class, as defined by Parragh et al. (2008a, 2008b). In this chapter, we focus on the related literature on this second class and its subdivisions, namely the PDP and the Dial-a-Ride Problem (DARP).

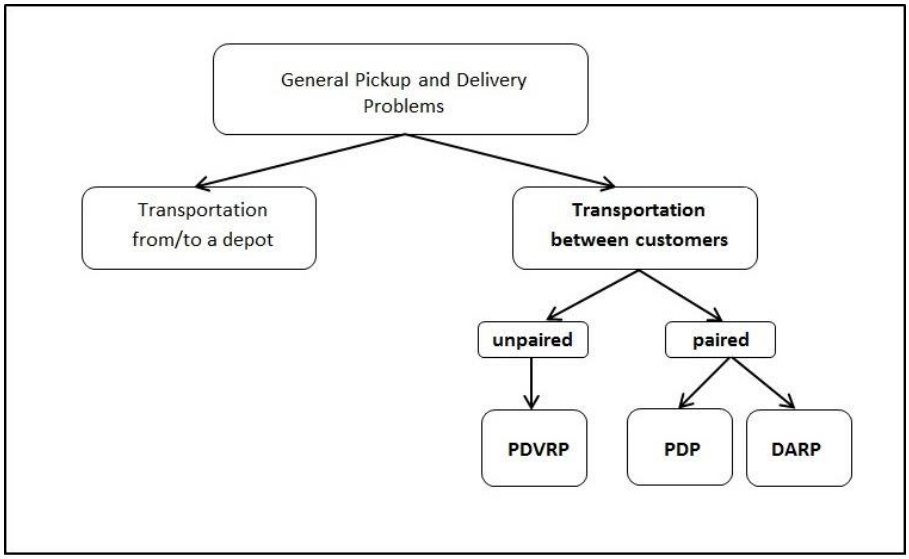


Figure 1 Classification of Pickup and Delivery Problems

In literature, various approaches are developed to solve PDP variations including time windows. Some of these are exact methods which guarantee the optimal solution, while the majority is on approximations providing feasible solutions in acceptable times (Savelsberg and Sol, 1995). As an example for studies on exact methods, Kalantari et al. (1985) present branch and bound algorithm for the single vehicle PDPTW, considering both infinite and finite vehicle capacity. Besides, two new formulations for the problem and the closely related DARP are proposed by Ropke et al. (2006) in which there are a limit on the elapsed time between pickup and delivery of a request. Valid inequalities are used to strengthen the formulations which are used within branch-and-cut algorithm. Ropke and Cordeau (2007) introduce a new branch-and-cut-and price algorithm for solving PDPTW in which lower bounds are computed by solving through column generation linear programming relaxation of a set partitioning formulation. There are two different problems as considered sub problems in column generation algorithm and valid

inequalities are added to strengthen the relaxations. It is reported that the proposed algorithm outperforms the recent branch-and-cut algorithms. In another study, Dumitrescu et al. (2008) model Travelling Salesmen Problem with Pickup and delivery (TSPPD) as an integer linear program and analyze its polyhedral structure. The dimension of TSPPD is determined and several valid inequalities are proposed. Furthermore, separation procedure and branch-and-cut are developed and computational results are presented.

Several studies report heuristic and metaheuristic approaches on the single vehicle PDPTW. Van der Bruggen et al. (1993) develop a variable-depth based local search algorithm for the problem. It consists of two phases in which a feasible route is constructed in the first phase, and the route is iteratively improved in the second phase. It is reported that the method provides near-optimal solutions in a reasonable computation time. Nanry and Barnes (2000) present a reactive tabu search approach using three different neighborhood moves highlighting the dominance of precedence or coupling constraints. A hierarchical search is used to alternate between neighborhoods dynamically. Landrieu et al. (2001) present a tabu search and a probabilistic tabu search for the problem, while Kammatri et al. (2004) provide a hybrid approach that uses an evolutionary algorithm with special genetic operators, tabu search and Pareto dominance method. The algorithm produces satisfying and feasible solutions minimizing the compromise between total travelled distance, total waiting time, and total tardiness. Hosny and Mumford (2010) also study the single vehicle PDPTW, and compare three different approaches; a genetic algorithm, a simulated annealing approach and a hill climbing algorithm. In all cases, they adopt a solution representation and present an intelligent neighborhood move that is guided by the time window. Huang and Ting (2010) develop an ant colony optimization (ACO) method for the problem. The constraint of the time window and the vehicle capacity is considered in the transition probability of the ants. In order to deal with infeasible routes, they propose a repair operator and it is reported as a result that the solution quality of the ACO method outperforms tabu search and genetic algorithm for the studied problem.

For the multi-vehicle PDPTW, Li and Lim (2001) propose a hybrid metaheuristic, in which a tabu-embedded simulated annealing restarts a search procedure from the best current solution after several non-improving search iterations. The authors report that their algorithm is

the first approach to solve large instances of the multiple-vehicle PDPTW with various distribution properties. Minic et al. (2004) focus on the dynamic multi-vehicle PDPTW, and describes a double-rolling-horizon-based solution method for the problem. They consider the short and long term effects of assigning a new request to a vehicle into consideration. Pankratz (2005) describes a Grouping Genetic Algorithm that has a feature of group-oriented genetic encoding, in which each gene represents a group of requests instead of a single request. In this study, all problem variants are transformed into a pickup and delivery model, and solved using the adaptive large neighborhood search (LNS). In another study that uses LNS, Bent and Van Hentenryck (2006) present a two-stage hybrid algorithm for solving the multi-vehicle PDPTW. In the first stage, they use a simple simulated annealing procedure to decrease the number of routes while they propose LNS in order to decrease the travel cost. Lu and Dessouky (2006) present a new insertion-based construction heuristic for the same problem. They consider the cost of reducing the time windows slack due to the insertion and present a non-standard measure, the Crossing Length Percentage, for evaluating the visual attractiveness of the proposed solution. They compare their procedure with previous parallel and sequential insertion heuristics, and report that the proposed method performs better regarding both standard and non-standard measures. A unified heuristic that is capable of solving five variants of the VRP (the vehicle routing problem with time windows, the capacitated vehicle routing problem, the multi-depot vehicle routing problem, the site-dependent vehicle routing problem, and the open-vehicle vehicle routing problem) is presented by Pisinger and Ropke (2007).

Recently, Lai et al. (2010) develop a two-stage hybrid metaheuristic for the multi-vehicle PDPTW, where the first stage is comprised of decreasing the number of used vehicles by simulated annealing and tabu search is used to decrease the total travel cost in the second stage. Hosny and Mumford (2011) compare several construction algorithms that are used to generate initial feasible solutions for the problem. The algorithms differ in whether generated algorithms are sequential or parallel, and the criteria for selecting requests and the routes that they will be inserted; inserting a request into a route is either based on a first acceptance criterion or a best acceptance criterion. Barbeglia et al (2010) present a recent and comprehensive survey on the dynamic PDP and solution strategies. Wang and Chen (2013) formulate a flexible PDPTW into mixed binary integer programming and develop a co-evolutionary algorithm with a variant of the

cheapest insertion method for expediting the solution procedure. Zou et al. (2013) study the multi-objective PDPTW with three objectives; minimizing the total number of vehicles utilized, the total travel distances, and the total waiting times. They build a mixed integer programming model and a novel hybrid particle swarm optimization algorithm that adds particles' neighbor information to diversify the particle swarm and use the variable neighborhood search (VNS) to enhance the convergence speed. Table 1 summarizes the studies on PDPTW reviewed in this chapter with respect to problem characteristics and the proposed solution methodologies.

DARP is a special case of the PDP and an extension from the typical Travelling Salesman Problem (TSP) where the objective is to minimize the total degree of customer dissatisfaction defined by the service waiting time of a customer (Psaraftis 1980). Mainly, the DARP aims at designing vehicle routes and schedules for *n* users with pickup and drop-off activities. The same user may have two requests during the same day; an outbound request (from home to destination) and an inbound request (for return). The fleet consists of identical vehicles based on a single depot and the aim is to accommodate as many as requests possible. Applications of the problem can be seen in door-to-door transportation services for elderly and disabled people (Cordeau and Laporte 2003a).

Sexton and Bodin (1985a) develop a model that focuses on solving the scheduling component of the single vehicle static DARP. Each of the given a set of *n* customers has a desired point of delivery and pickup, and also a service time. They solve the developed mathematical modeling formulation that has the objective of minimizing total customer inconvenience through Bender's decomposition. The authors test their heuristic algorithm on real data, and report high quality solutions in a reasonable computation times. In a later study, the same authors develop another algorithm for finding an initial solution of the route sequence for the same problem (Sexton and Bodin 1985b). They also present an improvement heuristic for the route sequence. Former scheduling and current routing algorithms are integrated, and satisfactory results from a number of computational experiments on actual data are presented in the study. Ozdemirel et al. (2012) deals with a similar problem and develops an integrated assignment-routing model with hard time windows and vehicle capacity constraint, as will be further explained in the third chapter.

Table 1 Summary of the reviewed studies on PDPTW

	# of vehicles		demand structure		solution method		
	single	multi	static	dynamic	exact	approx.	
Kalantari et al. (1985)	✔		✔		✔		branch and bound
Ropke et al. (2006)		✔	✔		✔		branch and cut
Ropke and Cordeau (2007)	✔		✔		✔		branch-and-cut-and-price
Dumitrescu (2008)	✔		✔		✔		integer programming, branch and cut
Van der Bruggen et al. (1993)	✔		✔			✔	variable depth search
Nanry and Barnes (2000)	✔		✔			✔	reactive tabu search
Landrieu et al. (2001)	✔		✔			✔	tabu search
Kammatri et al. (2004)	✔		✔			✔	hybrid evolutionary approach
Hosny and Mumford (2010)	✔		✔			✔	genetic algorithm, simulated annealing, hill climbing
Huang and Ting (2010)	✔		✔			✔	ant colony optimization
Li and Lim (2001)		✔	✔			✔	tabu-embedded simulated annealing
Minic et al. (2004)		✔		✔		✔	a double-horizon based heuristic
Pankratz (2005)		✔	✔			✔	a grouping genetic algorithm
Lu and Dessouky (2006)		✔	✔			✔	an insertion-based construction heuristic
Bent and Van Hentenryck (2006)		✔	✔			✔	simulated annealing, large neighborhood search
Pisinger and Ropke (2007)		✔	✔			✔	a unified heuristic
Lai et al. (2010)		✔	✔			✔	simulated annealing, tabu search
Barbeglia et al. (2010)		✔		✔		✔	various solution strategies
Hosny and Mumford (2011)		✔	✔			✔	initial feasible solution construction algorithm
Zou et al. (2013)		✔	✔			✔	hybrid particle swarm optimization
Wang and Chen (2013)		✔	✔			✔	a co-evolutionary algorithm

There are numerous studies in literature that focus on approximate methods for solving the DARP. Hame (2011) proposes an adaptive insertion algorithm that can produce optimal solutions for a time-constrained the single vehicle dynamic DARP. One of the heuristics for the multi-vehicle static DARP is proposed by Jaw et al. (1986). The heuristic includes time constraints, defined as the amount of time by which the pickup or delivery of a customer can deviate from the desired pickup or delivery time, and the time that a customer can spend riding in a vehicle. The algorithm consists of a sequential insertion procedure that enables assigning customers to the vehicles, and determining the time schedule of all pickup and delivery activities for every vehicle. The objective function holds the balance between the costs of providing service regarding customers' desired service times and short riding times. Madsen et al. (1995) describe a system for the solution of a static dial-a-ride routing and scheduling problem with time windows. The problem is characterized by multiple capacities and multiple objectives. Multiple capacity definitions refer to the fact that a vehicle may be equipped with different types of seats; e.g. normal seats, children seats or wheel chair places, whereas the objectives relate to a number of concerns such as short driving time, high vehicle utilization or low costs. A heuristic, called REBUS, is proposed in which new customer requests are dynamically inserted in vehicle routes. The algorithm is applied in a dynamic environment targeting for online scheduling. Ioachim et al. (1995) presents an approximate method to mini-clustering, which involves solving a static multi-vehicle DARP with time windows. They use several ways to enhance an existing optimal algorithm. In addition, they develop a heuristic that reduces the size of the solution network that causes only small losses in solution quality. In order to compare the result of optimization-based and local heuristic mini clustering, a parallel insertion approach to mini clustering is proposed.

In a more recent study, Cordeau and Laporte (2003b) propose a tabu search heuristic for the multi-vehicle static DARP with time windows, in which the users may specify the time windows on their desired departure or arrival times. The objective is to design a set of least-cost vehicle routes capable of fulfilling all user requests subject to vehicle capacity, route duration and the maximum ride time of any user. Diana and Dessouky (2004) present a parallel insertion heuristic for large-scale DARP with time windows. A new route initialization procedure is implemented that takes into account the spatial and temporal aspects of the problem, and also a regret insertion heuristic is performed to serve the remaining requests. The results show that this

approach is effective regarding trading-off solution quality and the computational times. Table 2 summarizes the discussed papers on DARP in terms of their problem characteristics and solution approaches.

Table 2 Summary of the reviewed studies on DARP

	# of vehicles		demand structure		solution method		
	single	multi	static	dynamic	exact	approx.	
Psaraftis (1980)	✔			✔	✔		dynamic programming
Sexton and Bodin (1985a)	✔		✔		✔		binary integer programming
Sexton and Bodin (1985b)	✔		✔			✔	a problem-specific heuristic
Ozdemirel et al. (2012)	✔		✔		✔		mixed integer programming
Hame (2011)			✔			✔	an adaptive insertion algorithm
Jaw et al. (1986)		✔	✔			✔	a sequential insertion heuristic
Madsen et al. (1995)		✔		✔		✔	an insertion-based heuristic
Iochim et al. (1995)		✔	✔			✔	a mini clustering algorithm
Cordeau and Laporte (2003b)		✔	✔			✔	tabu search
Diana and Dessouky (2004)		✔		✔		✔	a regret insertion heuristic

The routing component of our problem resembles the single vehicle static dial-a-ride problem more than other types of routing problems. Both the dial-a-ride problem and our problem consider pickup and delivery activities during the same routing schedule. However, there are several and crucial differences between the two problems, as will be more obvious in Chapter 3. For example, in the dial-a-ride problem, a client is picked up from his home and transported to a destination. After a while, he is picked up from that point and delivered to his home again. In contrast, an item is picked up from a donor's home and delivered to either the assigned recipient's home or the depot in our problem; there is no return flow for any item. From

the humanitarian or public welfare perspective, both the dial-a-ride problem and our problem have customer-based objectives. The former minimizes total customer inconvenience defined by excess ride time and delivery time deviations (Sexton and Bodin, 1985b), whereas the latter considers the routing of utility maximizing assignments.

