

**YAŞAR UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

MASTER THESIS

**AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH
ALGORITHM FOR THE HETEROGENEOUS PICK-UP
AND DELIVERY VEHICLE ROUTING PROBLEM
WITH TIME WINDOWS**

Gökberk Özsakallı

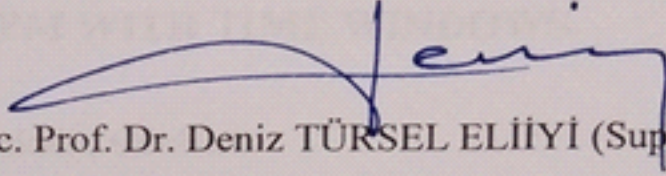
Thesis Advisor: Assoc. Prof. Dr. Deniz TÜRSEL ELİİYİ

Department of Industrial Engineering

Presentation Date: 10.06.2016

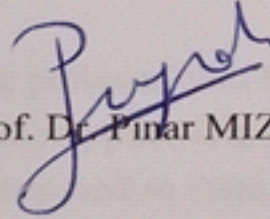
**Bornova-İZMİR
2016**

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.



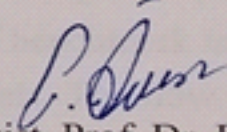
Assoc. Prof. Dr. Deniz TÜRKSEL ELİİYİ (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

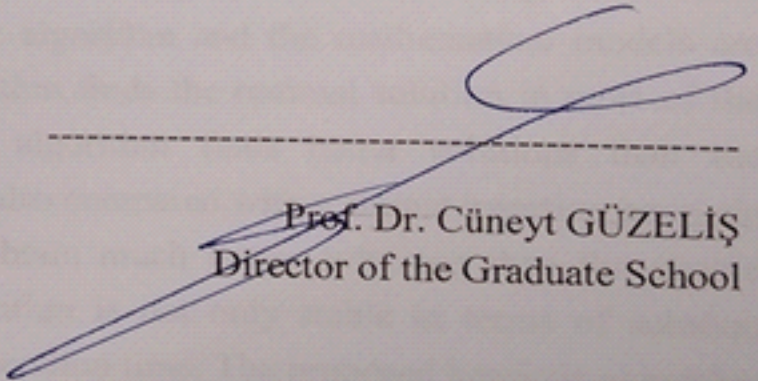


Assoc. Prof. Dr. Pınar MIZRAK ÖZFIRAT

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.



Assist. Prof. Dr. Erdinç ÖNER



Prof. Dr. Cüneyt GÜZELİŞ
Director of the Graduate School

ABSTRACT

AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH ALGORITHM FOR THE HETEROGENEOUS PICK-UP AND DELIVERY VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

Özsakallı, Gökberk

MSc in Industrial Engineering

Supervisor: Assoc. Prof. Dr. Deniz TÜRSEL ELİİYİ

June 2016, 69 pages

In this thesis, a heterogeneous vehicle routing problem with time windows and simultaneous pick-up and delivery, which has wide application areas, is handled. Three different types of mathematical models are proposed to formulate the problem. The first one is based on Miller-Tucker-Zemlin (1960) constraints. The other two are based on flow decision variables. To the best of our knowledge, the problem has not been studied in the vehicle routing literature. A new set of benchmark instances is also generated to compare lower bounds of mathematical models. The flow variable-based mathematical models provide the best results based on the computational experiments. As the mathematical models can solve only small sized instances, a heuristic algorithm based on Adaptive Large Neighborhood Search is proposed to solve larger real world instances. When the proposed heuristic algorithm and the mathematical models are compared, it is observed that the algorithm finds the optimal solution in most of the test instances. On the average, the algorithm finds better solutions than the mathematical models. The algorithm is also compared with a simple insertion heuristic for large instances, and is found to obtain much better solutions than the simple insertion heuristic. The proposed algorithm is not only stable in terms of solution quality, but also robust in terms of computation time. The proposed heuristic algorithm can be used in everyday logistics operations to obtain very fast and high quality solutions.

Keywords: Vehicle routing problem, heterogeneous vehicle fleet, pick-up and delivery, time windows, mixed-integer programming, heuristic algorithm.

ÖZET

HETEROJEN FİLOLU DAĞITIM, TOPLAMA VE ZAMAN PENCERELİ ARAÇ ROTALAMA PROBLEMİ İÇİN ADAPTİF GENİŞ KOMŞULUK ARAMA ALGORİTMASI

Gökberk Özsakallı

Yüksek Lisans Tezi, Endüstri Mühendisliği Bölümü

Tez Danışmanı: Doç. Dr. Deniz TÜRSEL ELİİYİ

Haziran 2016, 69 sayfa

Bu tezde heterojen araç filolu, eşzamanlı dağıtım ve toplamalı, ve müşterilerin mal kabul saatlerinde zaman pencereleri bulunan bir araç rotalama problemi ele alınmıştır. Ele alınan problem gerçek hayatta birçok uygulama alanına sahiptir. Problemi formüle etmek için üç farklı matematiksel model önerilmiştir. Bunlardan ilki Miller-Tucker-Zemlin (1960) kısıtları kullanılarak yazılmıştır. Diğer ikisi ise akış karar değişkenlerinden faydalanılarak yazılmıştır. Bilgimiz dahilinde, tezde ele alınan problem araç rotalama literatüründe henüz çalışılmamıştır. Tezde ayrıca matematiksel modelleri sağladıkları alt sınırlar üzerinden karşılaştırabilmek için yeni test problemleri oluşturulmuştur. Yapılan geniş çaplı sayısal deneyler, en iyi formülasyonun akış değişkenlerinin kullanıldığı formülasyon olduğunu göstermiştir. Matematiksel model ile ancak küçük test problemleri çözülebildiğinden, daha büyük boyutlu gerçek hayat problemlerini çözebilmek amacıyla Adaptif Geniş Komşuluk Arama bazlı bir sezgisel algoritma geliştirilmiştir. Geliştirilen algoritma matematiksel modeller sonucu bulunan çözümler ile karşılaştırıldığında algoritmanın birçok test probleminde optimum sonuç bulduğu ve ortalamada matematiksel model çözümlerinden daha iyi çözümler bulduğu görülmüştür. Büyük veriler üzerinde algoritmayı basit ekleme sezgiseli ile karşılaştırdığımızda ise algoritmanın ekleme sezgiselinden çok daha iyi sonuçlar verdiği, hem çözüm kalitesi açısından oldukça istikrarlı olduğu hem de çözüm süresinin problem boyutuyla çok fazla değişmediği görülmüştür. Önerilen algoritma sevkiyat planlaması yapan firmaların günlük planları için çok hızlı ve yüksek kaliteli sonuçlar üretmede kullanılabilir.

Anahtar sözcükler: Araç rotalama problemi, heterojen araç filosu, dağıtım ve toplama, zaman penceresi, karma tamsayı programlama, sezgisel algoritma.

ACKNOWLEDGEMENTS

First and foremost, I thank God for allowing and enabling me to complete my studies. I would like to thank my advisor, Dr. Deniz Trsel Eliiyi for her guidance, support and valuable comments throughout my Master thesis. It was a privilege to work with her.

I thank Dr. Pınar zfirat, Dr. Erdiñ ner and Dr. Uđur Eliiyi for their willingness to serve on my committee. Also, I would like to thank UNIVERA A.Ş., especially to Seđkin Karabacakođlu for bringing this problem to us. It was a pleasure to work for a real world problem. I give special thanks to Dr. Deniz zdemir; without her I would have never started to graduate education.

Last but not least, I am greatly thankful to my family for their unconditional love and constant support.

Gkberk ZSAKALLI
İzmir, 2016

TEXT OF OATH

I declare and honestly confirm that my study, titled “An Adaptive Large Neighborhood Search Algorithm for the Heterogeneous Pick-up and Delivery Vehicle Routing Problem with Time Windows” and presented as a Master’s Thesis, has been written without applying to any assistance inconsistent with scientific ethics and traditions, that all sources from which I have benefited are listed in the bibliography, and that I have benefited from these sources by means of making references.



TABLE OF CONTENTS

	Page
ABSTRACT	iii
ÖZET	iv
ACKNOWLEDGEMENTS	v
TEXT OF OATH	vi
TABLE OF CONTENTS	vii
INDEX OF FIGURES	x
INDEX OF TABLES	xi
INDEX OF SYMBOLS AND ABBREVIATIONS	xii
1 INTRODUCTION	1
1.1 Subject of the Thesis	1
1.2 Aims and Problem Definition	2
1.3 Context of the Thesis	3
1.4 Organization of the Thesis	4
2 LITERATURE REVIEW	5
2.1 Vehicle Routing Problem	5
2.2 Heuristic Approaches	7

2.3	Discussion	9
3	PROBLEM FRAMEWORK AND MATHEMATICAL FORMULATIONS	11
3.1	Formulation 1: Basic Formulation	12
3.2	Formulation 2: Demand Flow Formulation	14
3.3	Formulation 3: Time Flow Formulation	15
3.4	Valid Inequality	16
4	SOLUTION APPROACH	17
4.1	Minimum Number of Vehicles Heuristic	18
4.2	Initial Solution	20
4.3	Removal Heuristics	21
4.3.1	Random Removal	21
4.3.2	Worst Removal	21
4.3.3	Shaw Removal	22
4.4	Insertion Heuristics	24
4.4.1	Greedy Insertion	24
4.4.2	Regret Insertion	25
4.4.3	Time Windows Insertion	26
4.5	Local Search Heuristics	28

4.5.1	Swap-Based Local Search Heuristic	28
4.5.2	Arc Transfer Local Search Heuristic	29
4.6	Feasibility Check	30
4.7	Acceptance and Stopping Criteria	30
4.8	Adaptive Weight Adjustment	31
5	COMPUTATIONAL STUDY	33
5.1	Mathematical Model Results	33
5.2	Heuristic Results	48
5.3	Discussion	61
6	CONCLUSION	63
	REFERENCES	65
	CURRICULUM VITAE	69

INDEX OF FIGURES

Figure 1: Flow Chart of the Heuristic.