CHAPTER 3

PROBLEM DEFINITION

The main objective of the assignment/routing problem presented in this study is to maximize the total utility of the assignments. Two different models are considered for solving this problem, which differentiate from one another with regards to their objective functions. The first model, which was published previously in Ozdemirel et al. (2012) seeks for maximizing the total utility of daily assignments under a set of hard time window constraints. The second model developed in this thesis aims to maximize the total utility minus the aggregate penalty resulting from time window violations. Namely, the time window constraints are defined as soft constraints in the new model.

Before presenting the developed models, the procedure for computing the utility values for all feasible assignments is explained in detail in the following section.

3.1. Procedure for Computing Utilities

In order to compute the utility of an item-client assignment, several criteria are taken into consideration, including the travel time between the donor and the client, the income level and the age of the client, the number of household at the client's residence, the client's previous usage of the donation service, and the age/status of the donated item. These criteria, as listed in Table 3, are determined as a result of interviews with decision makers at the donation center in Balçova, İzmir. Obviously, the number of criteria, their levels and respective scores should be adjusted for other applications of the problem.

Table 3 Criteria and scores used in utility computations

Criteria / Score	$s_{i1} = 10$	$s_{i2} = 40$	$s_{i3} = 100$
Travel time	btw. 11-15 min.	btw. 5-10 min.	< 5 min.
Income	>1000 TL	btw. 500-1000 TL	< 500 TL
Age of client	btw. 20-50	btw. 50-70	> 70
# of household	2	btw. 3-5	> 5
# of service received	> 7	btw. 4-6	< 4
Item age	> 10	btw. 5-10	< 5

Among the criteria, the income level, age of the recipient/client and previous use of service are indicators for the indigent's social and economic status. The number of household and the age of the client are stated to be mutually exclusive by the decision makers; that is, only one is considered for the same recipient. Specifically, if the recipient lives alone, then the age criterion is taken into consideration, whereas if he doesn't, the number of household determines the priority of the client. The last criterion, namely the item age, is considered for a fair assignment in accordance with other criteria. Three levels are determined for the value ranges of each criterion.

Table 3 illustrates the levels and their scores for each criterion assumed in this thesis. At the beginning of the working day, all donor/client requests to be served are already determined; hence the scores are set for each possible assignment. The utility value corresponding to each possible assignment is calculated through a similar procedure as defined in Ozdemirel et al. (2012), although with updated levels and scores for the criteria. The procedure is presented with an example here for the sake of completeness.

Priorities among the criteria are determined through interviews with the decision makers of the donation center. For this purpose, a series of judgment-based pairwise comparisons among the criteria are made by the decision makers as in the Analytical Hierarchy Process (AHP), which is a method for determining the relative importance of various factors (Saaty, 1980). After this analysis, consistency check was completed and a relative importance matrix was formed. As a result, the relative importance values are determined as shown in Table 4, which establish the basis for utility computation of all possible assignments.

Table 4 Relative importance matrix for the criteria

Criterion	Travel Time	Income	Age/ # hh	# service	Item age
Travel time	1	1/7	1/5	1/9	3
Income	7	1	7	3	9
Age / # of household	5	1/7	1	1/5	5
# of service received	9	1/3	5	1	7
Item age	1/3	1/9	1/5	1/7	1

After the normalization procedure, the normalized weights of the criteria were set as w_k, k=1,…,6, where w_3 and w_4 are the mutually exclusive, as stated above. The determined weights for criteria are listed Table 5.

Table 5 Normalized weights of the criteria determined by pairwise comparison

Criteria	**Weight**
Travel time	0.06
Income	0.49
Age / # of household	0.13
# of service received	0.29
Item age	0.03
Total	1.00

Based on these weights, the utility value of each possible assignment is computed by multiplying the weight of each criterion with the score of the assignment in terms of that criterion, and taking the summation of these multiplications. The value that is obtained finally can be inserted into the objective function directly.

As a numerical example for utility computation, consider the feasible item/client assignment (i,j) with the following criteria levels: Let the time between nodes i and j be 7 minutes. The income level of the senior resident at node j, who lives alone, is less than 500 TL, and the age of the resident is 75. This resident has used the donation service before 4 times. And the item donated by node i has an age of 11 years. This assignment has the following respective scores from each criterion according to Table 3: Travel time: 40, Income: 100, Age of client: 100, # of service: 40, Item Age: 10. Multiplying each criterion's score with its respective weight in Table 3 yields the utility of assignment (i,j) as the following convex combination:

u_{ij} = 0.06(40) + 0.49(100) + 0.13(100) + 0.29(40) + 0.03(10)

u_{ij} = 76.3

A utility value in the range (10, 100) is obtained for any feasible assignment using the above procedure. In the next section, we present our first mathematical model, which uses maximizing the sum of computed utility values as its objective function.

3.2. Model 1: The Integrated Assignment/Routing Model with Hard Time Window Constraints

In an earlier publication, we present (Ozdemirel et al., 2012) a mathematical model for simultaneous decisions of the matching of the donated items and clients, and the routing of a single capacitated vehicle in the presence of pickups and deliveries and time windows. As it was stated before, this model assumes hard time window constraints.

The planning period is assumed as a working day. It is assumed that all requests that are taken at the start of the day are the input of the model. The parameters of the model are as follows:

n : the number of nodes (including the depot as the last node)

d : number of donor nodes, where $d < n$

i, j, h : the indices for all nodes, $i, j = 1, \ldots, n$

v_i : volume at node i, $i = 1, \ldots, n$

u_{ij} : the utility of assigning node i to node j

t_{ij} : travel time of the vehicle between node i and node j

a_i : start of time window for node i, $i = 1, \ldots, n$-1

b_i : end of time window for node i, $i = 1, \ldots, n$-1

C : total volume capacity of the vehicle

The following denote the decision variables:

x_{ij} : takes the value of 1 if node (item) *i* is assigned to node *j*, and 0 otherwise

y_{ij} : takes the value of 1 if the vehicle visits node *i* immediately before node *j*, and 0 otherwise

s_i : the start time of service at node *i*

c_i : load (in volume) of the vehicle after visiting node *i* ($c_i < C, \forall i$)

Node indices from 1 to *d* represent the donor nodes where indices from *d*+1 to *n*-1 stand for the resident nodes. The last index *n* symbolizes the depot.

Each request of an indigent resident (client) and each donated item are taken as separate records, since each matching may receive different scores from the criteria according to the status of the requested item. The model therefore keeps separate node indices for each request and each donation, although they may belong to the same address. Dummy nodes are created, as required by this representation. Items at the depot at the start of the day are also treated as separate donor nodes. The proposed network-based mixed integer programming model is given below.

Maximize $\sum_{i=1}^{d}\sum_{j=d+1}^{n} u_{ij}x_{ij}$

subject to:

$$\sum_{j=d+1}^{n-1} x_{ij} + x_{in} = 1 \qquad i = 1,...,d \tag{1}$$

$$\sum_{i=1}^{d} x_{ij} \le 1 \qquad j = d+1,\ldots,n-1 \tag{2}$$

$$\sum_{h=1,h\neq i}^{n} y_{ih} + \sum_{h=1,h\neq j}^{n} y_{hj} - x_{ij} \ge 1 \quad i = 1,...,d,\ j = d+1,\ldots,n-1 \tag{3}$$

$$\sum_{j=1,\,j\neq i}^{n} y_{ij} = 1 \qquad \forall i \tag{4}$$

$$\sum_{i=1,\,i\neq j}^{n} y_{ij} = 1 \qquad \forall j \tag{5}$$

$$\sum_{j=1}^{n-1} y_{nj} \leq 1 \tag{6}$$

$$\sum_{i=1}^{n-1} y_{in} = 1 \tag{7}$$

$$a_i \leq s_i \leq b_i \qquad \forall i \tag{8}$$

$$s_j - s_i + b_i\left(1 - x_{ij}\right) \geq 0 \qquad i = 1,\ldots,d,\ j = d+1,\ldots,n-1 \tag{9}$$

$$s_j - s_i - t_{ij} + \left(b_i + t_{ij}\right)\left(1 - y_{ij}\right) \geq 0 \qquad i = 1,\ldots,n,\ j = 1,\ldots,n-1, i \neq j \tag{10}$$

$$c_j - c_i - v_j + \left(C + v_j\right)\left(1 - y_{ij}\right) \geq 0 \qquad i = 1,\ldots,n,\ j = 1,\ldots,n-1, i \neq j \tag{11}$$

$$c_i \leq C \qquad \forall i \tag{12}$$

$$s_n, c_n = 0 \tag{13}$$

$$x_{ij}, y_{ij} \in \{0,1\},\ s_i, c_i \geq 0 \qquad \forall i, j \tag{14}$$

The objective function maximizes the total utility of the daily assignments. Constraint set (1) ensures that each item must be assigned to either a client or the depot. Constraint set (2) allows that a client may not receive an item although he has posted a request for the particular working day. Constraint set (3) ensures that a client cannot be visited before its related donor is visited.

Constraint sets (4) to (7) set the start and end of the tour as the depot. Also, they handle that each node is visited exactly once except the depot. Constraint set (8) establishes time windows while constraint sets (9) and (10) define the service start time for each node. Constraint set (11) and (12) together handle the capacity constraint of the vehicle. Constraint sets (13) states

that the vehicle is initially empty and it starts its tour at the depot. Finally, constraint set (14) imposes sign restrictions on the decision variables.

While maximizing total utility, the above model tries to handle the time window restrictions of the donors and clients using hard time window constraints. After initial experimentation it was observed that this aspect of the model is rather unrealistic, and may cause infeasibility based on the structure of the data. Based on the model, any client node that cannot be visited during its time window is left out of the solution and is not included in the tour as the model allows no assignments for some clients. On the other hand, this situation is not possible for the donor nodes; an infeasible solution is obtained if a donor node cannot be visited due to its time restrictions. This aspect illustrates the fact that the feasibility of the above model is highly dependent on data.

In addition, this model includes an artificially large node set, as it assumes that all requests that are taken for the particular working day become the input of the model. Note that the second constraint allows no assignments for some clients. This means that while the requests of all clients are collected and their time window requirements are taken through telephone calls, some of them will not be visited during that day. The time requests for these citizens should be taken again for the next day, although a visit is, again, not guaranteed. Consequently, while this aspect artificially increases the number of decision variables and constraints of the model, it is not reasonable in a practical sense.

For these reasons, we develop a second model that will overcome the shortages of the previous one. We explain our second model in the next section.

3.3. Model 2: The Integrated Assignment/Routing Model with Soft Time Window Constraints

As the first one, our second model utilizes a network structure, representing each donation and each client by a separate node. While the previous model assumes hard constraints, the second model employs a soft time window constraint structure for a more realistic representation of the problem environment. For this purpose, while allowing violations of time window preferences for

donor and client nodes, the model penalizes the violations, both earliness and tardiness, in its objective function.

As another difference from the earlier model, the second model assumes that, at the start of the working day, all n nodes that can be visited during the day are already determined on a first-come-first-serve basis through the incoming calls. This assumption is in line with the current practice at the donation center. Hence, the list of donors and clients that will be visited is known, though the assignments are not. This allows us to use equality in the second set of constraints.

The parameters of the second model are defined as follows:

$n + 1$: the total number of nodes

d : number of donor nodes, where $d < n$

i, j, h : indices for all nodes, $i, j = 1, \ldots, n$

v_i : volume at node i, $i = 1, \ldots, n$

u_{ij} : the utility of assigning node i to node j

t_{ij} : travel time (in minutes) of the vehicle between node i and node j

d_{ij} : travel time (in minutes) of the vehicle between node i and node j (excluding service time)

a_i : start of time window for node i, $i = 1, \ldots, n$ (in minutes)

b_i : end of time window for node i, $i = 1, \ldots, n$ (in minutes)

C : total volume capacity of the vehicle

T : total length of the working day (in minutes)

p_i : penalty of visiting node i out of its time window, $i = 1, \ldots, n$.

$I(i)$: incompatibility set for the item at donor node i, $i = 1, \ldots, d$. This is a list of the client nodes that cannot be assigned to node i, since they require different types of items.

The following denote the decision variables:

x_{ij} : takes the value of 1 if node (item) *i* is assigned to node *j*, and 0 otherwise

y_{ij} : takes the value of 1 if the vehicle visits node *i* immediately before node *j*, and 0 otherwise

s_i : the start time of service at node *i* (in minutes)

c_i : load (in volume) of the vehicle after visiting node *i* ($c_i < C, \forall i$)

e_i : amount of time that node *i* is served early (earliness value in minutes)

l_i : amount of time that node *i* is served late (tardiness value in minutes)

Node indices from 1 to *d* represent the donor nodes where indices from *d*+1 to *n* stand for the client nodes. Node 0 and node *n* + 1 represent the artificial nodes for source and sink. The mixed integer programming model is given below:

Maximize $\sum_{i=1}^{d}\sum_{j=d+1}^{n} u_{ij}x_{ij} - \sum_{i=1}^{n} p_i\left(e_i + l_i\right) - s_{n+1}$

subject to:

$$\sum_{j=d+1}^{n} x_{ij} = 1 \qquad i = 1,...,d \tag{15}$$

$$\sum_{i=1}^{d} x_{ij} = 1 \qquad j = d+1,\ldots,n \tag{16}$$

$$\sum_{i=1}^{d}\sum_{j=d+1}^{n} x_{ij} = 0 \qquad j \in I(i) \tag{17}$$

$$\sum_{j=1}^{d} y_{0j} = 1 \tag{18}$$

$$\sum_{j=d+1}^{n} y_{0j} = 0 \tag{19}$$

$$\sum_{i=1}^{d} y_{i,n+1} = 0 \tag{20}$$

$$\sum_{i=d+1}^{n} y_{i,n+1} = 1 \tag{21}$$

$$\sum_{\substack{j=1 \\ j \neq i}}^{n+1} y_{ij} = 1 \qquad i = 1,...,n \tag{22}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} y_{ij} = 1 \qquad j = 1,...,n \tag{23}$$

$$c_i \leq C \qquad i = 1,...,n \tag{24}$$

$$c_0, c_{n+1} = 0 \tag{25}$$

$$e_i \geq a_i - s_i \qquad i = 1,...,n \tag{26}$$

$$l_i \geq s_i - b_i \qquad i = 1,...,n \tag{27}$$

$$s_j \geq s_i + \min_{\substack{k=1,...,n \\ k \neq i}} \{d_{ik}\} - T\left(1 - x_{ij}\right) \qquad i = 1,...,d,\ j = d+1,...,n \tag{28}$$

$$s_j \geq s_i + t_{ij} - T\left(1 - y_{ij}\right) \qquad i = 0,...,n,\ j = 1,...,n+1, i \neq j \tag{29}$$

$$c_j \geq c_i + v_j - 2C\left(1 - y_{ij}\right) \qquad i = 0,...,n,\ j = 1,...,n+1, i \neq j \tag{30}$$

$$x_{ij}, y_{ij} \in \{0,1\},\ s_i, c_i, e_i, l_i \geq 0 \qquad \forall i, j \tag{31}$$

The objective function maximizes the total utility of the daily assignments while minimizing the total penalty resulting from violations of the time windows and the tour time. The tour time is identified as the service start time of the sink node. Constraint set (15) ensures that each donation must be assigned to a client. In the data set, several nodes may represent the depot as a client. Constraint set (16) ensures that all recorded/promised clients will receive an item during the working day. Constraint set (17) ensures that all clients receive items of their required types. Although the utility values for mismatched item type/donor assignments are set to zero, the model can make such assignments in order to decrease tour time or total time penalties. For this reason, constraint set (17) is necessary to completely avoid mismatching assignments.

Constraint (18) links the source node to a donor node at the start of the tour, where several nodes may represent the donated items at the depot. Constraint (19) ensures that the source node

cannot precede a client node; an empty vehicle cannot visit a client at the start of the working day. Similarly, constraint (20) does not allow a donor node to be visited at the end of the day by prohibiting all links to the sink node. Constraint (21) ensures that the sink node follows a client node (depot included). Constraint sets (22) and (23) control conservation of flow in the tour for the remaining nodes.

Constraint set (24) bound the load on vehicle after visiting each node by the vehicle capacity. The vehicle is empty at the start and end of the tour, as indicated by constraints (25). The earliness and lateness values at each node are calculated through constraint sets (26) and (27), respectively.

After experimentation with the previous model, it has been observed that constraint set (3) is redundant for linking the binary variables. The connection is ensured through constraint sets (28) and (29) in the second model. These constraint sets guarantee that an assigned pair of nodes should be visited in proper order during the tour, while simultaneously determining the service start time of each node. In the constraints, the length of the working day T is taken as a large number. Constraint set (30) ensures that the vehicle capacity is not exceeded after the visit of any node during the tour. Finally, constraint set (31) imposes sign restrictions. The problem on hand is clearly NP-hard, as it involves PDPTW, which is NP-hard.

CHAPTER 4

COMPUTATIONAL RESULTS

In this chapter, we analyze the results of our computational experimentation. We first explain the details of our experiment design for evaluating the performance of our model presented in Section 3.3. Then in Section 4.2, we present our computational experiment results.

4.1. Experiment Design

Client and donor addresses are generated uniformly over a 75 by 75 grid, by independently generating x and y coordinates. The depot's position is assumed to be fixed at the coordinates (25, 25). Travel times are computed using the rectilinear distance metric; the longest travel time between any two nodes on the grid is assumed to be 15 minutes (computed as (75+75)/10). All travel times are calculated accordingly.

As a client node's position is generated, its three attribute levels are also generated randomly. For each client, an income level is generated as 1, 2, or 3. Another attribute is used to determine the age level of the client (if the client lives alone) as one of three levels, or the level of the number of household. A third attribute is generated similarly for identifying previous service use.

As it was stated before, each request of a client and each donated item are generated as separate nodes, as each matching may receive different scores from the criteria according to the status of the requested item. We therefore assume that a single item is demanded by each client,

although several requests may correspond to the same address. It is also allowed that a donor can request an item at the same time, or a receiver can donate an item to the center.

Demanded and donated items are generated as one of four types: Appliances and electronics (washing machine, TV set, etc.), furniture (carpet, sofa, etc.), clothing, and house textile (pillow, blanket, curtain, etc.). The service times at each node are computed based on the item type. While delivering or picking up house textile or clothing is assumed to take a time of 5 minutes, furniture requires a service time of 10 minutes, and appliances or electronics require 7.5 minutes. These service times are defined for both donor nodes and client nodes, hence should be added twice for a pickup-delivery pair. The total time (t_{ij}) between any two nodes is then found as the sum of the travel time and the service time.

With each donated item, its age attribute is generated as well, belonging to one of three levels. The generated attribute list and levels are provided in Table 6, along with weights coming from AHP, as it was explained in the problem definition chapter.

Table 6 Generated attributes and their levels

Attribute/Level	*Level 1*	*Level 2*	*Level 3*	**Weight**
Travel time	btw. 10-15 min.	btw. 5-10 min.	< 5 min.	0.06
Income	>1000 TL	btw. 500-1000 TL	< 500 TL	0.49
Age of resident	btw. 20-50	btw. 50-70	> 70	0.13
# of household	2	btw. 3-5	> 5	0.13
# of service received	> 7	btw. 4-6	< 4	0.29
Item age	> 10	btw. 5-10	< 5	0.03

Small and large problem instance sizes are determined in (*Client, Donor*) pairs as follows: (3,5), (4,6), (4,7), (5,7), (6,7), (7,7), (5,10), (5,15), (10,15), (10,20), (15,20), (15,25), where the first six represent small problems and the rest represent larger ones. Hence, 12 different levels are set for the network size (including the depot). The number of items of each type from the donors and the items and at the depot are generated so that they can cover all requests of the working day; hence feasibility of the problem instance is maintained.

As it can be seen from the (*Client, Donor*) pair levels, the number of clients to be served during a working day is less than the number of donors. This is in line with the current practice of

the donation center. After the data is generated in this manner, an appropriate number of dummy depot nodes are created to balance the model; i.e. to assign the remaining donated items. For instance, for the (5,10) setting, 5 clients receive 5 of the donated items, of which some may be at the depot. For assigning the 5 remaining items, 5 dummy depot nodes are created. In addition, two artificial nodes are created to represent source and sink nodes. Hence, the total number of nodes in the mathematical model becomes $2d+2$, where d is the number of donors.

A working day is assumed to be 12 hours, therefore the time windows for all nodes are assumed to be in the interval [0,12]. However, in practice, the vehicle is allowed to finish its workday only after all pickups and deliveries of the day are completed. The time windows of the client nodes generated at 3 levels: 3-hour time windows, 6-hour time windows, and a time window having a random length in the set {3, 6, 9, 12}. The 3-hour time windows are uniformly distributed in one of the four quarters of the working day; i.e., [0,3), [3,6), [6,9), [9,12). Similarly, 6-hour time windows are uniformly distributed in one of these intervals: [0,6), [3,9), [6,12). The random-length time windows are generated such that the interval contains one or more consecutive quarters of the working day.

The time windows of the donor nodes are assumed to be stricter, and are set at 2 levels, having a length of 2 or 3 hours. Furthermore, we assume that the starting time of the time window can take any value within the day. Therefore, we generate a starting time (a_i) uniformly within the day; within the interval [0,10) for 2-hour time windows, and within [0,9) for 3-hour time windows.

The capacity of the single vehicle is assumed to be 100 units in volume, and the volumes of the item types are assumed to be fixed as provided in Table 7. We a assume unit penalty for time violations.

Table 7 Item size parameter

Item type	*Volume (units)*
Appliances and electronics	15
Furniture	30
Clothing	8
House textile	10

Based on the above parameter definitions, a full factorial experiment design is established by 12x3x2 = 72 parameter combinations. 10 problem instances are generated from each combination, totaling up to 720 instances. An example for the generated input files is provided in Appendix 1, and its legend in Appendix 2.

4.2. Computational Results

In order to test the limits of the model in Section 3.3, the generated instances are solved with CPLEX 10 solver in GAMS 23.9.4, using an Intel i7, 2.3 GHz PC with 8 GB RAM. In this section, we present the results of these experiment runs. An output file example and its legend are provided in Appendices 3 and 4, respectively.

4.2.1. Results with identical objective function coefficients

We first test our model with identical objective function coefficients for the three components in the objective function of the model. Recall that the objective function was expressed by the following:

$$\text{Maximize } \sum_{i=1}^{d}\sum_{j=d+1}^{n} u_{ij}x_{ij} - \sum_{i=1}^{n} p_i\left(e_i + l_i\right) - s_{n+1}$$

In fact, with this function, we try to achieve three objectives simultaneously; maximization of total utility, minimization of total time penalty, and minimization of total tour time. In its current form, the objective function coefficients for these three separate and conflicting objectives are identical, that is, one. No scaling is considered for any component. We evaluate the performance of our model by running it for all generated problem instances. The GAMS code for the model is provided in Appendix 5. We impose a 1200-second time limit on the solution time of all instances for the practicality reasons, as computation times were very long in the pilot runs. We first present and discuss the performance of our model on small problem instances. Tables 8 and 9 present the results of computational runs for these instances. In both tables, “Clients”, “Donors”, “Cwin” and “Dwin” columns specify the setting of the problem

instance in terms of number of clients, number of donors and experiment levels for client and donor time windows.

Table 8 Results regarding problem size and computation time for small instances

Clients	Donors	Cwin	Dwin	#Columns	#Rows	#Nonzeroes	#Nodes	Time (s.)
3	5	1	1	206	315	1111	10	67.4
			2	206	315	1108	10	35.0
		2	1	206	315	1108	10	73.3
			2	206	315	1109	10	115.8
		3	1	206	315	1110	10	43.7
			2	206	315	1109	10	104.4
4	6	1	1	282	430	1559	12	639.5
			2	282	430	1560	12	1100.0
		2	1	282	430	1561	12	1149.0
			2	282	430	1561	12	1175.7
		3	1	282	430	1560	12	1151.2
			2	282	430	1560	12	1180.6
4	7	1	1	370	563	2084	14	1200.0
			2	370	563	2084	14	1200.0
		2	1	370	563	2084	14	1200.0
			2	370	563	2083	14	1200.0
		3	1	370	563	2084	14	1200.0
			2	370	563	2083	14	1200.0
5	7	1	1	370	563	2084	14	1200.0
			2	370	563	2088	14	1200.0
		2	1	370	563	2086	14	1200.0
			2	370	563	2087	14	1200.0
		3	1	370	563	2087	14	1200.0
			2	370	563	2089	14	1200.0
6	7	1	1	370	563	2093	14	1200.0
			2	370	563	2093	14	1200.0
		2	1	370	563	2091	14	1200.0
			2	370	563	2088	14	1200.0
		3	1	370	563	2091	14	1200.0
			2	370	563	2090	14	1200.0
7	7	1	1	370	563	2094	14	1200.0
			2	370	563	2095	14	1200.0
		2	1	370	563	2093	14	1200.0
			2	370	563	2094	14	1200.0
		3	1	370	563	2093	14	1200.0
			2	370	563	2096	14	1200.0
							Avg.	989.9

Table 9 Solution results for small instances

Clients	Donors	Cwin	Dwin	#Opt	ObjVal	TourTime	TUtil	TPenalty	UB	RelGap
3	5	1	1	10	-480.2	466.8	148.0	161.4	-480.2	0.0%
			2	10	-375.7	396.6	142.1	121.2	-375.7	0.0%
		2	1	10	-315.8	386.8	185.6	114.5	-315.8	0.0%
			2	10	-319.0	400.0	155.5	74.5	-319.0	0.0%
		3	1	10	-451.6	455.0	146.4	143.0	-451.6	0.0%
			2	10	-346.9	398.9	155.1	103.1	-346.9	0.0%
4	6	1	1	8	-393.4	530.9	203.0	65.5	-374.2	6.0%
			2	2	-370.6	494.4	198.7	74.9	-228.3	36.8%
		2	1	1	-352.8	473.5	214.2	93.5	-114.6	75.6%
			2	1	-301.9	443.1	190.8	49.5	-49.3	95.1%
		3	1	3	-300.4	443.0	212.6	70.0	-62.4	89.2%
			2	1	-288.6	438.8	187.2	37.0	-63.8	130.0%
4	7	1	1	0	-393.6	547.3	217.3	63.6	138.2	143.6%
			2	0	-362.1	543.4	213.0	31.7	158.3	145.6%
		2	1	0	-386.4	530.4	222.2	78.2	162.1	154.5%
			2	0	-287.7	454.6	218.3	51.4	171.0	194.9%
		3	1	0	-342.8	532.2	203.0	13.6	199.5	166.6%
			2	0	-285.9	460.5	218.2	43.6	200.7	178.6%
5	7	1	1	0	-342.5	508.3	255.5	89.7	78.5	126.2%
			2	0	-323.0	497.5	256.9	82.4	129.3	161.4%
		2	1	0	-327.0	491.9	238.5	73.6	201.5	178.6%
			2	0	-213.8	431.0	265.6	48.4	252.1	237.7%
		3	1	0	-331.4	519.6	256.1	67.9	191.7	172.0%
			2	0	-275.6	449.5	258.3	84.4	222.7	337.0%
6	7	1	1	0	-347.7	576.4	283.0	54.3	21.4	148.2%
			2	0	-348.6	560.3	307.5	95.8	85.3	134.5%
		2	1	0	-240.7	472.8	316.0	83.9	227.0	259.4%
			2	0	-161.3	439.8	342.8	64.3	267.1	186.7%
		3	1	0	-316.6	490.6	309.3	135.3	159.9	227.2%
			2	0	-214.6	482.0	323.1	55.7	268.7	241.2%
7	7	1	1	0	-289.1	553.2	400.3	136.2	-20.8	112.6%
			2	0	-228.8	546.1	366.8	49.5	88.9	139.9%
		2	1	0	-197.0	458.4	407.9	146.5	263.6	718.7%
			2	0	-137.2	446.2	352.5	43.5	274.3	269.6%
		3	1	0	-235.0	542.3	363.3	56.0	309.6	295.6%
			2	0	-96.1	447.5	397.7	46.3	374.4	463.9%
			Total	76	**Avg.**	480.8	253.7	77.9		161.9%

In Table 8, “#Columns”, “#Rows”, “#Nonzeroes” and “#Nodes” are taken from the GAMS output of each model. These parameters, namely the number of columns and rows in each model, as well as the number of nonzero variables and the total number of nodes of the problem, give an idea about the growth of problem size as the experimental settings change. Each row in the tables belongs to a different problem setting. “Time” column lists solution times, averaged over 10 instances having the same problem setting.

From Table 8, we can observe that the size of the problem is determined by the number of donors, as the total number of nodes in the model is computed as exactly two times the number of donors. Recall that if there is a difference between the number of donors and clients, as is almost always the case, dummy clients are created to be able to make a one-to-one matching. The number of columns and rows in the model are also determined by this parameter as a result. Observing solution times, we can see that the problem instances cannot be solved within the given time limit of 1200 seconds when there are 14 nodes in the problem network, which corresponds to problem instances having 7 donors (excluding the source and sink). Analyzing the times for (4,6) setting with 12 nodes one can safely argue that, this size is probably the upper limit for obtaining exact solutions to the developed model under 20 minutes.

In Table 9, “#Opt” column presents the number of instances (out of 10) for which an optimal solution could be obtained within the given time limit, whereas “ObjVal” lists the obtained average objective function value of the 10 instances. Note that, if the instance is solved to optimality, this value is the optimal solution. On the other hand, if the model could not be solved within the time limit, the provided value is a lower bound. The “TourTime”, “TUtil” and “TPenalty” columns present the tour times, total utility values and total time penalty values averaged over 10 instances, representing different components of our objective function. Under the “UB” columns, the averages of the best upper bound values obtained by GAMS are listed. If all problem instances are solved to optimality, this value is equal to the objective function value, and the relative gap, provided by the “RelGap” column is zero. Otherwise, the gap is a positive percentage, computed as 100*(UB-ObjVal) / ObjVal.