18



INDEX OF TABLES

Table 1: 10 Customer Requests with Low Variance	35
Table 2: 10 Customer Requests with High Variance.....	37
Table 3: 15 Customer Requests with Low Variance	40
Table 4: 15 Customer Requests with High Variance.....	42
Table 5: 20 Customer Requests with Low Variance	44
Table 6: 20 Customer Requests with High Variance.....	46
Table 7: Parameters of ALNS.....	48
Table 8: Comparison of F2 and ALNS.....	50
Table 9: Comparison of F2 and ALNS.....	52
Table 10: Results of ALNS on 50 Requests with Low Variance	55
Table 11: Results of ALNS on 50 Requests with High Variance	56
Table 12: Results of ALNS on 100 Requests with Low Variance	58
Table 13: Results of ALNS on 100 Requests with High Variance	60
Table 14: Comparison of Results	62

INDEX OF SYMBOLS AND ABBREVIATIONS

<u>Abbreviations</u>	<u>Explanations</u>
VRP	Vehicle routing problem
CVRP	Capacitated vehicle routing problem
HVRP	Heterogeneous vehicle routing problem
VRPTW	Vehicle routing problem with time windows
SPDVRP	Simultaneous pick-up and delivery vehicle routing problem
PDPVRP	Pick-up and delivery vehicle routing problem
LP	Linear programming
TSP	Travelling salesman problem
PDPVRPTW	Pick-up and delivery vehicle routing problem with time windows
ALNS	Adaptive large neighborhood search
LNS	Large neighborhood search
HVRPTWSPD	Heterogeneous vehicle routing problem with time windows and simultaneous pick-up and delivery
HTW	Hard time windows
PD	Pick-up and delivery
SP'	No split delivery

ST	Single trip
MSTC	Minimizing total distance cost
HF	Heterogeneous fleet
FC	Fixed capacity
SC	Single compartment
MCP	Multiple compatible product
STP	Single time period
A-BFD	Adaptation of Best-Fit Decreasing

1 INTRODUCTION

One of the main processes in companies is logistics management. It consists of supply of raw materials, transportation of products, stock control etc., among which the most costly process is the transportation of products. Considerable expenses are made for this purpose. Due to this fact, it is obvious that even small improvements in transportation can lead to remarkable overall gains. Therefore, operations research techniques are crucial for the management of logistics operations.

1.1 Subject of the Thesis

The subject of the thesis is developing a problem framework and solution methodologies to a heterogeneous vehicle routing problem with time windows and simultaneous pickup and delivery. The problem handled in this thesis is one of the most comprehensive variants of vehicle routing.

Different variants of the vehicle routing problem were examined and some of the most similar researches were reviewed. Details of the problem such as parameters and constraints were identified. Three different mathematical models were proposed to learn which way of modelling is suitable to the particular problem on hand in terms of obtained lower bounds.

It is known that vehicle routing problem is NP-Hard. Our problem is also NP-Hard, since it generalizes the classical vehicle routing problem. Therefore, to solve real-world instances which are larger than classical benchmark instances, a heuristic approach is also proposed in this thesis. To the best of our knowledge, the problem in this thesis has not been studied in the literature. So, to test proposed mathematical models and proposed heuristic algorithm, a new set of instances was generated. A computational analysis was made, which includes small instances to compare the lower bounds obtained from different formulations, and to compare the results of the proposed heuristic approach to the optimal or best solutions from the developed formulations. Computational analysis also includes larger instances to measure the solution quality of the heuristic, by comparing our heuristic to a simple insertion based heuristic, as there are no available benchmark instances for our problem in the literature.

Most of the route planners confront with this problem almost every day. For this reason, a software company wanted us to develop a solution methodology to provide fast and good results for its customers. Therefore, this thesis is motivated from a concrete real life need.

1.2 Aims and Problem Definition

The aim of this thesis is to study a problem that has not been studied extensively. The problem has broad application areas from transporting goods to transporting people. The wide application area is critical for us, to be able to provide benefits to practitioners in different sectors, as the study is motivated from a real-life problem. As the total logistics costs are estimated as twenty percent of total production costs in OECD countries (Sudalaimuthu and Raj, 2009), another important aim is to develop an algorithm to obtain good solutions to real world instances, in order to help practitioners in terms of planning time and total logistics costs.

The problem interests any company distributing its products from the depot to some customers. The decisions of which product should be assigned to which vehicle, and determining the routes of the vehicles is of concern. In the problem studied in this thesis, the company may have a heterogeneous fleet where the vehicles may differ from one to another in terms of capacities and costs. We handle the problem from the company's point of view, considering customer requirements. For example, some customers may request specific time intervals to accept visits from the vehicles, or some customers may have not only delivery demands but also some pickup demands to be transported back to the depot. Therefore, in this study, three extensions of the basic vehicle routing problem are considered: heterogeneous vehicles, time windows and simultaneous pickup and delivery demands.

Therefore, the heterogeneous vehicle routing problem with time windows and simultaneous pick-up and delivery is considered in this thesis, where the objective is to minimize the total fixed cost of the used vehicles and the total travelling cost. To the best of our knowledge, this complex problem has not been studied yet in the vehicle routing literature. Three different types of mathematical models is proposed to formulate the problem. The first one is based on Miller-Tucker-Zemlin (1960) constraints. The other two are based on flow decision variables. A new set of benchmark instances is generated to compare the lower bounds obtained by the

mathematical models. The flow variables-based mathematical models give the best results based on computational experiments. As the mathematical models can solve only small sized problem instances to optimality, a heuristic algorithm is also proposed to solve larger real world instances. When the proposed heuristic and the models are compared, the algorithm finds optimal solution in most of the instances, and on the average the algorithm finds better solutions than the mathematical models. For larger instances, we compare the algorithm with a simple insertion heuristic to find that it finds much better solutions than the simple heuristic. The results of the experiments show that the proposed algorithm is not only stable in terms of solution quality but also robust in terms of computational time. As a result, the proposed heuristic algorithm can be securely used in everyday logistics operations to obtain fast and quality results. The proposed algorithm will be embedded in a software package, and will be provided as a logistics planning tool to the customers of the software company.

1.3 Context of the Thesis

The thesis content is based on the routing plan of vehicles in companies, which distribute their products to delivery locations. In the first part, we define the problem framework, which contains parameters and constraints. As the problem is motivated from a real-life need, the vehicle fleet is assumed to be heterogeneous, as most distribution companies have different types of vehicles. Time windows of the customers are respected while minimizing the total transportation cost. The time window of a customer is defined as the time interval in which the customer should receive the delivery by a vehicle. We consider hard time windows, which are identified as strictly determined deadlines. The pickup demands of customers are also included, as these can be seen as returned products, which have a wide application area in reverse logistics.

In the second part of the thesis, we review the literature on similar studies to our problem. Mathematical models of some specific vehicle routing problems are analyzed, and solution approaches are examined. As exact approaches can solve very limited number of instances, we concentrate on heuristic algorithms. In the last part, we propose three different formulations of the problem to examine which way of modeling performs better for obtaining lower bounds. To compare the lower bounds, we propose new sets of instances that consider different types of vehicles, pickup

demands and time windows, by extending Solomon's (1987) benchmark instances. We also propose a heuristic algorithm, inspired from the Adaptive Large Neighborhood Search by Ropke and Pisinger (2006), to solve large practical instances.

1.4 Organization of the Thesis

This thesis consists of six chapters and is organized as follows. The next chapter provides the literature review on related vehicle routing problems in three sections. The first section gives some background information about the vehicle routing problem and its variants. The second section consists of heuristic approaches to solve different variants of the problem. The last section concludes Chapter 2 with a discussion. Chapter 3 provides detailed information about the considered problem; three different mathematical models are presented, and a valid inequality proposed by Yaman (2006) is adapted in this chapter. In Chapter 4, a heuristic algorithm is proposed to obtain approximate solutions to the problem. The algorithm contains several sub-heuristics, and the pseudocode is provided for each. Chapter 5 reports the experimental analysis on the solutions of the mathematical models and the heuristic algorithm on generated test instances. Finally, the conclusions are given and the contributions of the thesis and future directions are discussed in Chapter 6.

2 LITERATURE REVIEW

As mentioned in the previous chapter, we consider a single depot, heterogeneous fleet, pick-up and delivery vehicle routing problem with time windows. Early works on vehicle routing problem (VRP) is started with Dantzig and Ramsey (1959). Since that date, VRP has been increasingly getting attention by the operations research community. This chapter is divided into three sections. The next section is a review of different class of VRPs, and the solution approaches are reviewed in the Chapter 2.2.

2.1 Vehicle Routing Problem

The three simpler variants of the problem considered in this study are introduced in this section. These are heterogeneous VRP (HVRP), VRP with time windows (VRPTW), and simultaneous pick-up and delivery VRP (SPDVRP). In the following section, different variants of the VRP are explained and some studies are reviewed. Mathematical definitions are excluded for conciseness. For more detailed information on the problem, we refer the reader to the excellent textbook by Toth and Vigo (2001), and to the survey by Laporte (1992). It should be noted that, since the classical VRP is NP-hard because it generalizes the Travelling Salesman Problem, all three variants are NP-hard, as well.

Heterogeneous Fleet VRP

In the capacitated VRP (CVRP), it is assumed that all vehicles are identical in terms of capacities and costs. The HVRP generalizes the CVRP by considering different types of vehicles. In general, the objective function is minimizing the fixed costs and the routing costs of vehicles. Although it is more realistic and the real-world problems deal with heterogeneous vehicles more than the identical vehicles, the HVRP has received much less attention by the researchers, possibly due to its complexity.