Although the problem can be solved optimally with up to 10 nodes, the relative gap values seem to grow rapidly after this problem size, looking at the results in Table 9. This is a

strong indication of the combinatorial nature of our problem. More nodes lead to a much larger solution space, and the routing component of the model, which is NP-hard on its own, becomes the determinant of long solution times as well as much larger values of relative gap. Only 76 of the 360 instances were solved to optimality, and the average relative gap was around 162% over all instances.

While 8 of the 10 instances were solved to optimality with the (4,6,1,1) setting, only 2 optimal solutions could be obtained with the (4,6,1,2) setting. Also, the average relative gaps of the instances having 2-hour donor time windows seem to be less than the gaps of the ones with 3-hour time windows. This is an expected result, as longer time windows broaden the solution space and make the problem more difficult. This effect can also be observed subtly in Table 8 in terms of solution times. The same argument can be made for client window settings. From Table 8 and 9, one can observe that the instances having 3-hour client time windows (setting 1) can be solved at shorter times on average than the one having 6-hour windows (setting 2).

In practice, the daily instance sizes at the donation center are in accordance with our small instance sizes most of the time. However, we also analyze larger problem instances to test the limits of our model. We present the results of the runs for these larger instances in Tables 10 and 11.

From Table 10, we can observe that none of the large problem instances could be solved to optimality within the given time limit. This is an expected result for instances with 20 or more nodes, regarding the performance with smaller instances. Note from Table 8 that instances with 14 nodes could not be solved in less than 20 minutes. For 20 instances in (15,20) setting, and 40 instances in (15,25) setting, GAMS could not even obtain a single feasible solution within the time limit. The corresponding rows are marked with "NA" in the tables.

The combinatorial nature of the problem reveals itself in the high relative gap values in Table 11, with and overall average of 257.5%. As the number of nodes increase, the routing component of the problem determines solution times, and results in larger relative gap values.

Table 10 Results regarding problem size and computation time for large instances

Clients	Donors	Cwin	Dwin	#Columns	#Rows	#Nonzeroes	#Nodes	Time (s.)
5	10	1	1	706	1070	4114	20	1200.0
			2	706	1070	4113	20	1200.0
		2	1	706	1070	4111	20	1200.0
			2	706	1070	4113	20	1200.0
		3	1	706	1070	4113	20	1200.0
			2	706	1070	4114	20	1200.0
5	15	1	1	1506	2275	9008	30	1200.0
			2	1506	2275	9008	30	1200.0
		2	1	1506	2275	9008	30	1200.0
			2	1506	2275	9010	30	1200.0
		3	1	1506	2275	9009	30	1200.0
			2	1506	2275	9008	30	1200.0
10	15	1	1	1506	2275	9034	30	1200.0
			2	1506	2275	9038	30	1200.0
		2	1	1506	2275	9039	30	1200.0
			2	1506	2275	9036	30	1200.0
		3	1	1506	2275	9035	30	1200.0
			2	1506	2275	9040	30	1200.0
10	20	1	1	2606	3930	15834	40	1200.0
			2	2606	3930	15836	40	1200.0
		2	1	2606	3930	15839	40	1200.0
			2	2606	3930	15837	40	1200.0
		3	1	2606	3930	15835	40	1200.0
			2	2606	3930	15833	40	1200.0
15	20	1	1	2606	3930	15868	40	1200.0
			2	NA	NA	NA	NA	1200.0
		2	1	NA	NA	NA	NA	1200.0
			2	2606	3930	15876	40	1200.0
		3	1	2606	3930	15881	40	1200.0
			2	2606	3930	15864	40	1200.0
15	25	1	1	NA	NA	NA	NA	1200.0
			2	NA	NA	NA	NA	1200.0
		2	1	NA	NA	NA	NA	1200.0
			2	4006	6035	24575	50	1200.0
		3	1	4006	6035	24567	50	1200.0
			2	NA	NA	NA	NA	1200.0
							Avg.	1200.0

Table 11 Solution results for large instances

Clients	Donors	Cwin	Dwin	#Opt	ObjVal	TourTime	TUtil	TPenalty	UB	RelGap
5	10	1	1	0	-386.6	561.3	252.2	77.5	198.1	161.5%
			2	0	-347.7	573.3	252.4	26.7	229.2	171.1%
		2	1	0	-294.7	531.9	269.5	32.2	270.0	212.4%
			2	0	-285.1	510.7	246.8	21.2	247.5	200.5%
		3	1	0	-325.8	568.0	253.4	11.2	253.6	181.6%
			2	0	-275.7	526.0	270.4	20.0	270.8	206.5%
5	15	1	1	0	-962.2	608.8	285.9	639.3	286.8	142.5%
			2	0	-564.0	593.2	258.5	229.3	243.0	155.7%
		2	1	0	-646.3	612.4	261.9	295.8	265.6	150.5%
			2	0	-514.9	562.8	246.4	198.6	249.7	161.4%
		3	1	0	-668.9	620.7	241.0	289.2	225.3	147.3%
			2	0	-479.8	577.2	266.6	169.2	268.6	161.1%
10	15	1	1	0	-883.8	632.6	522.1	773.3	476.3	193.2%
			2	0	-237.6	589.3	529.8	178.1	518.7	697.2%
		2	1	0	-354.3	617.0	551.1	288.3	554.3	652.8%
			2	0	-353.5	623.2	556.7	287.0	559.9	577.6%
		3	1	0	-307.0	618.1	476.4	166.2	474.7	559.2%
			2	0	-364.5	629.6	472.2	207.0	454.4	287.2%
10	20	1	1	0	-1413.6	636.3	534.9	1312.3	509.2	157.9%
			2	0	-1224.1	641.7	493.9	1076.3	422.3	134.1%
		2	1	0	-1027.1	678.2	520.3	872.3	526.1	198.9%
			2	0	-618.9	610.7	517.8	526.0	525.3	196.9%
		3	1	0	-712.3	674.4	544.3	582.2	539.9	185.3%
			2	0	-1418.9	649.7	523.1	1292.3	527.0	155.0%
15	20	1	1	0	-2992.2	708.5	772.4	3056.0	775.5	168.7%
			2	0	NA	NA	NA	NA	NA	NA
		2	1	0	NA	NA	NA	NA	NA	NA
			2	0	-639.7	692.3	758.6	706.0	765.8	357.6%
		3	1	0	-971.6	666.5	768.5	1073.5	773.0	436.0%
			2	0	-652.8	653.0	876.2	876.0	882.2	468.3%
15	25	1	1	0	NA	NA	NA	NA	NA	NA
			2	0	NA	NA	NA	NA	NA	NA
		2	1	0	NA	NA	NA	NA	NA	NA
			2	0	-3401.1	767.0	792.9	3427.0	805.5	123.7%
		3	1	0	-3793.8	707.0	841.2	3928.0	856.5	122.6%
			2	0	NA	NA	NA	NA	NA	NA
			Total	0	**Avg.**	621.4	471.9	754.6		257.5%

4.2.2. Results with non-identical objective function coefficients

In this section, we present the results of performance evaluation of the model in the presence of non-identical objective function coefficients for the three components. In particular, we test our model for scenarios where one of the objective function components is significantly more important than others. For this purpose, we test the following objective functions for the model:

Z(*U*): Maximize $100\sum_{i=1}^{d}\sum_{j=d+1}^{n} u_{ij}x_{ij} - \sum_{i=1}^{n} p_i\left(e_i + l_i\right) - s_{n+1}$

Z(P): Maximize $\sum_{i=1}^{d}\sum_{j=d+1}^{n} u_{ij}x_{ij} - 100\sum_{i=1}^{n} p_i\left(e_i + l_i\right) - s_{n+1}$

Z(T): Maximize $\sum_{i=1}^{d}\sum_{j=d+1}^{n} u_{ij}x_{ij} - \sum_{i=1}^{n} p_i\left(e_i + l_i\right) - 100s_{n+1}$

While the first objective *Z*(*U*) favors utility maximization much more than the other two objective function components, the second and the third objectives do the same for penalty and tour time minimization, respectively. For illustrative purposes, and without loss of generality, we present the results of this experimentation only for the smallest problem instances, namely, the problem instances "c03d05cw1dw1PR01.txt" through "c03d05cw1dw1PR10.txt", having 3 clients and 5 donors with time window settings level 1. The larger problem instances take too much solution time and behave identically with the smaller instances.

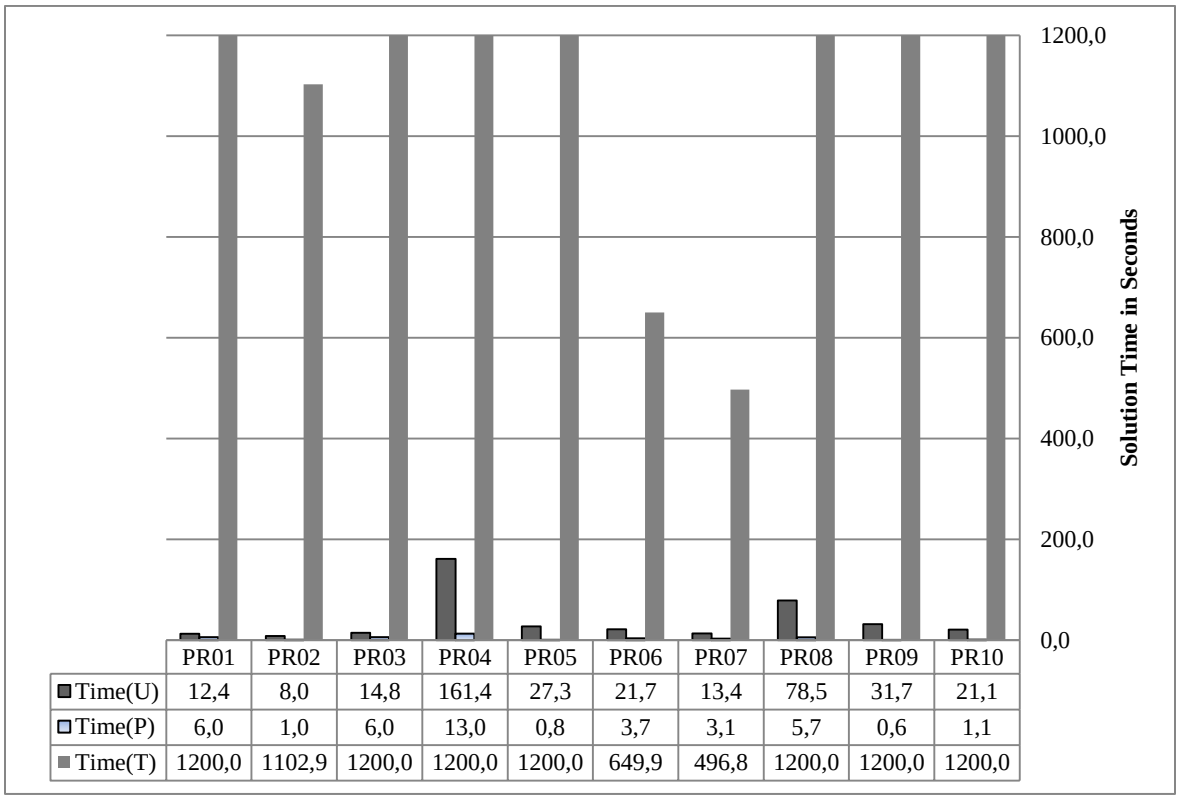
First, we present the summary of the results in Table 12, then we analyze performance in terms of each measure in detail. The results are compared in terms of solution times, as well as the values for the separate objectives of the model. The "time" columns list the solution times (in seconds) for each of the ten problem instances under each of the three objective functions, while "Tourtime" columns show the total time of the tour (in minutes) provided by each solution, "TUtil" list the value of the total utility component of the objective and "TPenalty" provide the total penalty for each solution.

Table 12 Summary of the results with non-identical objective function coefficients

	Utility Coefficient = 100				Penalty Coefficient = 100				Tour Time Coefficient = 100			
Problem Instance	**Time**	**TourTime**	**TUtil**	**TPenalty**	**Time**	**TourTime**	**TUtil**	**TPenalty**	**Time**	**TourTime**	**TUtil**	**TPenalty**
c03d05cw1dw1PR01	12.4	415	128.7	75	6.0	490	128.7	0	1200.0	137	128.7	1492
c03d05cw1dw1PR02	8.0	554	176.7	58	1.0	612	176.7	0	1102.9	119	176.7	1787
c03d05cw1dw1PR03	14.8	547	150.3	135	6.0	547	150.3	135	1200.0	120	144	2107
c03d05cw1dw1PR04	161.4	488	155.7	275	13.0	493	152.1	146	1200.0	143	152.1	1437
c03d05cw1dw1PR05	27.3	386	139.8	162	0.8	548	139.8	0	1200.0	134	139.8	1034
c03d05cw1dw1PR06	21.7	582	160.5	263	3.7	582	160.5	263	649.9	125	160.5	2699
c03d05cw1dw1PR07	13.4	390	159.3	158	3.1	548	159.3	0	496.8	111	159.3	1234
c03d05cw1dw1PR08	78.5	538	171.6	295	5.7	552	168	278	1200.0	126	166.2	1929
c03d05cw1dw1PR09	31.7	411	139.5	136	0.6	547	139.5	0	1200.0	134	137.7	1279
c03d05cw1dw1PR10	21.1	435	106.5	110	1.1	545	106.5	0	1200.0	145	98.4	1446
average	39.0	475	148.9	166.7	4.1	546	148.1	82.2	1065.0	129	146.3	1644.4
minimum	8.0	386	106.5	58.0	0.6	490	106.5	0.0	496.8	111	98.4	1034.0
maximum	161.4	582	176.7	295.0	13.0	612	176.7	278.0	1200.0	145	176.7	2699.0

Let us first examine the results in terms of solution times. The following graph in Figure 2 illustrates the results visually. Time(U), Time(P) and Time(T) represent the solution times for models with objectives $Z(U)$, $Z(P)$, and $Z(T)$, respectively.

From the graph in Figure 2, it can be clearly seen that the more emphasis is given on the routing part of the objective (that is, minimization of the tour time), the more time it takes to find the optimal solution. One can easily observe that seven of the problem instances could not be solved in the 1200-second time limit with objective $Z(T)$, while the smallest solution times are clearly obtained with objective $Z(P)$, as it can also be observed from Table 12. This is mainly because of the fact that the computational difficulty of our integrated routing-assignment problem stems from the routing part. As the importance of this part grows, the difficulty grows dramatically. The average solution time is 1065 seconds for objective $Z(P)$. Objective $Z(U)$ looks for a feasible tour with moderate penalty values while giving emphasis on maximizing utility, therefore it can find an optimal solution in much shorter time than objective $Z(T)$, in 39 seconds on the average. When objective $Z(P)$ is favored, it very quickly reaches optimal solutions minimizing penalty by complying with the given time windows for the nodes, making feasible assignments with a feasible tour. The average solution time is 4.1 seconds for this objective, much shorter than others.

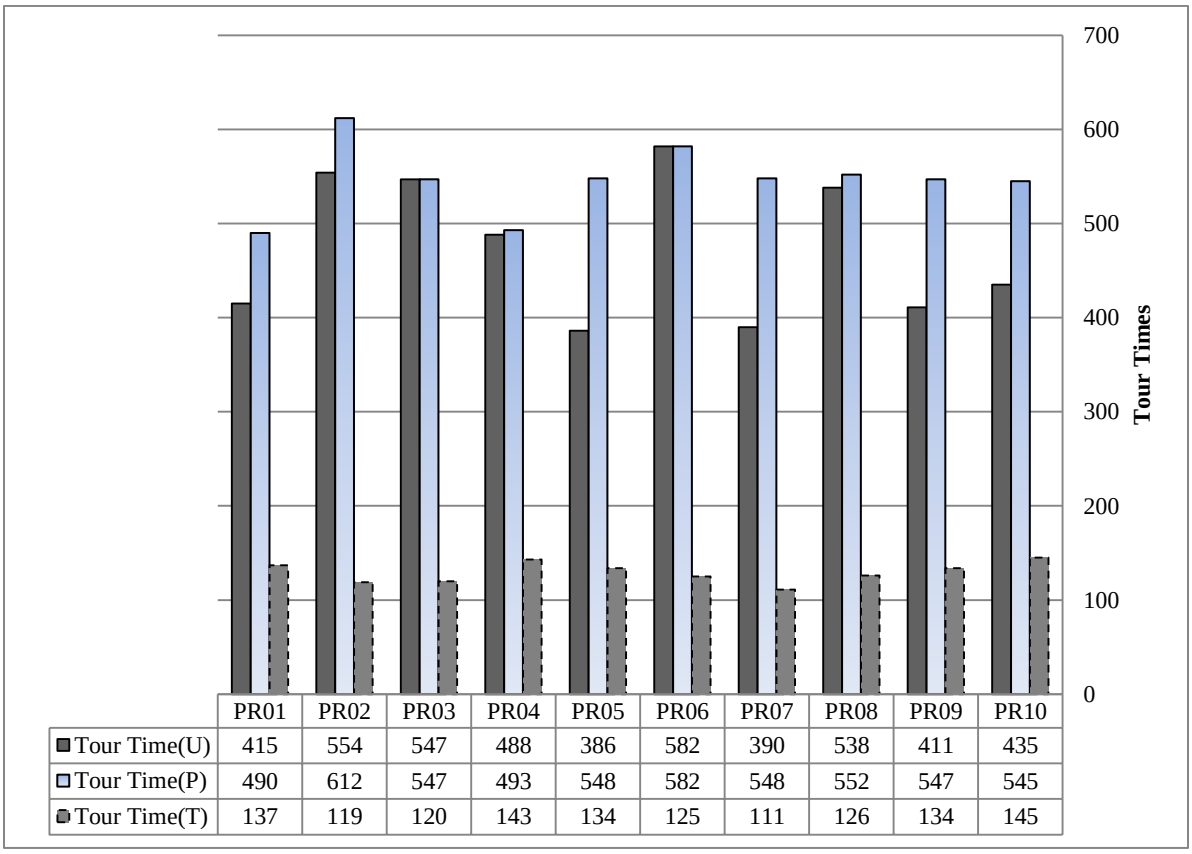


	PR01	PR02	PR03	PR04	PR05	PR06	PR07	PR08	PR09	PR10
Time(U)	12,4	8,0	14,8	161,4	27,3	21,7	13,4	78,5	31,7	21,1
Time(P)	6,0	1,0	6,0	13,0	0,8	3,7	3,1	5,7	0,6	1,1
Time(T)	1200,0	1102,9	1200,0	1200,0	1200,0	649,9	496,8	1200,0	1200,0	1200,0

Figure 2 Comparison of solution times with different objective function coefficients

Next, the results are analyzed based on tour time performances. The following graph in Figure 3 illustrates the results visually. Tour Time(U), Tour Time(P) and Tour Time(T) represent the tour times in the solutions for models with objectives $Z(U)$, $Z(P)$, and $Z(T)$, respectively.

Expectedly, as more emphasis is given to tour time minimization with objective $Z(T)$, the best results are achieved with it (129 minutes on average) at the cost of the significant increase in computation times. This average value is clearly much lower than the one with objective $Z(U)$ (475), or $Z(P)$ 546. Based on the results, it seems that objective $Z(U)$ does better than $Z(P)$ in terms of this measure. This is because of the fact that, while giving a certain emphasis on utility maximization, objective $Z(U)$ gives identical importance to tour time and time penalty. On the other hand, objective $Z(P)$ favors penalty much more than tour time. Hence, the tour times obtained by objective $Z(P)$ are worse than those by $Z(U)$.

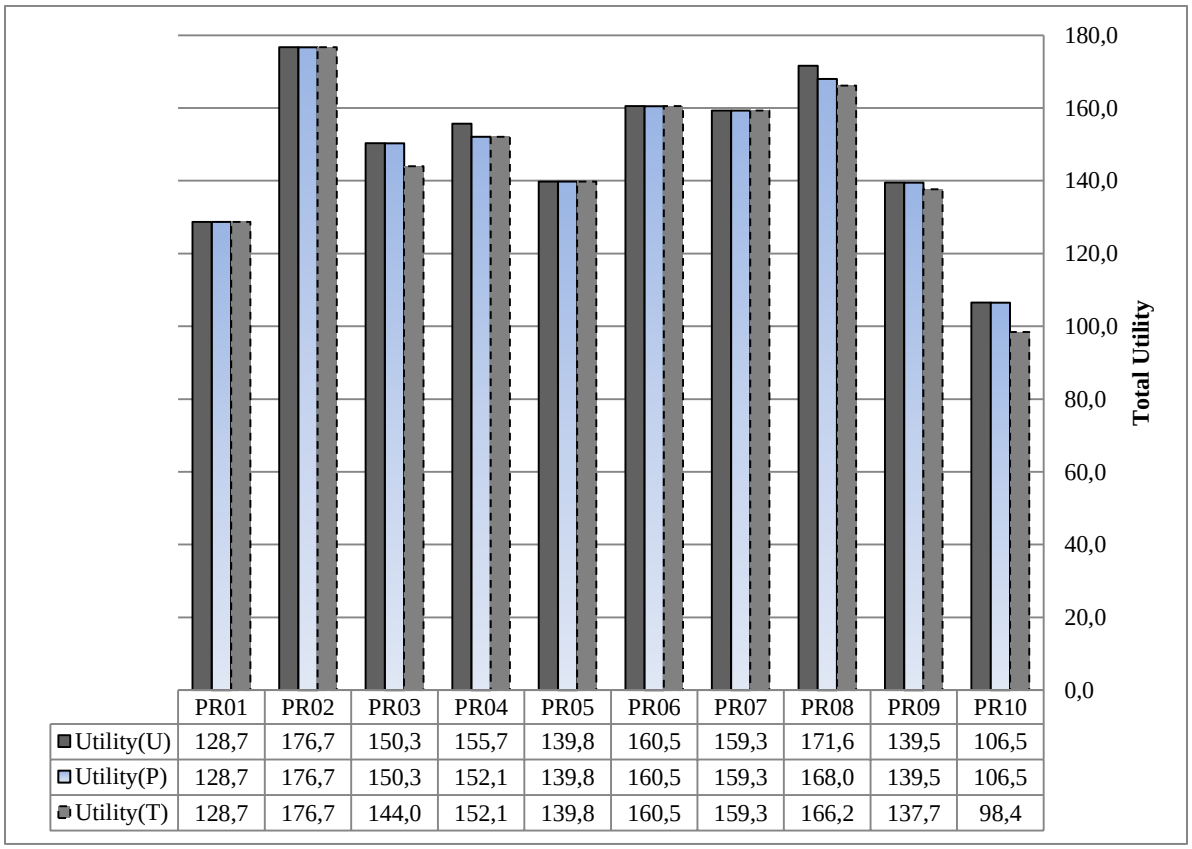


	PR01	PR02	PR03	PR04	PR05	PR06	PR07	PR08	PR09	PR10
Tour Time(U)	415	554	547	488	386	582	390	538	411	435
Tour Time(P)	490	612	547	493	548	582	548	552	547	545
Tour Time(T)	137	119	120	143	134	125	111	126	134	145

Figure 3 Comparison of tour times with different objective function coefficients

We now investigate total utility values. The following graph in Figure 4 illustrates the results visually. Utility(U), Utility(P) and Utility(T) represent the utility values in the solutions for models with objectives $Z(U)$, $Z(P)$, and $Z(T)$, respectively.

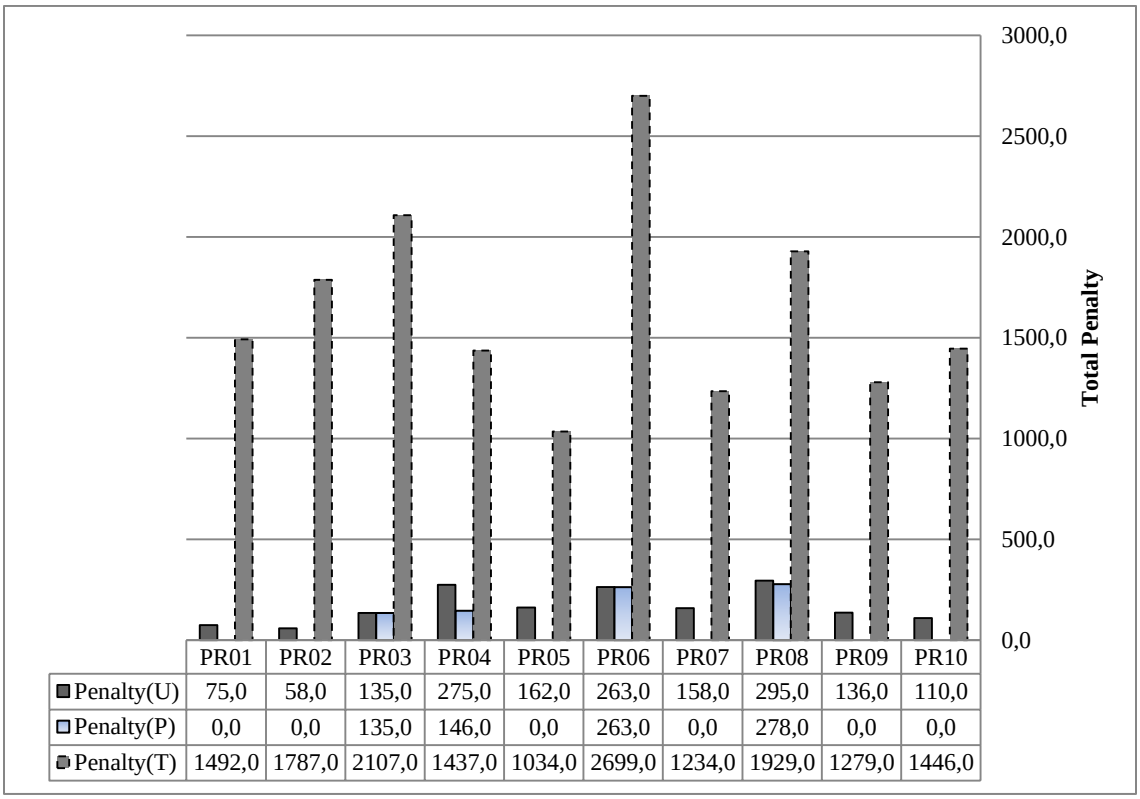
Sensibly, as more emphasis is given to maximizing utility with, the best results are achieved with objective $Z(U)$ (148.9 on average). The values with objective $Z(P)$ and $Z(T)$ are 148.1 and 146.3, respectively. The utility values do not differ significantly between models, as there are few options of alternative assignments in this small problem setting. As the problem size grows, the difference between the total utility values between different objectives is expected to increase. Objective $Z(U)$ seems to be a sensible objective when time violations are not crucial for the decision makers, since it turns out optimal solutions in small computation times, gives the highest utility, and does better than $Z(P)$ in terms of tour time.



	PR01	PR02	PR03	PR04	PR05	PR06	PR07	PR08	PR09	PR10
Utility(U)	128,7	176,7	150,3	155,7	139,8	160,5	159,3	171,6	139,5	106,5
Utility(P)	128,7	176,7	150,3	152,1	139,8	160,5	159,3	168,0	139,5	106,5
Utility(T)	128,7	176,7	144,0	152,1	139,8	160,5	159,3	166,2	137,7	98,4

Figure 4 Comparison of utility values with different objective function coefficients

Finally, we examine the performances of the objectives in terms of total time penalty. The following graph in Figure 5 illustrates the results visually. Penalty(U), Penalty (P) and Penalty (T) represent the penalty values in the solutions. As expected, objective $Z(P)$ performs best in terms of total penalty, with an average penalty value of 82.2 over all instances. Objective $Z(U)$ follows with an average value of 166.7. Objective $Z(T)$ performs much worse than the other two, with an average of 1644.4. Recall that with this objective, the computation times are also much worse, and optimal solutions cannot be achieved within the time limit for many of the instances. The reason why $Z(U)$ works better for penalty minimization than $Z(T)$ is very similar to why it works better for tour time minimization than $Z(P)$. While focusing on utility maximization, objective $Z(U)$ values tour time and time penalty identically. On the other hand, objective $Z(T)$ favors tour time minimization much more than penalty. Hence, the tour times obtained by objective $Z(T)$ are much worse than those by $Z(U)$.



	PR01	PR02	PR03	PR04	PR05	PR06	PR07	PR08	PR09	PR10
Penalty(U)	75,0	58,0	135,0	275,0	162,0	263,0	158,0	295,0	136,0	110,0
Penalty(P)	0,0	0,0	135,0	146,0	0,0	263,0	0,0	278,0	0,0	0,0
Penalty(T)	1492,0	1787,0	2107,0	1437,0	1034,0	2699,0	1234,0	1929,0	1279,0	1446,0

Figure 5 Comparison of total time penalty with different objective function coefficients

Based on the analysis of the results in this section, it can be said that objectives $Z(U)$ and $Z(P)$ perform much better than $Z(T)$ in terms of solution time. The overall performance of these objectives is also better than $Z(T)$ in terms of the other measures analyzed above. Both of these objectives work very well on small problems, hence the decision makers can alter the objective function coefficients on a daily basis to best satisfy their priorities. Objective $Z(U)$ can be used when making the best assignments is crucially important or there are lots of assignment alternatives to choose from. In such cases, it seems that it will perform better than $Z(P)$ in terms of utility and tour time at a cost of extra computation time. On the other hand, when very quick solutions are needed, when the assignment alternatives are few, or when it is critical to meet the time window requirements of donors and clients, objective $Z(P)$ can be very useful. Although this objective allows longer tour times, it can satisfy the daily needs of the donation center in very short computation times. $Z(P)$ also works very well in terms of utility maximization when there are few assignment alternatives for each donated item.