Yaman (2006) also notes that the lack of interest may be caused by the difficulty of finding good lower bounds for HVRP. The reason of the difficulty is that, the fixed costs dominate the routing costs in the optimal solution of the linear

programming (LP)-relaxation. Therefore, the vehicles make fractional subtours. Also, Yaman (2006) proposes six different formulations for the HVRP; four of them are based on Miller-Tucker-Zemlin constraints and two are based on flow formulations. Flow formulations lead to much better lower bounds than Miller-Tucker-Zemlin based formulations. However, the computational time of strong formulations increases significantly for large instances. To improve lower bounds, several valid inequalities covering type inequalities, subtour elimination inequalities, generalized large multistar inequalities and valid inequalities based on lifting of the Miller-Tucker-Zemlin constraints are proposed by the author. Computational results show that the valid inequalities, especially covering type inequalities, improve the lower bounds significantly.

The PhD dissertation of Özfirat (2008) considers three different variants of VRP, namely HVRP, split delivery (SDVRP) and VRPTW. A threshold algorithm is proposed to solve HVRP, SDVRP and small-sized VRPTW. The developed threshold algorithm is a two-stage algorithm, where the customers are clustered in the first stage and the routes are determined for each cluster in the second stage. A fuzzy goal programming approach is proposed for the clustering stage, and a constraint programming model is developed for the routing stage. In addition, a set covering algorithm is proposed for large scale VRPTW instances.

Baldacci et al. (2008) reviews different formulations and valid inequalities that are proposed by other researchers. Also, the authors briefly describe heuristic approaches proposed in the literature, and compare computational performances.

VRP with Time Windows

The VRPTW is the extension of the CVRP where each customer has a time interval and a vehicle must visit the customer in that interval. There are two types of time windows, defined as soft and hard. Soft time windows allow the vehicles to visit customers outside their time windows with a penalty cost. On the contrary, the hard time windows cannot be violated. For detailed information, we refer the reader to the surveys by Cordeau et al. (2001), Kallehauge et al. (2005) and Kallehauge (2008). Cordeau et al. (2001) presents a multi-commodity network flow model to formulate VRPTW, and describes different heuristic approaches to derive upper bounds, and

two decomposition approaches (Lagrangian relaxation and column generation) for deriving lower bounds. Kallehauge et al. (2005) focuses on column generation approach in general, and path formulation is reviewed extensively. Also, the authors propose several acceleration strategies improving the overall approach considerably. Kallehauge (2008) reviews four types of formulations of Travelling Salesman Problem (TSP), which are the arc formulation, the arc-node formulation, the spanning tree formulation and the path formulation, and gives polyhedral results. The reason for reviewing different formulations of TSP is that, if one applies a decomposition approach, the subproblem can be formulated as TSP. Thus, the TSP forms a basis for the VRPTW. The author also gives a detailed literature review of solution approaches for the path formulation.

It should be noted that the existing solution approaches for the VRPTW are dominated by column generation. One of the most important reasons for this dominance might be the advancements in the solution of the subproblem, which is a path formulation, as there are pseudo-polynomial algorithms for the Shortest Path Problem with Resource Constraints.

Pick-up and Delivery VRP

Another important class of VRP is the pickup and delivery problems (PDVRP). In this problem category, people or goods are to be transported to some destination points. In general, each request is defined by a pickup location and a delivery location. Delivery location may be the same as the pickup location. If that is the case, the problem is called as SPDVRP. Otherwise, the problem has some extra constraints such as coupling and precedence. In the VRP literature, the SPD has not received much attention as other pickup and delivery problems. For more information on PDVRP, we refer the reader to surveys by Berbeglia et al. (2007) and Parragh et al. (2008).

2.2 Heuristic Approaches

In last two decades, several unsolved VRP problem instances have been solved optimally. However, for practical instances, exact algorithms are not reliable for solving the problem in terms of variability of computational time. From a practical point of view, the computational time is as important as the quality of the solution.

Therefore, heuristic approaches are still the only viable option for large instances. Several surveys on heuristic approaches have been published. We refer the reader to these surveys by Laporte et al. (2000), Cordeau et al. (2005), and Vidal et al. (2013) for detailed information.

Dethloff (2001) proposes a simple insertion-based heuristic for simultaneous pick-up and delivery VRP. The insertion heuristic includes the function of the total length of vehicle, the function of the remaining capacity of vehicle, and a distance function which prevents the late and unfavorable insertions of distant located customers. The algorithm starts with selecting a seed customer to create a route. The insertion cost is calculated for all unassigned customers (the customers who has not assigned to any vehicle/route) for each possible insertion to the route. Then the best candidate in terms of minimum cost is assigned to the route. This step is continued until no unassigned customer can be inserted into the route. After that step, if there are some unassigned customers, a new route is created with the same steps. The computational complexity of this very fast algorithm is $O(n^4)$. However, the algorithm assumes homogenous vehicles and does not consider the time window constraints.

Li and Lim (2003) use a metaheuristic approach to solve the pick-up and delivery VRP with time windows (PDPVRPTW). The proposed algorithm is a hybrid method of simulated annealing and tabu search. The algorithm starts with finding an initial solution by using a simple insertion method. Then to find better feasible solutions, three types of neighborhood structures are used within a simulated-annealing-like multiple-restart strategy. The algorithm also contains a tabu list to avoid cycling.

Lu and Dessouky (2006) develop an insertion-based heuristic for solving the PDVRPTW. The insertion heuristic considers the total length of the route as well as the total slack time of the time windows. In addition, a nonstandard measure “crossing length percentage” is presented to determine the visual attractiveness of the solution. Its value takes zero if the route does not have any crossing points in itself, and the value increases with respect to the length of the crossing distance. The routes with high crossing length percentage values are not accepted. The algorithm can be summarized as follows: Initial routes are determined by solving a maximum clique problem with a greedy algorithm. The solution gives a set of customers which cannot

be assigned to the same route. Then the insertion cost is calculated for each unassigned customer for each possible insertion to the routes. This process is continued until there is no customer to be assigned. The computational complexity of the algorithm is $O(n^4)$. However, this algorithm also assumes the vehicles are identical.

Ropke and Pisinger (2006), and Pisinger and Ropke (2007) mention that there are many heuristic methods in the literature, but these methods have highly strict structures for specific VRP variants. These restrictions prevent the algorithms from applying to different problem types. Thus, the authors propose an algorithm that can be applied to five different VRP variants: VRPTW, CVRP, multi depot VRP, open VRP, and site-dependent VRP. First, the problems are transformed into PDVTPTW. Then, Adaptive Large Neighborhood Search (ALNS) is employed, which is an extension of the large scale neighborhood search heuristic of Shaw (1997, 1998). ALNS includes several insertion and removal heuristics, and applies these heuristics with an adaptive layer. In each iteration, a removal heuristic is selected to remove a predetermined number of customer requests from routes, and an insertion heuristic is selected to insert the unassigned customer requests to the routes. The selection of heuristics is adaptively made by considering the contribution in previous iterations. More information on Large Neighborhood Search (LNS) can be found in a survey proposed by Pisinger and Ropke (2010). The computational results show that the proposed algorithm is very stable in terms of the solution quality. The only disadvantage of the algorithm is, as ALNS is a metaheuristic, there are several parameters to fine-tuned. Due to the effectiveness of ALNS, the number of papers employing ALNS has been increasing in recent years (Riberio and Laporte, (2012), Hemmelmayr et al., (2012), Demir et al., (2012), Azi et al., (2014) Luo et al., 2016)).

2.3 Discussion

Vehicle routing has been getting attention from the operations researchers for a long time. This interest leads researchers to study different variants of the problem. Also, as the different variants have been studied, a lot of effective algorithms have been proposed. However, the mainstream of the research focused on solving some benchmark problem instances better than others. In our opinion, it will be better to study practical realistic variants of the problem by a utilitarian approach. In other words, studying variants of the problem that have not received much attention despite

their broad application areas might be worthwhile from a practical point of view. For this purpose, we study the heterogeneous vehicle routing problem with time windows and simultaneous pickup and delivery (HVRPTWSPD). To the best of our knowledge, this problem has not been studied in the literature.

A recent study (Taşar, 2016) proposes a taxonomy for the VRP, which involves a five-field notation including operational policy, objective function, vehicle features, product features and the planning period. While operational policy field includes time windows, carrying types, split deliveries and trips, the objective function field contains minisum, minimax and load balancing type objectives. Vehicle features involve fleet type, capacity and compartment and the product features represent different product types in the problem. Finally, the planning period field contains the number of periods. Based on this notation, our operational policies include hard time windows (HTW), pick-up and delivery (PD), no split delivery (SP') and single trip (ST). Our objective function is minimizing the total distribution cost (MSTC). The vehicle features are heterogeneous fleet (HF), fixed capacity (FC) and single compartment (SC), while the product features include multiple compatible products (MCP). Our planning period is a single time period (STP). Hence, our problem can be represented as follows, using the taxonomy by Tasar (2016):

HTW, PD, SP', ST | MSTC | HF, FC, SC | MCP | STP.

In the next chapter, we examine the details of the problem and propose three different formulations.

3 PROBLEM FRAMEWORK AND MATHEMATICAL FORMULATIONS

In this study, three extensions of the basic vehicle routing problem are considered together, which are VRPTW, HVRP and SPDVRP. It should be noted that this practical problem has been brought to our attention by a software company, which needed to form a route optimization algorithm as a logistics planning tool for its customers.