4.2.3. Results with objective Z(P) for small instances

Based on the results of the previous section, additional runs were made with objective $Z(P)$, as this objective yields quick optima. The results are presented in Tables 13 and 14.

Looking at the numerical results and comparing them with the ones in Tables 8 and 9, we can observe the following:

- The number of instances (out of 360) for which an optimal solution was obtained within the time limit has increased from 76 to 269 with this objective.
- The solution times with objective $Z(P)$ are much less compared to the original objective. The overall average for solution time is around 413 seconds with $Z(P)$, whereas it was 990 seconds with the identical-coefficient objective function. Note that the difference in solution times would be even more if a longer time limit was imposed, as most of the instances with the original objective were forcefully terminated at time 1200.
- As a result of the above findings, the relative gap with objective $Z(P)$ has dropped to 42.8% (averaged over all instances) from 161.9%.
- The average tour time obtained with the original objective function over all instances was 480.8 minutes, whereas with objective $Z(P)$ it expectedly increased, and the average was obtained as 530.7 minutes. The reason for this increase was explained in the previous section in detail.
- The performance of the model with objective $Z(P)$ seems excellent in terms of the obtained average total utility value. Compared to the original objective function with an average total utility of 253.2, objective $Z(P)$ yielded 253.7; only a very slight increase and therefore quite satisfactory.
- The total time penalty averaged over all instances was around 78 minutes with the original objective, while it decreased to 30 minutes with objective $Z(P)$, which is less than half of what it was before.

Table 13 Results with $Z(P)$ regarding problem size and computation time for small instances

Clients	Donors	Cwin	Dwin	#Columns	#Rows	#Nonzeroes	#Nodes	Time (s.)
3	5	1	1	206	315	1111	10	2.8
			2	206	315	1108	10	1.3
		2	1	206	315	1108	10	13.2
			2	206	315	1109	10	27.9
		3	1	206	315	1110	10	4.2
			2	206	315	1109	10	10.1
4	6	1	1	282	430	1559	12	7.0
			2	282	430	1560	12	60.9
		2	1	282	430	1560	12	89.3
			2	282	430	1561	12	361.3
		3	1	282	430	1560	12	384.8
			2	282	430	1560	12	259.7
4	7	1	1	370	563	2084	14	289.9
			2	370	563	2084	14	440.2
		2	1	370	563	2084	14	888.8
			2	370	563	2083	14	808.2
		3	1	370	563	2084	14	1045.5
			2	370	563	2083	14	1184.7
5	7	1	1	370	563	2084	14	253.9
			2	370	563	2088	14	336.2
		2	1	370	563	2086	14	368.2
			2	370	563	2088	14	854.9
		3	1	370	563	2087	14	998.5
			2	370	563	2089	14	877.7
6	7	1	1	370	563	2093	14	189.5
			2	370	563	2093	14	389.6
		2	1	370	563	2091	14	397.7
			2	370	563	2088	14	920.7
		3	1	370	563	2092	14	338.1
			2	370	563	2090	14	636.5
7	7	1	1	370	563	2094	14	21.5
			2	370	563	2095	14	65.5
		2	1	370	563	2092	14	588.7
			2	370	563	2095	14	613.0
		3	1	370	563	2093	14	569.6
			2	370	563	2096	14	568.4
							Avg.	413.0

Table 14 Solution results with $Z(P)$ for small instances

Clients	Donors	Cwin	Dwin	#Opt	ObjVal	TourTime	TUtil	TPenalty	UB	RelGap
3	5	1	1	10	-8618.3	546.4	148.1	82.2	-8618.3	0.0%
			2	10	-623.2	515.3	142.1	2.5	-623.2	0.0%
		2	1	10	-1078.1	493.6	185.6	7.7	-1078.1	0.0%
			2	10	-319.0	474.5	155.5	0.0	-319.0	0.0%
		3	1	10	-4342.6	558.6	146.0	39.3	-4342.6	0.0%
			2	10	-1129.0	494.1	155.1	7.9	-1129.0	0.0%
4	6	1	1	10	-3115.9	568.9	203.0	27.5	-3115.9	0.0%
			2	10	-3192.1	540.8	198.7	28.5	-3192.1	0.0%
		2	1	10	-2539.1	562.5	223.4	22.0	-2539.1	0.0%
			2	8	-301.5	492.2	190.8	0.0	-270.8	16.9%
		3	1	8	-2270.0	492.6	212.6	19.9	-2106.4	1.2%
			2	9	-288.6	475.8	187.2	0.0	-262.4	69.6%
4	7	1	1	9	-4046.7	574.0	217.3	36.9	-4033.9	4.4%
			2	9	-1193.7	566.7	213.0	8.4	-1188.7	1.4%
		2	1	3	-4693.8	564.6	220.8	43.5	-4095.6	74.0%
			2	4	-2228.1	486.4	218.3	19.6	-1899.1	64.6%
		3	1	3	-342.6	548.4	205.8	0.0	-41.0	94.2%
			2	3	-1223.3	488.0	224.7	9.6	-825.7	93.7%
5	7	1	1	9	-6440.9	536.4	255.5	61.6	-6398.7	12.3%
			2	8	-4056.9	543.7	266.8	37.8	-3515.3	4.3%
		2	1	8	-515.8	563.6	237.8	1.9	-240.9	37.5%
			2	4	-415.3	485.9	260.6	1.9	-203.2	126.8%
		3	1	2	-4847.7	543.1	255.4	45.6	-4346.3	115.1%
			2	5	-1286.5	545.7	249.2	9.9	-642.6	74.1%
6	7	1	1	9	-5715.1	569.8	284.7	54.3	-5465.1	0.9%
			2	7	-6371.7	568.7	307.1	61.1	-5698.6	12.2%
		2	1	9	-2013.0	563.2	320.3	17.7	-1697.0	3.6%
			2	3	-4804.6	457.0	342.5	95.6	-4316.8	207.9%
		3	1	8	-11615.2	552.2	307.0	113.7	-11530.1	41.0%
			2	7	-660.5	533.3	322.8	4.5	-370.7	58.7%
7	7	1	1	10	-8175.7	576.0	400.3	80.0	-8175.7	0.0%
			2	10	-4236.3	552.7	366.4	40.5	-4236.3	0.0%
		2	1	6	-7826.7	511.5	404.8	77.2	-7408.3	143.9%
			2	5	-960.7	495.9	325.2	7.9	-221.9	90.4%
		3	1	6	-2173.3	576.2	362.9	19.6	-1763.0	127.0%
			2	7	-531.5	488.1	396.6	4.4	-449.6	64.3%
			Total	269	**Avg.**	530.7	253.2	30.3		42.8%

Based on these results, objective $Z(P)$ can be safely recommended for daily usage at the donation center, as it provides very good assignments and minimizes schedule violations in terms of client and donor time window requests. It is also very satisfactory in terms of solution time. A shorter time limit can also be imposed for obtaining even quicker solutions.

The tour time objective can be considered to have the lowest priority as the problem on hand includes a public service, and the humanitarian nature of the problem dictates utility prioritization in this respect. However, sticking to the schedule without violating the time window requirements is also crucial from an operational point of view, as violations may result in disruptions in collection, or collected but non-delivered items. Therefore, this objective may best suit the necessities of the problem's environment.

4.2.4. Results with objective Z(U) for small instances

Although the model with objective $Z(U)$ provides solution in longer times than with $Z(P)$, it seems to be the most appropriate from a humanitarian perspective For this reason, the model was executed with this objective over all 360 small instances. The results are presented in Tables 15 and 16.

Comparing the results with the ones in Tables 8, 9, 13 and 14, we can observe the following:

- The number of instances (out of 360) for which an optimal solution was obtained within the time limit has increased from 101 with this objective, which is between the values of 76 for the original objective and 269 for $Z(P)$.
- The solution times are much longer than those with objective $Z(P)$, and similar to the ones with the original objective.
- Although optimal solutions are obtained only for a third of the instances, the relative gaps found in 1200 seconds are extremely low compared to the previous objectives. Recall that the relative gap (averaged over all instances) with objective $Z(P)$ was 42.8%, and 161.9% with the original objective. With objective $Z(U)$, this gap dropped to 1.1%, which provides a significant result. This small average gap

partially results from the fact that the objective function values and the upper bounds are large positive numbers as a result of the large coefficient of utility in the objective function. The gap values drop significantly due to this scaling.

- As the relative gap values are very small, we can conjecture that many of the incumbent solutions found in twenty minutes may also be optimal. This conjecture can be verified by increasing the time limit.
- The average tour time was 480.8 minutes with the original objective function and 530.7 minutes with objective $Z(P)$. As it was explained in Section 4.2.2, objective $Z(U)$ performs much better than $Z(P)$ in this respect, yielding an overall average of 482.1 minutes, which is very close to the original value.
- The performance of the model with objective $Z(U)$ is expectedly the best among all objectives in terms of average total utility, providing an overall average of 255.3. Recall that the original objective function yielded an average total utility of 253.2, while objective $Z(P)$ yielded 253.7. We can once again conclude that $Z(P)$ is also satisfactory in this regard.
- The total time penalty averaged over all instances was around 93.7 minutes under objective $Z(U)$. It was 78 minutes with the original objective, while it decreased to 30 minutes with objective $Z(P)$. Hence, we can conclude that the higher performance of objective $Z(U)$ in total utility comes with a cost of higher deviations from the time window requirements compared to the other objectives.

Based on these results, objective $Z(U)$ can also be recommended for daily usage at the donation center, as the relative gap values are so small. The model seems to work very well with this objective in terms of utility and tour time values. With even a shorter solution time limit such as 10 or even 5 minutes, the model can provide excellent near-optimal results.

Hence, based on the results from this and the previous section, the decision makers at the donation center can benefit from both objectives, and change them according to their needs and strategies. Other coefficient combinations can also be tried for balancing the three components of the objective.

Table 15 Results with $Z(U)$ regarding problem size and computation time for small instances

Clients	Donors	Cwin	Dwin	#Columns	#Rows	#Nonzeroes	#Nodes	Time (s.)
3	5	1	1	206	315	1111	10	26.7
			2	206	315	1108	10	15.0
		2	1	206	315	1108	10	43.6
			2	206	315	1109	10	55.4
		3	1	206	315	1110	10	28.6
			2	206	315	1109	10	56.8
4	6	1	1	282	430	1559	12	247.2
			2	282	430	1560	12	822.0
		2	1	282	430	1561	12	865.5
			2	282	430	1561	12	994.5
		3	1	282	430	1560	12	796.8
			2	282	430	1560	12	1018.1
4	7	1	1	370	563	2084	14	1200.0
			2	370	563	2084	14	1200.0
		2	1	370	563	2084	14	1200.0
			2	370	563	2083	14	1200.0
		3	1	370	563	2084	14	1200.0
			2	370	563	2083	14	1200.0
5	7	1	1	370	563	2084	14	1188.7
			2	370	563	2088	14	1200.0
		2	1	370	563	2086	14	1200.0
			2	370	563	2087	14	1200.0
		3	1	370	563	2087	14	1200.0
			2	370	563	2089	14	1200.0
6	7	1	1	370	563	2093	14	1200.0
			2	370	563	2093	14	1200.0
		2	1	370	563	2091	14	1200.0
			2	370	563	2088	14	1200.0
		3	1	370	563	2091	14	1200.0
			2	370	563	2090	14	1200.0
7	7	1	1	370	563	2094	14	1104.9
			2	370	563	2095	14	1153.6
		2	1	370	563	2092	14	1200.0
			2	370	563	2094	14	1200.0
		3	1	370	563	2093	14	1200.0
			2	370	563	2096	14	1200.0
							Avg.	933.8

Table 16 Solution results with $Z(U)$ for small instances

Clients	Donors	Cwin	Dwin	#Opt	ObjVal	TourTime	TUtil	TPenalty	UB	RelGap
3	5	1	1	10	14244.7	474.6	148.9	166.7	14244.7	0.0%
			2	10	13730.7	401.2	142.5	118.1	13730.7	0.0%
		2	1	10	18169.6	381.5	186.8	129.9	18169.6	0.0%
			2	10	15179.8	399.6	156.6	83.6	15179.8	0.0%
		3	1	10	14039.0	460.3	146.4	137.7	14039.0	0.0%
			2	10	15042.4	414.6	155.6	104.0	15042.4	0.0%
4	6	1	1	10	19702.6	531.3	203.0	65.1	19702.6	0.0%
			2	6	19351.4	487.6	199.4	100.0	19392.9	0.3%
		2	1	5	20851.0	478.6	214.2	88.4	20917.6	0.4%
			2	4	18675.0	440.1	191.7	58.9	18754.1	0.4%
		3	1	8	20792.0	460.8	213.1	54.2	20820.7	0.1%
			2	5	18340.8	436.5	188.2	42.7	18386.1	0.3%
4	7	1	1	0	21342.5	541.5	220.0	113.0	21847.9	2.5%
			2	0	20822.4	553.7	214.1	35.9	21260.2	2.1%
		2	1	0	21644.8	511.2	222.7	111.0	22177.4	2.5%
			2	0	21351.6	455.9	218.6	51.5	21803.4	2.2%
		3	1	0	20108.8	523.6	206.7	32.6	20588.4	2.5%
			2	0	21496.5	464.1	220.3	69.4	21958.1	2.2%
5	7	1	1	1	25158.3	496.2	258.4	180.5	25532.2	1.6%
			2	0	26131.6	516.6	267.6	110.8	26517.7	1.5%
		2	1	0	23390.1	487.9	239.9	109.0	23833.0	2.0%
			2	0	26223.1	443.7	267.1	47.2	26649.8	1.7%
		3	1	0	25022.5	511.3	256.1	76.2	25519.1	2.0%
			2	0	25295.1	463.4	258.3	70.5	25752.5	1.9%
6	7	1	1	0	27962.1	566.1	286.6	127.8	28279.5	1.2%
			2	0	30210.7	561.5	308.5	76.8	30431.2	0.7%
		2	1	0	31225.4	471.1	318.2	127.5	31657.2	1.4%
			2	0	33926.5	442.3	344.4	74.2	34287.1	1.1%
		3	1	0	30427.5	493.5	310.8	162.0	30874.5	1.5%
			2	0	31857.0	492.8	324.1	62.2	32279.0	1.4%
7	7	1	1	1	39481.5	554.9	401.7	134.6	39677.3	0.5%
			2	1	36147.6	544.4	367.9	95.0	36381.5	0.7%
		2	1	0	40140.0	453.9	407.7	177.1	40555.1	1.1%
			2	0	34858.8	451.7	353.6	48.5	35224.8	1.1%
		3	1	0	36589.8	544.0	371.9	58.2	37085.1	1.4%
			2	0	39323.8	444.2	398.4	73.0	39762.9	1.1%
			Total	101	**Avg.**	482.1	255.3	93.7		1.1%

CHAPTER 5

CONCLUSION

In this thesis, we have considered a real-life assignment and routing problem at a donation center. A mixed integer pickup and delivery model with soft time windows and vehicle capacity restriction was proposed for solving the associated assignment/routing problem.

Parameter combinations were created to reflect the problem environment regarding clients and items' attributes and the levels, the size of the problem network (number of donors and clients), and service time preferences of donors and clients. An experiment design was made including 72 parameter combinations. Ten problem instances were generated from each combination, summing up to 720 instances. In order to test the limits of the proposed model, the generated instances were solved using GAMS and the results were presented in detail for small and large instances of the problem. Different versions of the model were tested based on the weights of the three components in the objective function value, which are total utility, total penalty, and total tour time. In the first version, identical objective function coefficients were assumed for all components. The second and third versions involved penalty prioritization and utility prioritization. A 1200-second solution time limit was imposed and the corresponding performance of the model was analyzed.

With the identical-weight objective function, the model yielded optimal solutions for all instances with up to 6 donors within the given time limit, whereas none of the large problem instances could be solved optimally within this time. Next, on a small set of problem instances, different objective functions were tried. It was observed that giving emphasis to tour time

resulted in much longer solution times, while best results in solution time was obtained with the penalty-prioritizing objective.

Based on this result, additional runs were made with the penalty-prioritizing objective for all of the 360 small instances. From the results, it can be deducted safely that this objective can be recommended for practical use at the donation center, as it performs very well in terms of good assignments, and it minimizes client and donor time schedule violations for operational concerns. When the experiment was repeated for the utility-prioritizing objective, although the number of optimal solutions was less and the solution times were much longer, the overall relative gap values decreased drastically, down to approximately 1%. Consequently, this objective can also be useful when the focus is on highest-utility matching and tour time.

As the problem studied in this thesis is NP-hard, although the developed model can find optimal solutions for the defined assignment/routing problem, the solution times are clearly unacceptable for large problem instances. For such problems, different procedures may be needed for obtaining satisfactory solutions. As a future work, we propose a decomposition of the problem in the following manner. The assignment component of the problem can be solved easily utilizing any commercial optimization software or by the Hungarian algorithm in $O(n^3)$ time. Considering this feature, the problem can be decomposed into its two natural parts while developing the first three of our heuristics; namely, the assignment and the routing components. While the assignment part can be solved optimally using the model explained below, different heuristic approaches can be developed for the routing component of the problem.

The model for the assignment part can be formulated as a basic assignment model with a utility maximization objective. After donor/client pairings are obtained from this model, heuristics can be executed for obtaining the daily route of the single vehicle. The parameters and the decision variable of the assignment model can be defined as follows:

i : indices for the donated items that will be collected in working day:

j : indices for the clients that will be served in the working day:

u_{ij} : the utility of assigning item i to client j defined for i and $j = 1, \ldots, n$,

x_{ij}: 1 if item at node *i* is assigned to node *j*, 0 otherwise.

Similar to our model in Chapter 3, each request of a client can be assumed as a separate record, as it may receive different scores from the criteria according to the status of the item requested. The utility values are not defined for item-depot assignments, and are therefore not included in the objective function. The corresponding assignment model becomes:

Maximize $\sum_{i=1}^{n}\sum_{j=1}^{n} u_{ij} x_{ij}$

subject to:

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1,...,n$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1,\ldots,n$$

$$x_{ij} \in \{0,1\} \qquad \forall i, j$$

The utility maximizing objective function considers all daily assignments. The first constraint set ensures that each item must be assigned to either a client node or the depot, whereas the second assures that all clients that are scheduled for the particular working day will be assigned an item. Once this model is solved, donated item/recipient pairings are formed. After this stage, one of the following simple heuristic approaches can be executed to find a route for distributing the items to their respective recipients.

Nearest-neighbor-based heuristic (NNH):

In this procedure, the typical nearest neighbor idea for the TSP is modified through considering capacity restriction and the assigned node pairs, while penalizing time window violations. With this approach, the vehicle starts its tour from the depot, and visits the nearest feasible node. Feasibility is defined by remaining capacity of the vehicle for donor nodes. If a client node is the nearest node in sequence, the vehicle can only visit that node if it has already visited its assigned

donor. The penalties are updated as the tour is formed, and the algorithm stops once all nodes are visited.

The heuristic is very simple to apply as it visits the nearest neighbor of the current node without considering time window preferences. However, it may tend to have long return times for the vehicle.

Nearest-time-interval-based heuristic (NTH):

This procedure applies the nearest neighbor logic to the time windows of the donors and clients. Therefore, it prioritizes time windows while considering capacity restriction. Putting all unvisited nodes in a list in ascending order of their time window start times, the vehicle starts its tour from the depot and visits nodes in this order, again considering feasibility. The structure of NTH is very similar to NNH, and is also simply applicable. It visits the nodes in the order of their time preferences without explicitly considering travel times; hence the principal motive is to minimize the penalty segment of the objective function.

Other simple heuristics based on insertion or savings can also be adapted for the routing component of the problem. Another future work direction can be converting the earliness, tardiness and tour time computed in each model to utility values, thereby subtracting these from the total utility of assignments to form a single utility maximization objective.

Our problem can be extended/adapted to be used in diverse contexts involving public service or disaster relief situations. The model developed in this study is intended as a basis for such extensions involving assignment and routing decisions simultaneously. Humanitarian problems require urgent attention by the authorities most of the time, and should be solved effectively and efficiently. Therefore, coming up with efficient decomposition or meta-heuristic approaches to this problem or its extensions can be worth studying to solve larger instances.

Another future work may involve considering a multi-objective decision making perspective. This will require obtaining pareto-optimal solutions to this multi-faceted problem. The static problem defined in this study can also be converted into a dynamic one by including the requests coming during the day into the model. This case will require an online solution approach.

BIBLIOGRAPHY

Barbeglia, G., Cordeau, J.F. and Laporte, G. (2010). "Dynamic pickup and delivery problem," *European Journal Operational Research*, 202, 8-15.

Bent, R. and Van Hentenryck, P. (2006). "A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows," *Computers & Operations Research*, 33, 875-893.

Clarke, G. and Wright, J.W. (1964). "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, 12(4), 568-581.

Cordeau, J.F. and Laporte, G. (2003a). "The dial-a-ride problem: Variants, modeling issues and algorithms," *4OR: A Quarterly Journal of Operations Research*, 1, 89-101.

Cordeau, J.F. and Laporte, G. (2003b). "A tabu search heuristic for the static multi-vehicle dial-a-ride problem," *Transportation Research Part B: Methodological*, 37, 579-594.

Diana, M. and Dessouky, M.M. (2004). "A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows," *Transportation Research Part B: Methodological*, 38, 539-557.

Dumitrescu, I., Ropke, S., Cordeau, J.F. and Laporte, G. (2010). "The travelling salesmen problem with pickup and delivery: polyhedral results and branch-and-cut algorithm," *Mathematical Programming Series A*, 121, 269-305.

Hame, L. (2011). "An adaptive insertion algorithm for the single-vehicle dial-a-ride problem with narrow time windows," *European Journal of Operational Research*, 209, 11-22.

Hosny, M.I. and Mumford, C.L. (2010). "The single vehicle pickup and delivery problem with time windows: intelligent operators for heuristic and metaheuristic algorithms," *Journal of Heuristics*, 16, 417-439.

Hosny, M.I. and Mumford, C.L. (2012). "Constructing initial solutions for the multiple vehicle pickup and delivery problem with time windows," *Journal of King Saud University-Computer and Information Sciences*, 24, 59-69.

Huang, Y.H. and Ting, C.K. (2010). "Ant colony optimization for the single vehicle pickup and delivery problem with time windows," *Proceedings of International Conference on Technologies and Applications of Artificial Intelligence,* 537-543.

Ioachim, I., Desrosiers, J., Dumas, Y., Solomon, M.M. and Villeneuve, D. (1995). "A request clustering algorithm for door-to-door handicapped transportation," *Transportation Science*, 29, 63-78.

Jaw, J.J., Odoni, A.R., Psaraftis, H.N. and Rygaard, J.M. (1995). "A heuristic algorithm for the multi vehicle advance request dial-a-ride problem with time windows," *Transportation Research,* 20, 243-257.

Kalantari, B., Hill, A.V. and Adora, S.R. (1985). "An algorithm for the travelling salesman with pickup and delivery customers," *European Journal of Operational Research,* 22, 377-386.

Kammatri, R., Hammadi, S., Borne, P. and Ksouri M. (2004). "A new hybrid evolutionary approach for the pickup and delivery problem with time windows," *Proceedings of IEEE International Conference on Systems, Men and Cybernetics,* 1498-1503.

Lai, M.Y., Liu, C.S. and Tong, X.J. (2010). "A two-stage hybrid metaheuristic for pickup and delivery vehicle routing problem with time windows," *Journal of Industrial and Management Optimization,* 6(2), 435-451.

Landrieu, A., Mati, Y. and Binder, Z. (2001). "A tabu search heuristic for the single vehicle pickup and delivery problem with time windows," *Journal of Intelligent Manufacturing,* 12, 497-508.

Li, H. and Lim, A. (2001). "A metaheuristic for the pickup and delivery problem with time windows," *Proceedings of 13th IEEE ICTAI,* 160-167.

Lu, Q. and Dessouky, M.M. (2005). "A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows," *European Journal of Operational Research,* 175, 672-687.

Madsen, O.B.G., Ravn, H.F. and Rygaard, J.M. (1995). "A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities and multiple objectives," *Annals of Operations Research*, 60, 193-208.

Minic, S.M., Krishnamurti, R. and Laporte, G. (2004). "Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows," *Transportation Research Part B*, 38, 669-685.

Nanry, W.P. and Barnes, J.W. (2000). "Solving the pickup and delivery problem with time windows using reactive tabu search," *Transportation Research Part B*, 34, 107-121.

Ozdemirel, A., Eliiyi, D.T. and Gokgur, B. (2012). "An integrated assignment and routing problem with time windows and capacity restrictions," *Procedia-Social and Behavioral Sciences*, 54, 149-158.

Pankratz, G. (2005). "A grouping genetic algorithm for the pickup and delivery problem with time windows," *OR Spectrum*, 27, 21-41.

Parragh, S.N., Doerner, K.F. and Hartl, R.F. (2008a). "A survey on pickup and delivery problems Part I: Transportation between customers and depot," *Journal für Betriebswirtschaft*, 58, 21-51.

Parragh, S.N., Doerner, K.F. and Hartl, R.F. (2008b). "A survey on pickup and delivery problems Part II: Transportation between pickup and delivery locations," *Journal für Betriebswirtschaft*, 58(2), 81-118.

Pisinger, D. and Ropke, S. (2007), "A general heuristic for vehicle routing problems," *Computers & Operations Research*, 34, 2403-2435.

Psaraftis, H.N. (1980). "A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem," *Transportation Science*, 14(2), 130-154.

Ropke, S. and Cordeau, J.F. (2009). "Branch-and-cut-and-price for the pickup and delivery problem with time windows," *Transportation Science*, 43(3), 267-286.

Ropke, S., Cordeau, J.F. and Laporte, G. (2007). "Models and branch-and-cut algorithms for pickup and delivery problems with time windows," *Networks*, 49(4), 258-272.

Rosenkrantz, D.J., Stearns, R.E. and Lewis P.M. (1974). "Approximate algorithms for the travelling salesperson problem," *IEEE Conference Record of 15th Annual Symposium on Switching and Automata Theory*, 33-42.

Savelsbergh, M.W.P. and Sol, M. (1995). "The general pickup and delivery problem," *Transportation Science*, 29(1), 17-29.

Sexton, T. and Bodin, L.D. (1985a). "Optimizing single vehicle many-to-many operations with desired delivery times: I. Scheduling," *Transportation Science*, 19, 378-410.

Sexton, T. and Bodin, L.D. (1985b). "Optimizing single vehicle many-to-many operations with desired delivery times: II. Routing," *Transportation Science*, 19, 411-435.

Van der Bruggen, L.J.J., Lenstra, J.K. and Schuur, P.C. (1993). "Variable depth-search for the single-vehicle pickup and delivery problem with time windows," *Transportation Science*, 27(3), 298-311.

Wang, H.F. and Chen, Y.Y. (2013). "A coevolutionary algorithm for the flexible delivery and pickup problem with time windows," *International Journal of Production Economics*, 141, 4-13.

Zou, S., Li, J. and Li, X. (2013). "A hybrid particle swarm optimization algorithm for multi-objective pickup and delivery problem with time windows," *Journal of Computers*, 8(10), 2583-2589.

APPENDICES

APPENDIX 1 – An input file example.

INPUT FILE NAME: c03d05cw1dw1PR01.txt

3	5	100	12						
75	75	25	25	4	3				
1	40	32	3	6	2	1	2	0	2
2	32	62	6	9	3	1	1	0	3
3	26	3	6	9	2	2	2	0	3
1	36	66	2	4	2	30	2	20	
2	36	75	4	6	3	8	3	10	
3	25	25	0	12	2	30	3	20	
4	33	19	5	7	2	30	1	20	
5	53	11	8	10	4	10	1	10	

Depot-Donor distribution for client requests: 50-50

4 item types used:

A-Appliances

F-Furniture

C-Clothing

T-House textile

APPENDIX 2– Legend for input files.

File Name: “c05d10cw1dw1s1PR01.txt” means 5 clients, 10 donors, client time window level 1, donor time window level 1, item type size level 1, instance no 1.

client time window levels: 1 (3 hrs.), 2 (6 hrs.), 3 (3,6,9 or 12 hrs.)

Donor time window levels: 1 (2 hrs.), 2 (3 hrs.)

4 item types used:

A-Appliances

F-Furniture

C-Clothing

T-House textile

Item sizes: fixed size: 15 x 30 x 8 x 10

Depot-Donor distribution for client requests: 50% (Probability that each of the 5 donors generated for the 5 clients is depot)

#Clients	#Donors	Vehicle capacity	Planning horizon (hrs.)					
5		10	100			12		
Gridwidth	GridHeight		DepotX-Coord	DepotY-Coord	#itemTypes	ClientAttributeScale		
75		75	25		25	4		3
Client	CoorX	CoorY	TStart	Tend	Type	IncLevel	HHLevel	AgeLevel	ServiceLevel
1	40	32	3	6	2	1	2	0	2
2	32	62	6	9	3	1	1	0	3
3	26	3	6	9	2	2	2	0	3
4	1	53	6	9	2	2	2	0	3
5	53	28	3	6	3	1	3	0	3

Donor	X	Y	TStart	TEnd	Type	Size	AgeLevel	ServiceTime (min.)
1	36	66	2	4	2	30	2	20
2	36	75	4	6	3	8	3	10
3	25	25	0	12	2	30	3	20
4	25	25	0	12	2	30	3	20
5	16	33	4	6	3	8	2	10
6	33	19	5	7	2	30	1	20
7	53	11	8	10	4	10	1	10
8	12	25	10	12	2	30	2	20
9	13	2	8	10	4	10	1	10
10	22	36	8	10	4	10	3	10

APPENDIX 3 – An output file example.