At the beginning of a planning day, all vehicles are ready to start their routes at the depot. When a vehicle is loaded with a set of customer delivery demands, it starts its route at time 0. Next, the vehicle visits its first customer to satisfy their delivery demand. If the customer has a pick-up demand as well, the pick-up demand must be picked up by the same vehicle at the same time. Hence, we assume that each customer is visited exactly once. For representation purposes, it is assumed that depot has zero pick-up and delivery demand. The arrival time of the vehicle to the first customer is the departure time from the depot plus the travel time between the depot and the first customer. If there is any service time at the depot, that must also be included in the arrival time to the first customer. The vehicle visits all its assigned customers in its route in the same manner, before returning to the depot. The vehicles may differ from each other in terms of their capacities, fixed costs, and cost per kilometer. We assume that the customers and the depot have time windows for receiving their service by a vehicle, i.e. a vehicle arriving to a customer before the start of their time window should wait until the start. Similarly, it is not allowed to serve a customer after the end of the customer's time window. Latest arrival time to the depot indicates the permissible working hours of the vehicles. The objective function is to minimize the total fixed cost of vehicles, as well as total travelling costs.

The problem can be formulated as follows: $G = (V, A)$ is a complete directed graph with node set $V = \{0, 1, \dots, n, n + 1\}$ and arc set $A = \{(i, j) : i, j \in V, i \neq j\}$. Nodes 0 and $n + 1$ denote the initial and the terminal depot, which may correspond to the same geographical location. Subset $V_c = \{1, 2, \dots, n\}$ denotes customer nodes. All vehicles start their routes at node 0 and finish at node $n + 1$. The set $K = \{1, \dots, m\}$ denotes the vehicles, while m represents the number of available vehicles.

In this thesis, we propose three different formulations to the problem. The first formulation is based on Miller-Tucker-Zemlin (1960) constraints, and the last two are based on flow variables. The following parameters and decision variables are common to the three formulations.

Common Parameters:

d_i : delivery demand of node $i \in V$.

p_i : pick-up demand of node $i \in V$.

$t_{i,j}$: distance between node i and node j , $(i,j) \in A$.

h_i : service time of node $i \in V$.

$[a_i, b_i]$: earliest and latest possible arrival time of node $i \in V$.

c^k : capacity of vehicle $k \in K$.

γ^k : fixed cost of vehicle $k \in K$.

α^k : cost per kilometer of vehicle $k \in K$.

β^k : average speed of vehicle $k \in K$.

M : A large number, e.g. $M = \max\{\sum_{i \in V} d_i + p_i, \sum_{(i,j) \in A} t_{i,j}\}$.

Common Decision Variables:

$x_{i,j}^k$: 1, if vehicle $k \in K$ travels directly from node $i \in V$ to node $j \in V$. 0, otherwise.

w^k : 1, if vehicle $k \in K$ is used. 0, otherwise.

3.1 Formulation 1: Basic Formulation

The first formulation is based on Miller-Tucker-Zemlin (1960) constraints.

Additional Decision Variables:

s_i^k : the time vehicle $k \in K$ arrives at node $i \in V$.

l_i^k : the load of vehicle $k \in K$ after leaving node $i \in V$.

With the above definitions, the mathematical model becomes:

$$\min: \sum_{(i,j) \in A} \sum_{k \in K} \alpha^k t_{i,j} x_{i,j}^k + \sum_{k \in K} \gamma^k w^k \quad (0)$$

subject to:

$$\sum_{i \in V} \sum_{k \in K} x_{i,j}^k = 1 \quad j \in V_c \quad (1)$$

$$\sum_{i \in V} x_{i,j}^k - \sum_{i \in V} x_{j,i}^k = 0 \quad j \in V_c, k \in K \quad (2)$$

$$s_j^k \geq s_i^k + \frac{t_{i,j}}{\beta^k} + h_i - M(1 - x_{i,j}^k) \quad (i,j) \in A, k \in K \quad (3)$$

$$a_i \leq s_i^k \leq b_i \quad i \in V, k \in K \quad (4)$$

$$l_j^k \geq l_i^k + p_i - d_i - M(1 - x_{i,j}^k) \quad (i,j) \in A, k \in K \quad (5)$$

$$l_0^k = \sum_{(i,j) \in A} d_i x_{i,j}^k \quad k \in K \quad (6)$$

$$l_i^k \leq c^k \quad i \in V, k \in K \quad (7)$$

$$\sum_{j \in V_c} x_{0,j}^k = w^k \quad k \in K \quad (8)$$

$$x_{i,j}^k \in \{0, 1\} \quad (i,j) \in A, k \in K \quad (9)$$

$$s_i^k, l_i^k \in \mathbb{R}^+ \quad i \in V, k \in K \quad (10)$$

The objective function minimizes the fixed cost of vehicles and the total travel cost. Constraint set (1) ensures that each customer request is satisfied. Constraint set (2) guarantees that, if a vehicle visits one node it must leave that node. Constraint set (3) imposes consistency of the visiting times of nodes. Constraint set (4) ensures that the arrival times of nodes are within time windows. Constraint set (5) imposes consistency of loads of vehicles. Constraint set (6) determines the load of vehicles upon leaving the depot. Constraint set (7) ensures that the vehicle capacities are not exceeded. Constraint set (8) determines which vehicles are used. Constraint sets (9) and (10) represent the ranges of decision variables.

3.2 Formulation 2: Demand Flow Formulation

The second formulation is based on demand flow variables. Two additional decision variables are used to represent pick-up and delivery demand quantities along arcs.

Additional Decision Variables:

s_i^k : the time which vehicle $k \in K$ arrives at node $i \in V$.

$f_{i,j}^+$: the delivery demand flow of arc $(i,j) \in A$.

$f_{i,j}^-$: the pick-up demand flow of arc $(i,j) \in A$.

With the additional decision variables, the mathematical model becomes:

min: (0)

subject to:

(1) – (4), (8) – (9), and

$$\sum_{i \in V} f_{i,j}^+ - \sum_{i \in V} f_{j,i}^+ = d_j \quad j \in V_c \quad (11)$$

$$\sum_{k \in K} d_j x_{i,j}^k \leq f_{i,j}^+ \leq \sum_{k \in K} (c^k - d_i) x_{i,j}^k \quad (i,j) \in A \quad (12)$$

$$\sum_{j \in V} f_{i,j}^- - \sum_{j \in V} f_{j,i}^- = p_i \quad i \in V_c \quad (13)$$

$$\sum_{k \in K} p_i x_{i,j}^k \leq f_{i,j}^- \leq \sum_{k \in K} (c^k - p_j) x_{i,j}^k \quad (i,j) \in A \quad (14)$$

$$f_{i,j}^+ + f_{i,j}^- \leq \sum_{k \in K} c^k x_{i,j}^k \quad k \in K \quad (15)$$

$$s_i^k, f_{i,j}^+, f_{i,j}^- \in \mathbb{R}^+ \quad (i,j) \in A, k \in K \quad (16)$$

Constraint set (11) guarantees that the amount of delivery flow d_j should be sent to node $j \in V_c$. Constraint set (12) ensures that, if arc $(i,j) \in A$ is traversed by a vehicle, delivery flow on it should be at least the delivery demand of node $j \in V_c$, and the delivery flow on the arc cannot exceed the capacity of the vehicle. Constraint set (13) guarantees that, amount of pick-up flow p_i should be sent to node $i \in V_c$. Constraint set (14) ensures that if arc $(i,j) \in A$ is traversed by a vehicle, the delivery flow on it should be at least the pick-up demand of node $i \in V_c$, and the delivery flow on the arc cannot exceed the capacity of the vehicle. Constraint set (15) ensures that vehicle capacity is not exceeded. Constraint set (16) represents the range of decision variables.

3.3 Formulation 3: Time Flow Formulation

The third formulation is based on time flow variables. An additional decision variable is used to represent time flow along arcs.

Additional Decision Variables:

l_i^k : the load of vehicle $k \in K$ after leaving node $i \in V$.

$s_{i,j}$: the time when the node $i \in V$ is left in direction to node $j \in V$.

With the above additional decision variables, the mathematical model becomes:

min: (0)

subject to:

(1) – (2), (5) – (9), and

$$\sum_{j \in V} s_{i,j} \geq \sum_{j \in V} s_{j,i} + \sum_{j \in V} \sum_{k \in K} x_{j,i}^k \left(\frac{t_{j,i}}{\beta^k} + h_i \right) \quad i \in V \quad (17)$$

$$\sum_{k \in K} x_{i,j}^k (a_i - h_i) \leq s_{i,j} \leq \sum_{k \in K} x_{i,j}^k (b_i - h_i) \quad i \in V \quad (18)$$

$$s_{i,j}, l_i^k \in \mathbb{R}^+ \quad (i,j) \in A, k \in K \quad (19)$$

Constraint set (17) determines the time a vehicle leaves node $i \in V$. Constraint set (18) ensures that the arrival time to the nodes must be within their respective time windows. Constraint set (19) represents the range of decision variables.

3.4 Valid Inequality

As mentioned in the literature review, the LP-relaxation solution of the HVRP is too weak (Yaman, 2006). The other disadvantage of solving the HVRP is, since the capacities and costs of the vehicles are different, the selection of which vehicles are used is crucial. This situation significantly increases the size of branch-and-bound tree in the exact solution. Therefore, the compound effect of these two difficulties makes the problem hardly solvable even for small instances. This fact is verified in our computational experiments, as well. To overcome these disadvantages, Yaman (2006) proposed several valid inequalities to the HVRP. The most promising valid inequality is the covering type inequality, which is also valid for the HVRP with time windows. Therefore, we add this inequality to our formulations for $Q > 0$, as follows:

$$\sum_{j \in V_c} \sum_{k \in K} \left\lceil \frac{c^k}{Q} \right\rceil x_{0,j}^k \geq \sum_{i \in V_c} \left\lceil \frac{d_i}{Q} \right\rceil \quad (20)$$

4 SOLUTION APPROACH

As the VRP is NP-Hard, it is quite difficult to solve real-world instances. Under heterogeneous vehicles assumption, it is harder to solve even small instances optimally. In this thesis, a heuristic approach is proposed to overcome this difficulty. Our heuristic is inspired by the algorithm of Ropke and Pisinger (2006), namely the Adaptive Large Neighborhood Search (ALNS), which was developed for the PDVRPTW. In our case the problem is HVRPTWSPD, which leads to two extensions in the algorithm; computing the necessary number of vehicles, and a new insertion heuristic for time windows. The proposed heuristic consists of two stages. The main objective is minimizing the number of vehicles, and the second aim is minimizing the total travelling distance of the vehicles. The algorithm considers all assumptions of the problem like time windows, pickup demands, and heterogeneous fleet.

The details of the proposed heuristic are explained in the following sections. Also, a flow chart is presented in Figure 1. The underlined words below indicate the heuristics (subroutines) explained in this chapter.

The ALNS Heuristic:

- S1. Calculate the Minimum Number of Vehicles.
- S2. Find an Initial Solution. Set this solution as the current solution and the incumbent solution.
- S3. If the solution is infeasible, increase the number of vehicles and go to S2. Otherwise, go to S4.
- S4. Generate new solution:
 - a. Select a Removal heuristic based on removal heuristic scores, and apply to the current solution.
 - b. Select an Insertion Heuristic based on insertion heuristic scores, and apply to the current solution.
- S5. Apply a Local Search algorithm to the new solution.
- S6. If the Acceptance Criteria is met, set new solution to the current solution.
- S7. Update Heuristic Scores.
- S8. If the total cost of the current solution is less than the incumbent solution, set

current solution to the best solution.

S9. If the Stopping Criteria is met, go to S10. Otherwise, go to S4.

S10. If the last found incumbent solution is feasible, decrease the number of vehicles by one and go to S2. Otherwise, go to S11.

S11. Report the last found best solution.

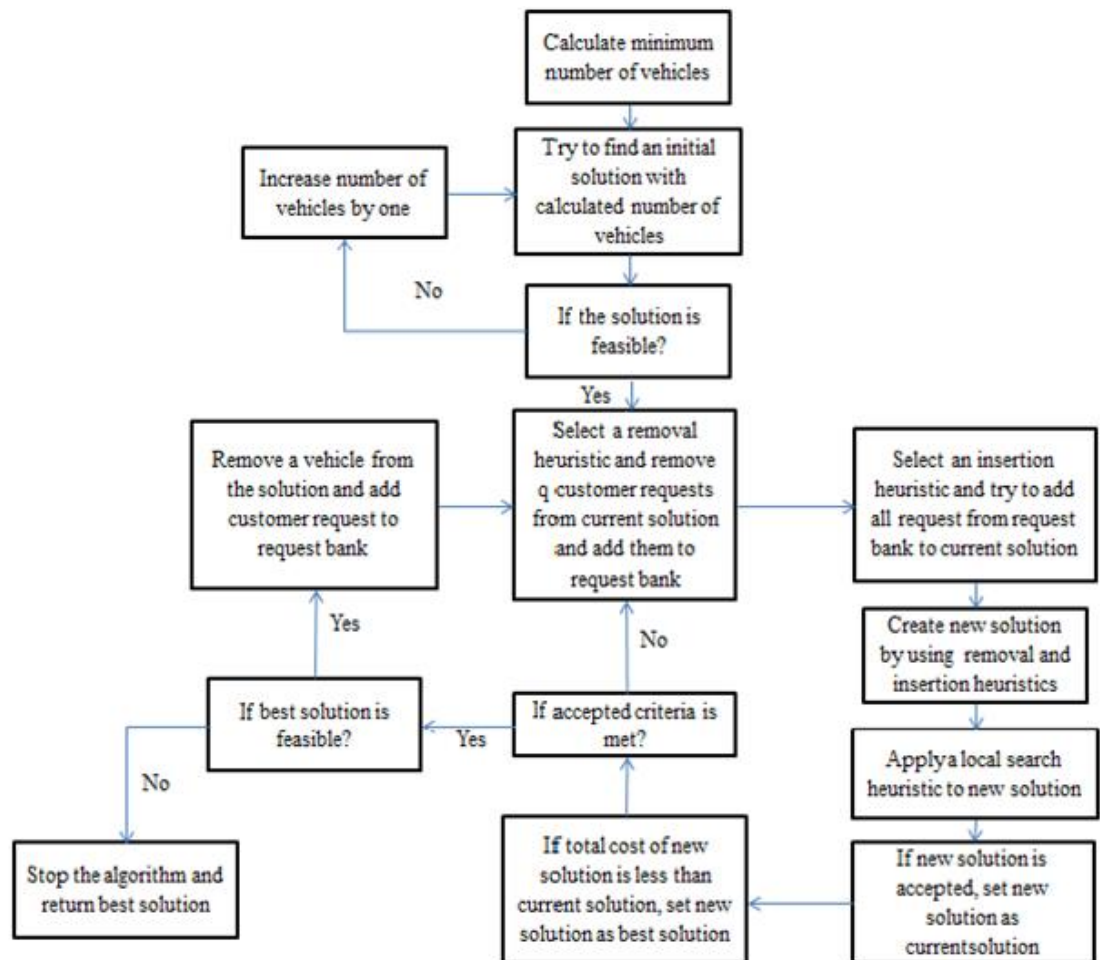


Figure 1: Flow Chart of the Heuristic.

4.1 Minimum Number of Vehicles Heuristic

The heuristic algorithm starts with finding a lower bound on the needed number of vehicles. Unlike our strategy, Ropke and Pisinger (2006) starts by finding an upper

bound on the number of vehicles. The advantage of starting with a lower bound instead of an upper bound is decreasing the total computation time of the algorithm. As the initial solution algorithm is *regret-2*, it finds a solution very quickly, as it will be explained later. This way, the main part of the heuristic can spend more time to improve the initial solution with the least number of vehicles.

Finding a minimum number of vehicles becomes a variant of the bin packing problem. As the vehicle fleet is heterogeneous, the bins are heterogeneous as well. However, it is time consuming to solve this problem to optimality as the bin packing problem is NP-Hard. Also, in our problem, the lower bound on the number of vehicles may be weak due to the existence of the time windows. Therefore, bin packing with heterogeneous bins problem is solved by an adaptation of the Best-Fit Decreasing heuristic (A-BFD), proposed by Crainic et al. (2011). The pseudocode of the algorithm is below.

Inputs:

V_c : Set of customers

K : Set of vehicles

S : Set of selected vehicles ($S = \{\emptyset\}$)

d_i : Delivery demand of customer $i \in V_c$

p_i : Pickup demand of customer $i \in V_c$

v_i : $v_i = \max\{d_i, p_i\}$, $i \in V_c$

γ_k : Fixed cost of vehicle $k \in K$

c_k : Capacity of vehicle $k \in K$

Sort the customer requests in V_c in their nonincreasing order of v_i , $i \in V_c$

Sort the vehicles in K in their nonincreasing order of ratio γ_k/c_k , $k \in K$, and nondecreasing order of c_k when the fixed costs γ_k , $k \in K$ are equal.

Minimum Number of Vehicles (Crainic et al., 2011)

```

FOR All customer requests  $i \in V$ ;
  IF Customer request  $i$  can be assigned to a vehicle in  $S$ 
    Assign the customer request  $i$  to the best vehicle in  $S$  (vehicle with
    maximum idle capacity)
  ELSE
    Add the first vehicle,  $b'$ , in  $K$  to the  $S$ ,  $S = S \cup \{b'\}$ , and assign the
    customer request  $i$  to vehicle  $b'$ 
  ENDIF
ENDIFOR
FOR All vehicles,  $j \in S$ ;
  FOR All vehicles,  $k \in K \setminus S$ ;
     $U_j = \sum_{i \text{ loaded in } j} v_i$ 
    IF  $c_k \geq U_j$  and,  $\gamma_k < \gamma_j$ 
      Move all customer requests from  $j$  to  $k$ 
    ENDIF
  ENDFOR
ENDFOR
Return the vehicles in  $S$  and the total number of vehicles, ( $|S|$ )

```

4.2 Initial Solution

As stated in the previous section, the initial solution is found by *regret-2* heuristic, which is explained in the insertion heuristic section. At the beginning of the algorithm, all customer requests are placed into a request bank, and *regret-2* is run in parallel for all vehicles found by the Minimum Number of Vehicles heuristic. If the initial solution is infeasible, the first unused vehicle in K is added to the selected vehicles set S . This procedure is repeated until a feasible solution is found.

4.3 Removal Heuristics

The removal heuristics remove a predetermined number of customer requests from the current solution, and add them to the request bank. The request bank contains a set of customer requests not assigned to any vehicle. We propose three types of removal heuristics, as *random*, *worst*, and *Shaw Removal*.

4.3.1 Random Removal

This is a simple removal heuristic that selects r customer requests at random, and removes them from the current solution, as can be seen in the pseudocode below. The idea of random removal is applying randomization to the search process.

Inputs:

n : Total number of customer requests

r : Total number of removed customer requests

X : Set of customer requests in the current solution

D : Request bank

Random Removal
WHILE r customer request is selected; Select a customer requests at random from X and remove the selected demand from the current solution, and assign it into the request bank, D ENDWHILE Return the request bank and new solution

4.3.2 Worst Removal

This subroutine selects r customer requests from the current solution that are most costly, and removes them from the solution. To achieve this, the costs of all customer requests are determined by removing the demand from the solution and

calculating cost of the demand. As the demands placed in the request bank have high cost, they are not assigned to the request bank but removed temporarily from the solution. Then, the customer request with the highest cost is removed. The details of the worst removal heuristic are provided in the following pseudocode.