File Name: c05d10cw1dw1s1PR01.gout1

1	1						
-361.30	532.5250	1795796					
181	314	1087					
10	5	108					
411							
1	8	61.00					
2	7	40.60					
3	6	27.10					
4	10	0.00					
5	9	0.00					
0	3	0	0	0	0	0	30
1	4	120	25	300	30	60	30
2	7	342	12	360	8	68	-8
3	1	0	26	120	30	30	30
4	6	300	22	322	30	90	-30
5	10	401	10	411	10	40	-10
6	2	322	20	342	-30	60	8
7	8	360	22	382	-8	60	-30
8	5	382	19	401	-30	30	10
9	11	411	0	411	-30	0	0
10	9	411	0	411	-10	30	-30

APPENDIX 4 – Legend for output files.

File Name: c05d10cw1dw1s1PR01.gout1

1 (model status-optimal) 1 (solver status-optimal) --> Look at GAMS user guide for detailed explanations (MODELSTATUS: 1 OPTIMAL, 4 INFEASIBLE, 8 INTEGER SOLUTION, 14 NO SOLUTION RETURNED – SOLVERSTATUS: 1 NORMAL COMPLETION, 2 ITERATION INTERRUPT, 3 RESOURCE INTERRUPT)

-361.30 (obj value) 532.5250 (time in seconds) 1795796 (# of nodes) -362 Estimated Upper bound 0.00 Relative Gap (Upper Bound - obj value / obj value)

181 (# of columns) 314 (# of rows) 1087 (# of nonzero entries)

10 (2*donors) 5 (donors) 108 (total of donated item volumes including depot)

411 (tour time)

Rows for donors client assignments:

1 8 61.00 (donor 1 assigned to client 8 with utility 61)

2 7 40.60

3 6 27.10

4 10 0.00

5 9 0.00

Rows for route, including source node:

0 3 0 0 0 0 0 30 (source --> donor 3, 0 start time, 0 service time, 0 start time at donor 3, 0 item volume at source, 0 load at source, 30 item volume at donor node 3 - donor node 3 is depot, look from input file)

1 4 120 25 300 30 60 30 (donor 1 --> donor 4, 120 start time, 25 service+travel time, 300 start time at donor 4, 30 item volume at donor 1, 60 load at donor 1, 30 item volume at donor node 4)

2 7 342 12 360 8 68 -8

3 1 0 26 120 30 30 30

4 6 300 22 322 30 90 -30

5 10 401 10 411 10 40 -10

6 2 322 20 342 -30 60 8

7 8 360 22 382 -8 60 -30

8 5 382 19 401 -30 30 10

9 11 411 0 411 -30 0 0

10 9 411 0 411 -10 30 -30

APPENDIX 5 – GAMS code for the model.

```
$title INTEGRATED ROUTING ASSIGNMENT MODEL WITH SOFT TIME CONSTRAINTS - 1-MIP

SETS

k set for the node / 0*11 /

i(k)    / 1*10 /;


alias(i,j,h);

alias(k,p,q);


SCALARS

vCap Vehicle capacity / 100 /

HorizonLength Total length of the time horizon (big M) / 1440 /

d Number of donors / 5 /

c Number of clients / 3 /

n Number of nodes (2d) / 10 /;


Parameter dist(i,j) travel times between nodes

/1.1 = 0, 1.2 = 1, 1.3 = 6, 1.4 = 5, 1.5 = 8, 1.6 = 4, 1.7 = 1, 1.8 = 8, 1.9 = 6, 1.10 = 6

2.1 = 1, 2.2 = 0, 2.3 = 7, 2.4 = 6, 2.5 = 9, 2.6 = 5, 2.7 = 2, 2.8 = 9, 2.9 = 7, 2.10 = 7

3.1 = 6, 3.2 = 7, 3.3 = 0, 3.4 = 2, 3.5 = 5, 3.6 = 3, 3.7 = 5, 3.8 = 3, 3.9 = 0, 3.10 = 0

4.1 = 5, 4.2 = 6, 4.3 = 2, 4.4 = 0, 4.5 = 3, 4.6 = 2, 4.7 = 5, 4.8 = 3, 4.9 = 2, 4.10 = 2

5.1 = 8, 5.2 = 9, 5.3 = 5, 5.4 = 3, 5.5 = 0, 5.6 = 4, 5.7 = 8, 5.8 = 4, 5.9 = 5, 5.10 = 5

6.1 = 4, 6.2 = 5, 6.3 = 3, 6.4 = 2, 6.5 = 4, 6.6 = 0, 6.7 = 4, 6.8 = 5, 6.9 = 3, 6.10 = 3

7.1 = 1, 7.2 = 2, 7.3 = 5, 7.4 = 5, 7.5 = 8, 7.6 = 4, 7.7 = 0, 7.8 = 7, 7.9 = 5, 7.10 = 5

8.1 = 8, 8.2 = 9, 8.3 = 3, 8.4 = 3, 8.5 = 4, 8.6 = 5, 8.7 = 7, 8.8 = 0, 8.9 = 3, 8.10 = 3

9.1 = 6, 9.2 = 7, 9.3 = 0, 9.4 = 2, 9.5 = 5, 9.6 = 3, 9.7 = 5, 9.8 = 3, 9.9 = 0, 9.10 = 0

10.1 = 6, 10.2 = 7, 10.3 = 0, 10.4 = 2, 10.5 = 5, 10.6 = 3, 10.7 = 5, 10.8 = 3, 10.9 = 0, 10.10 = 0

/;

Parameter t(p,q) process times including item service times

/0.1 = 6, 0.2 = 7, 0.3 = 0, 0.4 = 2, 0.5 = 5, 0.6 = 3, 0.7 = 5, 0.8 = 3, 0.9 = 0, 0.10 = 0, 0.11 = 0
```

1.1 = 0, 1.2 = 16, 1.3 = 26, 1.4 = 25, 1.5 = 23, 1.6 = 24, 1.7 = 16, 1.8 = 28, 1.9 = 16, 1.10 = 16, 1.11 = 6

2.1 = 16, 2.2 = 0, 2.3 = 22, 2.4 = 21, 2.5 = 19, 2.6 = 20, 2.7 = 12, 2.8 = 24, 2.9 = 12, 2.10 = 12, 2.11 = 7

3.1 = 26, 3.2 = 22, 3.3 = 0, 3.4 = 22, 3.5 = 20, 3.6 = 23, 3.7 = 20, 3.8 = 23, 3.9 = 10, 3.10 = 10, 3.11 = 0

4.1 = 25, 4.2 = 21, 4.3 = 22, 4.4 = 0, 4.5 = 18, 4.6 = 22, 4.7 = 20, 4.8 = 23, 4.9 = 12, 4.10 = 12, 4.11 = 2

5.1 = 23, 5.2 = 19, 5.3 = 20, 5.4 = 18, 5.5 = 0, 5.6 = 19, 5.7 = 18, 5.8 = 19, 5.9 = 10, 5.10 = 10, 5.11 = 5

6.1 = 24, 6.2 = 20, 6.3 = 23, 6.4 = 22, 6.5 = 19, 6.6 = 0, 6.7 = 19, 6.8 = 25, 6.9 = 13, 6.10 = 13, 6.11 = 3

7.1 = 16, 7.2 = 12, 7.3 = 20, 7.4 = 20, 7.5 = 18, 7.6 = 19, 7.7 = 0, 7.8 = 22, 7.9 = 10, 7.10 = 10, 7.11 = 5

8.1 = 28, 8.2 = 24, 8.3 = 23, 8.4 = 23, 8.5 = 19, 8.6 = 25, 8.7 = 22, 8.8 = 0, 8.9 = 13, 8.10 = 13, 8.11 = 3

9.1 = 16, 9.2 = 12, 9.3 = 10, 9.4 = 12, 9.5 = 10, 9.6 = 13, 9.7 = 10, 9.8 = 13, 9.9 = 0, 9.10 = 0, 9.11 = 0

10.1 = 16, 10.2 = 12, 10.3 = 10, 10.4 = 12, 10.5 = 10, 10.6 = 13, 10.7 = 10, 10.8 = 13, 10.9 = 0, 10.10 = 0, 10.11 = 0

/;

Parameter utility(i,j) utility matrix entries

/1.6 = 25.30, 1.8 = 61.00

2.7 = 40.60

3.6 = 27.10, 3.8 = 59.20

4.6 = 24.40, 4.8 = 56.50

/;

Parameter v(p) volume of node p

/0 0, 1 30, 2 8, 3 30, 4 30, 5 10, 6 -30, 7 -8, 8 -30, 9 -30, 10 -10, 11 0

/;

Parameter a(i) lower time specification of node i

/1 120, 2 240, 3 0, 4 300, 5 480, 6 180, 7 360, 8 360, 9 0, 10 0

/;

Parameter b(i) upper time specification of node i

/1 240, 2 360, 3 1440, 4 420, 5 600, 6 360, 7 540, 8 540, 9 1440, 10 1440

/;

Parameter penalty(i) penalty for node i

/1 1.0, 2 1.0, 3 0.0, 4 1.0, 5 1.0, 6 1.0, 7 1.0, 8 1.0, 9 0.0, 10 0.0

/;

VARIABLES

obj objective function value

BINARY VARIABLES

x(i,j) 1 if item i is assigned to need j

y(p,q) 1 if node q follows node p in the tour : p-->q

POSITIVE VARIABLE

s(p) starting time of service at node p

l(p) load of vehicle after visiting node p

EP(i) amount of time that node i is served early

LP(i) amount of time that node i is served late

;

EQUATIONS

OBJECTIVE maximize total utility - time penalties - tour time

c1_assign(i) each item must be assigned to either a client or the depot

c2_assign(j) a client must be assigned to a donor (including depot items)

c1_2_exclude_assign incompatible items excluded

c3_source_donor

c4_source_client

c5_donor_sink

c6_client_sink

c7_flow(i)

c8_flow(j)

c9_vCap(i) handle the capacity constraint of the vehicle

c10_loadSourceSink(p) vehicle begins and ends the tour empty

c11_earliness(i) calculate earliness of each node

c12_lateness(i) calculate lateness of each node

c13_RouteByAssign(i,j) the service start time for each node due to assignment variables

c14_TimeByRouting(p,q) the service start time for each node due to routing variables

c15_LoadByRouting(p,q) load at each node due to routing variables

;

OBJECTIVE.. obj=e=sum((i,j)$(ord(i) <= d and ord(j) >= d+1),utility(i,j)*x(i,j)) - sum(i, penalty(i)*(EP(i)+LP(i)))- sum(p$(ord(p)>n+1), s(p));

c1_assign(i)$(ord(i) <= d).. sum(j$(((ord(j) > d)and(ord(j) <= d+c)and(utility(i,j)>0))or(ord(j)>d+c)), x(i,j)) =e= 1;

c1_2_exclude_assign.. sum((i,j)$((ord(i) <= d) and (ord(j) <= d+c) and (utility(i,j)=0)), x(i,j)) =e= 0;

c2_assign(j)$(ord(j) >= d+1).. sum(i$(ord(i) <= d), x(i,j)) =e= 1;

c3_source_donor.. sum(p$(ord(p) <= d+1 and ord(p) >= 2), y("0",p)) =e= 1;

c4_source_client.. sum(p$(ord(p) >= d+2), y("0",p)) =e= 0;

c5_donor_sink.. sum((p,k)$(ord(p) <= d+1 and ord(p) >= 2 and ord(k)>n+1), y(p,k)) =e= 0;

c6_client_sink.. sum((p,k)$(ord(p) >= d+2 and ord(p) <= n+1 and ord(k)>n+1), y(p,k)) =e= 1;

c7_flow(i).. sum(k$(ord(i)+1<>ord(k) and ord(k)>1),y(i,k)) =e= 1;

c8_flow(j).. sum(k$(ord(k)-1<>ord(j) and ord(k)<=n+1),y(k,j)) =e= 1;

c9_vCap(i).. l(i) =l= vCap;

c10_loadSourceSink(p)$(ord(p)=1 or ord(p)>n+1).. l(p) =e= 0;

c11_earliness(i).. EP(i) =g= a(i) - s(i);

c12_lateness(i).. LP(i) =g= s(i) - b(i);

c13_RouteByAssign(i,j)$(ord(i) <= d and ord(j) >= d+1).. s(j) =g= s(i) + smin(h, dist(i,h)) - HorizonLength*(1-x(i,j));

c14_TimeByRouting(p,q)$(ord(p)<>ord(q) and ord(p) <= n+1 and ord(q) >= 2).. s(q) =g= s(p) + t(p,q)- HorizonLength*(1-y(p,q));

c15_LoadByRouting(p,q)$(ord(p)<>ord(q) and ord(p) <= n+1 and ord(q) >= 2).. l(q) =g= l(p) + v(q)- 2*vCap*(1-y(p,q));

MODEL IntegratedAssignment /ALL/;

option optcr = 0, optca = 0, iterlim = 99999999;

option reslim = 1200;

SOLVE IntegratedAssignment USING MIP MAXIMIZING obj;

execerror=0;

```
file textOutput /c03d05cw1dw1PR01.gout1/;

textOutput.pc=6;

put textOutput, IntegratedAssignment.modelstat:0:0, IntegratedAssignment.solvestat:0:0 /;

put IntegratedAssignment.objVal:0:2, IntegratedAssignment.resUsd:0:4, IntegratedAssignment.nodUsd:0:0, IntegratedAssignment.objEst:0:2 /;

put IntegratedAssignment.numVar:0:0, IntegratedAssignment.numEqu:0:0, IntegratedAssignment.numNZ:0:0 /;

put card(i):0:0, d:0:0, sum(i$(ord(i)<=d), v(i)):0:0 /;

loop(p $ (ord(p) > n+1), put s.l(p):0:0 /);

loop(i $ (ord(i)<=d), loop(j $ x.l(i,j), put ord(i):0:0, ord(j):4:0, utility(i,j):0:2 /;)

);

loop(p $ (ord(p)<=n+1), loop(q $ y.l(p,q), put (ord(p)-1):0:0, (ord(q)-1):4:0,

s.l(p):0:0, t(p,q):0:0, s.l(q):0:0, v(p):0:0, l.l(p):0:0, v(q):0:0 /;)

);
```