Inputs:

x : Current solution

D : Request bank

$f(x)$: Cost of the current solution

f_{-i} : Cost of current solution after customer request i is removed

Δf_i : Cost of customer request i

$$\Delta f_i = f(x) - f_{-i} \quad (21)$$

Worst Removal
<p>WHILE r customer request is selected;</p> <p style="padding-left: 40px;">FOR All customer requests in solution, $i \in x$;</p> <p style="padding-left: 80px;">Calculate the cost (21) of customer i</p> <p style="padding-left: 40px;">ENDFOR</p> <p style="padding-left: 40px;">Select the customer with the highest cost and remove it from the solution x, and add to the request bank, D</p> <p>ENDWHILE</p> <p>Return the request bank and new solution</p>

4.3.3 Shaw Removal

This heuristic was proposed by Shaw (1997, 1998). The aim is removing customer requests from the current solution that are similar in terms of their delivery and pickup demands and locations. The idea is to reassign such requests to the same

vehicle, if possible. The heuristic starts by selecting and removing a customer request at random from the current solution, and assigning this customer to the removed list. Then, it randomly selects a customer request from the removed list and calculates the similarity of the selected customer and the rest of the customers by using a relatedness measure. The most similar customer request is removed from the solution, and added to the removed list. These steps are repeated until r customer requests are removed from the current solution. The pseudocode of the heuristic is below.

Inputs:

V_c : Set of customers

B : Removed list

D : Request bank

x : Current solution

$t_{i,j}$: Distance between customer $i \in V_c$ and customer $j \in V_c$

d_i : Delivery demand of customer $i \in V_c$

p_i : Pickup demand of customer $i \in V_c$

$R_{i,j}$: Relatedness measure of customer i and j

$$R_{i,j} = \alpha * t_{i,j} + \beta * (|d_i - d_j| + |p_i - p_j|) \quad (22)$$

Shaw Removal (Shaw, 1997)

Remove a customer request from solution $i \in x$ at random, and assign it to the removed list, B .

Select a customer at random from removed list, $i \in B$

WHILE $r - 1$ customer request is removed;

 FOR All customer requests from current solution, $j \in x$;

```
        Calculate  $R_{i,j}$  (22)
    ENDFOR
    Remove the customer from the solution with the lowest  $R_{i,j}$  value, and assign it
    to removed list  $j \in B$ 
ENDWHILE
Move all customer requests from  $B$  to  $D$  and return new solution
```

4.4 Insertion Heuristics

There are two main types of insertion heuristics in the VRP literature, which are sequential insertion and parallel insertion. Sequential insertion methods construct routes one by one whereas parallel insertion constructs several routes concurrently. Our proposed algorithm contains three parallel insertion heuristics, as *Greedy*, *Regret*, and *Time Windows Insertion*.

4.4.1 Greedy Insertion

In this basic insertion heuristic, all customer requests from the request bank are tried for all possible positions in vehicles, to calculate an insertion cost for each position. If a request cannot be inserted to a position, the cost is set to infinity. Then, the customer request with the least insertion cost is assigned to its determined position and vehicle. These steps are repeated until the request bank is empty or no insertion is possible. Since only one route is changed at each iteration, the insertion costs for the other vehicles do not need to be calculated. This implementation improves the computation time of all insertion heuristics. The pseudocode is below.

Inputs:

D : Request bank

S : Set of vehicles used in new solution

$t_{i,j}$: Distance between customer $i \in V_c$ and customer $j \in V_c$

$\Delta_{i,k,j}^l$: change in objective value by inserting customer k between customers i and j in route l

$$\Delta_{i,k,j}^l = t_{i,k} + t_{k,j} - t_{i,j} \quad (23)$$

Greedy Insertion

WHILE All customer requests from request bank, D , are assigned or no more requests can be assigned;

 FOR All customer requests from request bank, $i \in D$;

 FOR All vehicles, $k \in S$;

 Calculate (23) for $i \in D$ and for each position in $k \in S$

 ENDFOR

 ENDFOR

 Assign customer request which have least insertion cost to determined position and vehicle, and remove the request from request bank

ENDWHILE

Update and return the new solution

4.4.2 Regret Insertion

Regret heuristics have been used by Potvin and Rousseau (1993). These insertion heuristics try to improve the basic greedy approach through not only using the best position, but also the second and the third best positions (depending on choice). Customer requests are assigned to positions in order to maximize the regret or opportunity cost, computed as the difference between the best and the second (or the third) best position costs. In this respect, the greedy heuristic can be seen as regret-1 heuristic. In our algorithm, we use *regret-2* and *regret-3* insertions.

Inputs:

D : Request bank

S : Set of vehicles used in new solution

$t_{i,j}$: Distance between customer $i \in V_c$ and customer $j \in V_c$

$\Delta_{i,k,j}^l$: Change in objective value by inserting customer k between customers i and j in route l

$cost_k^q$: Minimum q th cost of customer request k , i. e., $cost_k^1 = \min_{i,j,l} \{\Delta_{i,k,j}^l\}$

Regret- m Insertion

WHILE All customer requests are assigned or no more requests can be assigned

 FOR All customer requests from request bank, $i \in D$;

 FOR All vehicles, $k \in S$;

 Calculate (23) for customer request i and all feasible positions in vehicle k

 ENDFOR

 ENDFOR

 Assign customer request which have maximum regret- m value, $\sum_{l=2}^m cost_i^m - cost_i^1$, $i \in D$ to the determined position and vehicle. Remove the assigned request from request bank

ENDWHILE

Update and return the new solution

4.4.3 Time Windows Insertion

Time windows insertion (Lu and Dessouky, 2006) is different from the regret heuristics in the sense that it considers the time windows of the customers and the waiting times of the vehicles as a cost function instead of the distances between customers. The cost function contains the difference between the time window end time of a customer and the arrival time to that customer, which is named as *slack in the time window*; and the time window start time of a customer and the arrival time to that customer, which is called the *waiting time*. We use a simple algorithm to evaluate the cost function. The pseudocode of our time insertion heuristic is shown below.

Inputs:

Let following be a route that starts from depot and visits k customers:

$0, v_1, \dots, v_{i-1}, v_i, v_{i+1}, \dots, v_k, n + 1;$

D : Request bank

S : Set of vehicles used in new solution

a_i : Earliest possible arrival time of node $i \in V$

b_i : Latest possible arrival time of node $i \in V$

h_i : Service time of node $i \in V$

$t_{i,j}$: Distance between node $i \in V$ and node $j \in V$

A_i : Earliest arrival time of node $i \in V$

$$A_{v_i} = \max(A_{v_{i-1}}, b_{v_{i-1}}) + h_{v_{i-1}} + t_{v_{i-1}, v_i} \quad (24)$$

W_i : Waiting time at node $i \in V$

$$W_{v_i} = \max(0, a_{v_{i-1}} - A_{v_{i-1}}) \quad (25)$$

Y_i : Maximal postponed time interval of node $i \in V$

$$Y_{v_i} = \begin{cases} b_{v_k} - \max(a_{v_k}, A_{v_k}), & i = k \\ \min\{b_{v_i} - \max(a_{v_k}, A_{v_k}), Y_{v_{i+1}} + W_{v_{i+1}}\}, & i = k - 1, \dots, 1 \end{cases} \quad (26)$$

Time Windows Insertion

WHILE All customer requests are assigned or no more requests can be assigned
--


```
FOR All customer requests from request bank,  $i \in D$ ;
```

```
    FOR All vehicles,  $k \in S$ ;
```

```
        Calculate (24), (25) and (26) for customer request  $i \in D$  and for all  
        feasible locations in  $k \in S$ 
```

```
    ENDFOR
```

```
ENDFOR
```

```
Assign customer request which have the largest value of summation of (25) and  
(26) to determined position and vehicle
```

```
ENDWHILE
```

```
Update and return the new solution
```

4.5 Local Search Heuristics

After applying removal and insertion heuristics, we keep this solution as the new solution. Then, a local search heuristic is applied to the new solution for improvement. One of two types of local search heuristics is employed, at random.

4.5.1 Swap-Based Local Search Heuristic

Swap-based local search heuristic chooses two customer requests at random and interchanges them. If the new solution is feasible and costs less, the heuristic is re-called. The process is repeated until the new solution is infeasible or its cost larger.

Inputs:

x' : New solution

$f(x')$: Cost of the new solution

Swap Heuristic

DO

Select a customer request at random from the new solution, $i \in x'$.

Select another customer request at random from the new solution, $j \in x' \setminus \{i\}$.

Interchange selected customer requests i and j and calculate the new solution cost.

WHILE The new solution is feasible and cost of the new solution, $f(x')$, is smaller

Update and return the new solution.

4.5.2 Arc Transfer Local Search Heuristic

Arc Transfer starts with selecting two adjacent customer requests from a route at random. Then, selected requests are removed from the route, and are added to the last position of another route. If the new solution is feasible and its cost is less, the heuristic is re-called. Else, the procedure stops.

Inputs:

x' : New solution

S : Set of vehicles used in new solution

$f(x')$: Cost of the new solution

Arc Transfer Heuristic

DO

Select a vehicle at random, $k \in S$

Select two adjacent customer requests at random from selected vehicle k , $i, j \in$

$x'(k)$

Select a vehicle at random, $l \in S \setminus \{k\}$

Insert selected customers to the last position of the vehicle, $i, j \in x'(k), l \in S \setminus \{k\}$ and calculate the new solution cost

WHILE The new solution is feasible and cost of the new solution, $f(x')$, is smaller

Update and return the new solution.

4.6 Feasibility Check

Insertion heuristics assign customer request to a position if and only if the new solution is feasible. The feasibility check contains two criteria: the capacities of the vehicles and the time window constraints.

4.7 Acceptance and Stopping Criteria

When any local search heuristic is applied, we check if the new solution is accepted or not. If the new solution is accepted, it is set as the current solution for the next iteration. As in Ropke and Pisinger (2006, 2007), we use the acceptance criterion from the well-known simulated annealing (Kirkpatrick et al., 1983). The acceptance criterion consists of two parts. If the cost of the new solution is less than the current solution, we set the new solution as current solution. Otherwise, we accept the solution based on the following probability.

$$\text{Acceptance Probability} = e^{-\left(f(x') - f(x)\right)/T} \quad (27),$$

where

$f(x)$: Cost of the current solution.

$f(x')$: Cost of the new solution.

T : Temperature.

c : Cooling rate, $0 < c < 1$.

The new solution is accepted based on value of formula (27). We set the cost of the initial solution as the initial value of the temperature, and the temperature is decreased at each iteration by multiplying it with a constant cooling factor c :

$$T = T * c \quad (28)$$

The algorithm stops after a predetermined number of iterations, w .

4.8 Adaptive Weight Adjustment

Removal and insertion heuristics are selected by considering their contributions to the solution in the past iterations. At the beginning of the algorithm, the scores of all heuristics are identical. The search is divided into segments of predetermined consecutive iterations, and the scores of heuristics are updated at the end of each segment. At each iteration, the selected removal and insertion heuristics are considered as a pair, and their scores are updated to the same value. Three types of scores are used. The first one is the best score, and if the heuristic pair is able to find the new best solution, their scores are increased by the best score. If a heuristic pair finds a solution that is better than the current solution, their scores are increased by the second best score. Finally, if the pair finds a solution worse than the current one but is accepted, their scores are increased by the third best score. The procedure of updating scores and selecting heuristics is shown below:

Inputs:

σ_1 : The best score when the heuristic pair finds the new best solution

σ_2 : The second best score when the heuristic pair finds a better solution than the current

σ_3 : The third best score when heuristics finds a nonimproving accepted solution

φ : Segment, i. e. , number of iterations

$\pi_{i,j}$: The score of heuristic i used in segment j

ρ : Reaction parameter that controls how quickly the weight adjustment reacts to changes

u_i : The number of times that heuristic i has been used in the last segment

Removal and insertion heuristics are selected by a roulette wheel mechanism independent from each other:

$$\text{Selection probability} = \frac{\pi_{i,j}}{\sum_l \pi_{l,j}} \quad (29)$$

At the end of each segment, φ , heuristic scores are updated as:

$$\pi_{i,j+1} = \rho \frac{\pi_{i,j}}{u_i} + (1 - \rho)\pi_{i,j+1} \quad (30)$$

5 COMPUTATIONAL STUDY

In this chapter, we report the comparison of mathematical model results and the heuristic algorithm. We have extended Solomon's benchmark instances for 100 customers for the VRPTW to test our mathematical models and the proposed heuristic algorithm. To create pickup demands, we select a delivery demand from the data set at random and assign that delivery demand to the pickup demand. For pickup demands, we assume that some customers do not need pickups. We fix this ratio to 10% of the total number of customers. Two different approaches are used to have a heterogeneous fleet. In the first approach, we assume that the cost and capacity of vehicles are similar, hence a low variance is obtained in terms of cost and capacity. In the second approach, the vehicles have a high variance in terms of their cost and capacity. The instances having a high variance are found to be harder to solve than the low-variance ones, as can be seen in the computational results.

Solomon's benchmark contains 56 instances with 100 customer requests. The instances have 6 types of categories, as R1, R2, C1, C2, RC1 and RC2. R instances have randomly distributed locations. C instances have clustered locations. RC instances have partially randomly distributed and partially clustered locations. The last number indicates whether instances have short or long planning horizons, and small or large vehicle capacities. The instances containing number 1 have short planning horizons, and others have long planning horizons. We add two number prefixes to the name of each instance to distinguish different problem types. The first number (0) indicates pickup demands and the second number (0 or 1) indicates heterogeneous fleet. For heterogeneous fleet, the instances containing number 0 have low variance and the others have high variance.

5.1 Mathematical Model Results

To compare the proposed mathematical models, we use 10, 15, and 20 customer requests, as it is very difficult to solve even medium-sized instances to optimality. These problem instances are all extracted from the extended 100-customer instances explained above. In the following tables, F1 indicates the Miller-Tucker-Zemlin-based formulation. F2 and F3 indicate our Demand Flow and Time Flow formulations, respectively. The F2 and F3 formulations contain valid inequality (20). For the formulation F1, we consider two cases with and without valid inequality (20).

A time limit of 1800 seconds is imposed on each instance. The solutions which have “*” are the ones that cannot be solved in the allowed time limit. In the tables, “LP Relax.” indicates the value of the LP-relaxation at the root node, “UB” indicates the best upper bound value obtained within the time limit, and “CPU” indicates the computation time in seconds. All experiments are performed on an Intel Core i7-3770 CPU with 3.4GHZ and 8 GB RAM, and IBM ILOG Cplex 12.6 was used.

The solutions of the 112 instances having 10 customer requests are shown in Table 1 and 2. F1 formulation has provided very poor linear relaxation solutions as expected. As it can be seen, the valid inequality improves the lower bound significantly, when we compare the solutions of F1 and F1 with Covering Type Inequality. F1 could not solve 8 of the 56 instances with a low variance, and could not solve 14 of 56 instances with a high variance, due to the poor lower bound performance. For these smallest instances, the best lower bound and the fastest computational time have been obtained by F2 formulation, which is the demand flow formulation. One important observation is that, F1 has found near-optimal solutions with the linear relaxation at the root node, as it can be observed from the CPU time. Average CPU times of 56 instances are only 0.98 seconds and 16.72 seconds for low and high-variances, respectively. The difficulty increases with increasing capacity of vehicles. When the vehicles have larger capacities, the feasible solution space broadens, and the total number of nodes in the branch-and-bound tree increases significantly. Hence, without using problem-specific efficient branching and bounding rules, the necessary CPU time to solve the problem is amplified considerably.

All 10-customer instances with high variance could be solved by only F2 formulation. It seems that on most instances, F1 and F3 have also found the optimal solutions as they reached the time limit. However, the optima could not be verified, as it was not possible to bound and/or prune all nodes in the branch-and-bound tree within the given time. This observation is also valid for 15 and 20-customer request instances.

Table 1: 10 Customer Requests with Low Variance

10 customers	F1			F1 with Covering Type Inequality			F2			F3		
	No:	LP Relax.	UB	CPU (sec.)	LP Relax	UB	CPU sec.	LP Relax	UB	CPU sec.)	LP Relax	UB
00c101	23.36	1572.2	5.85	1547.08	1572.2	0.21	1555.70	1572.2	0.69	1550.3	1572.2	2.05
00c102	23.36	1572.1	397.93	1547.08	1572.1	3.31	1555.52	1572.1	1.14	1548.8	1572.1	10.61
00c103	23.36	1572.1	393.69	1547.08	1572.1	3.01	1555.52	1572.1	1.05	1548.8	1572.1	10.58
00c104	23.36	1569.3	1800.02*	1547.08	1569.3	4.72	1554.73	1569.3	1.05	1548.3	1569.3	17.68
00c105	23.36	1572.2	4.72	1547.08	1572.2	0.27	1554.63	1572.2	0.30	1550.1	1572.2	3.75
00c106	23.36	1572.2	5.03	1547.08	1572.2	0.91	1554.89	1572.2	0.57	1550.1	1572.2	1.83
00c107	23.36	1572.2	4.75	1547.08	1572.2	0.29	1555.76	1572.2	0.49	1549.9	1572.2	2.25
00c108	23.36	1569.3	51.74	1547.08	1569.3	0.85	1555.76	1569.3	0.57	1549.6	1569.3	2.77
00c109	23.36	1569.3	133.46	1547.08	1569.3	1.27	1555.76	1569.3	0.46	1549.2	1569.3	3.42
00c201	101.54	1642.0	1.27	1587.71	1642.0	0.21	1614.83	1642.0	0.43	1588.9	1642.0	2.50
00c202	101.54	1642.0	1119.55	1587.71	1642.0	6.57	1614.83	1642.0	1.21	1588.0	1642.0	90.00
00c203	101.54	1642.0	1129.80	1587.71	1642.0	5.65	1614.83	1642.0	1.19	1588.0	1642.0	81.95
00c204	101.54	1642.0	1800.02*	1587.71	1642.0	12.5	1614.83	1642.0	1.39	1588.0	1642.0	30.44
00c205	101.54	1642.0	33.39	1587.71	1642.0	1.38	1614.83	1642.0	0.52	1588.8	1642.0	16.39
00c206	101.54	1642.0	38.93	1587.71	1642.0	2.28	1613.56	1642.0	0.47	1588.8	1642.0	5.23
00c207	101.54	1642.0	22.25	1587.71	1642.0	1.39	1611.68	1642.0	0.57	1588.9	1642.0	13.59
00c208	101.54	1642.0	15.45	1587.71	1642.0	2.61	1614.83	1642.0	0.80	1588.8	1642.0	20.44
00r101	145.98	1548.5	0.18	1452.18	1548.5	0.19	1463.14	1548.5	0.25	1462.5	1548.5	8.16
00r102	145.98	1483.9	227.58	1452.18	1483.9	5.36	1465.07	1483.9	1.91	1456.9	1483.9	6.76
00r103	145.98	1483.9	228.17	1452.18	1483.9	4.68	1465.07	1483.9	1.88	1456.9	1483.9	6.60
00r104	145.98	1470.5	1215.33	1452.18	1470.5	0.85	1465.30	1470.5	0.46	1456.7	1470.5	1.38
00r105	145.98	1509.5	0.38	1452.18	1509.5	0.18	1464.94	1509.5	0.24	1460.7	1509.5	1.85
00r106	145.98	1479.5	120.55	1452.18	1479.5	2.28	1465.07	1479.5	0.86	1456.9	1479.5	5.73
00r107	145.98	1479.5	119.16	1452.18	1479.5	2.24	1465.07	1479.5	0.85	1456.9	1479.5	5.57
00r108	145.98	1470.5	1800.00*	1452.18	1470.5	0.77	1465.30	1470.5	0.29	1456.7	1470.5	0.88
00r109	145.98	1493.4	5.76	1452.18	1493.4	1.04	1462.45	1493.4	0.77	1458.6	1493.4	1.63
00r110	145.98	1479.5	32.59	1452.18	1479.5	0.75	1462.45	1479.5	0.71	1458.1	1479.5	1.52
00r111	145.98	1479.5	487.35	1452.18	1479.5	2.74	1464.63	1479.5	0.89	1458.1	1479.5	2.89
00r112	145.98	1479.5	1104.62	1452.18	1479.5	3.48	1462.45	1479.5	1.18	1457.6	1479.5	2.49
00r201	145.98	1500.5	6.35	1452.18	1500.5	0.22	1465.41	1500.5	0.13	1457.2	1500.5	4.12

Table 1: 10 Customer Requests with Low Variance (cont.)

10 customers	F1			F1 with Covering Type Inequality			F2			F3		
	No:	LP Relax.	UB	CPU (sec.)	LP Relax	UB	CPU sec.	LP Relax	UB	CPU sec.)	LP Relax	UB
00r202	145.98	1479.5	915.39	1452.18	1479.5	3.73	1462.59	1479.5	1.33	1453.1	1479.5	5.71
00r203	145.98	1479.5	918.73	1452.18	1479.5	3.64	1462.59	1479.5	1.32	1453.1	1479.5	5.70
00r204	145.98	1470.5	1800.05*	1452.18	1470.5	1.75	1462.48	1470.5	0.41	1453.1	1470.5	1.66
00r205	145.98	1483.9	83.82	1452.18	1483.9	1.02	1464.93	1483.9	0.80	1454.3	1483.9	2.24
00r206	145.98	1470.5	1800.01*	1452.18	1470.5	3.31	1463.12	1470.5	0.46	1453.1	1470.5	1.83
00r207	145.98	1470.5	1800.10*	1452.18	1470.5	3.19	1463.12	1470.5	0.49	1453.1	1470.5	1.47
00r208	145.98	1470.5	1800.02*	1452.18	1470.5	1.41	1462.49	1470.5	0.41	1453.1	1470.5	1.33
00r209	145.98	1479.5	110.56	1452.18	1479.5	0.96	1462.66	1479.5	0.61	1453.6	1479.5	2.05
00r210	145.98	1479.5	1179.87	1452.18	1479.5	2.59	1462.74	1479.5	1.08	1453.4	1479.5	7.04
00r211	145.98	1470.5	1800.02*	1452.18	1470.5	1.13	1461.24	1470.5	0.46	1453.3	1470.5	0.91
00rc101	42.39	2407.3	2.92	2291.43	2407.3	0.66	2305.18	2407.3	0.60	2293.7	2407.3	4.86
00rc102	42.39	2399.4	77.26	2291.43	2399.4	38.4	2305.18	2399.4	1.35	2292.2	2399.4	54.43
00rc103	42.39	2399.4	77.33	2291.43	2399.4	44.2	2305.18	2399.4	1.38	2292.2	2399.4	54.36
00rc104	42.39	2399.4	447.79	2291.43	2399.4	63.4	2306.00	2399.4	1.86	2292.2	2399.4	54.54
00rc105	42.39	2403.7	70.56	2291.43	2403.7	8.21	2305.20	2403.7	1.14	2292.4	2403.7	42.95
00rc106	42.39	2407.3	15.87	2291.43	2407.3	16.9	2305.18	2407.3	1.28	2292.4	2407.3	34.79
00rc107	42.39	2405.8	52.89	2291.43	2405.8	12.1	2306.26	2405.8	1.72	2292.2	2405.8	69.99
00rc108	42.39	2399.4	192.82	2291.43	2399.4	25.4	2306.26	2399.4	1.16	2291.9	2399.4	51.66
00rc201	42.39	2407.3	2.70	2291.43	2407.3	0.96	2305.44	2407.3	0.68	2293.6	2407.3	24.06
00rc202	42.39	2399.4	155.96	2291.43	2399.4	47.6	2305.18	2399.4	0.97	2291.8	2399.4	69.57
00rc203	42.39	2399.4	165.07	2291.43	2399.4	48.9	2305.18	2399.4	0.97	2291.8	2399.4	69.27
00rc204	42.39	2399.4	303.01	2291.43	2399.4	64.3	2304.82	2399.4	2.05	2291.8	2399.4	110.52
00rc205	42.39	2405.8	314.74	2291.43	2405.8	26.6	2303.99	2405.8	3.50	2291.8	2405.8	174.26
00rc206	42.39	2407.3	48.24	2291.43	2407.3	9.29	2305.18	2407.3	1.85	2292.6	2407.3	66.04
00rc207	42.39	2407.3	106.04	2291.43	2407.3	8.16	2306.27	2407.3	2.13	2292.1	2407.3	72.78
00rc208	42.39	2399.4	442.24	2291.43	2399.4	63.7	2303.34	2399.4	1.5	2291.5	2399.4	108.24
Avg.			481.96			10.2			0.98			26.02

Table 2: 10 Customer Requests with High Variance

10 customers	F1			F1 with Covering Type Inequality			F2			F3		
	No:	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)	LP Relax	UB
01c101	258.34	939.60	1.15	785.25	939.60	0.37	897.39	939.60	0.46	783.36	939.60	38.30
01c102	24.75	939.60	820.19	780.96	939.60	1146.79	893.02	939.60	4.82	781.33	939.60	1800*
01c103	24.75	939.60	812.42	780.96	939.60	1134.53	893.02	939.60	4.08	781.33	939.60	1800*
01c104	23.36	937.39	1800.00*	780.96	937.39	1800.02*	891.65	937.39	12.30	781.18	937.39	1800*
01c105	258.21	939.60	0.56	785.12	939.60	0.81	897.27	939.60	1.81	783.03	939.60	56.87
01c106	258.34	939.60	3.40	785.25	939.60	0.72	897.39	939.60	0.35	783.12	939.60	153.5
01c107	258.21	939.60	1.00	785.12	939.60	1.34	897.27	939.60	0.97	782.85	939.60	293.7
01c108	25.41	938.86	140.17	782.94	938.86	144.86	896.15	938.86	10.52	782.60	938.86	261.7
01c109	24.33	938.74	26.55	782.94	938.74	29.76	894.41	938.74	1.24	782.24	938.74	1800*
01c201	394.66	1002.27	0.97	852.27	1002.27	0.30	965.13	1002.27	0.63	832.30	1002.27	251.6
01c202	118.14	1002.27	1320.53	833.32	1002.27	1800.00*	953.19	1002.27	3.08	827.83	1002.27	1800*
01c203	118.14	1002.27	1373.10	833.32	1002.27	1800.00*	953.19	1002.27	2.89	827.83	1002.27	1800*
01c204	101.54	1002.27	1800.03*	824.54	1002.27	1800.00*	948.49	1002.27	5.78	826.15	1002.27	1802*
01c205	317.17	1002.27	17.86	830.81	1002.27	38.37	954.81	1002.27	1.55	830.91	1002.27	1338
01c206	135.76	1002.27	51.73	830.81	1002.27	25.94	954.15	1002.27	2.20	830.11	1002.27	1800*
01c207	330.34	1002.27	19.83	832.34	1002.27	26.90	958.72	1002.27	0.98	830.98	1002.27	1616
01c208	124.62	1002.27	133.58	830.81	1002.27	115.78	954.15	1002.27	3.17	828.53	1002.27	1800*
01r101	853.09	1087.34	0.14	1003.09	1087.34	0.17	1011.73	1087.34	0.29	907.46	1087.34	14.01
01r102	145.98	983.92	175.81	879.40	983.92	45.51	887.35	983.92	165.88	884.22	983.92	315.80
01r103	145.98	983.92	176.45	879.40	983.92	46.24	887.35	983.92	160.76	884.22	983.92	342.46
01r104	145.98	908.29	983.73	879.40	908.29	5.76	887.35	908.29	5.56	884.01	908.29	3.41
01r105	664.92	1009.53	0.31	964.92	1009.53	0.25	964.92	1009.53	1.41	889.32	1009.53	6.54
01r106	145.98	941.11	437.94	879.40	941.11	42.51	887.35	941.11	107.68	884.22	941.11	149.97
01r107	145.98	941.11	435.70	879.40	941.11	42.68	887.35	941.11	107.62	884.22	941.11	146.54
01r108	145.980	908.29	1668.12	879.40	908.29	7.97	887.35	908.29	4.84	884.01	908.29	30.43
01r109	400.81	993.44	1.70	900.83	993.44	1.51	908.84	993.44	17.43	886.48	993.44	96.24
01r110	171.23	947.85	37.14	885.47	947.85	5.09	890.19	947.85	8.29	885.70	947.85	10.36
01r111	145.98	941.11	310.80	879.40	941.11	62.85	887.35	941.11	106.52	885.52	941.11	188.91
01r112	145.98	924.95	1390.62	879.40	924.95	21.90	887.35	924.95	120.37	885.15	924.95	85.07

Table 2: 10 Customer Requests with High Variance (cont.)

10 customers	F1			F1 with Covering Type Inequality			F2			F3		
	No:	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)	LP Relax	UB
01r201	453.87	949.74	1.28	944.12	949.74	0.20	944.68	949.74	0.44	885.44	949.74	3.44
01r202	145.98	917.24	1308.94	879.40	917.24	11.44	887.35	917.24	4.76	880.40	917.24	35.95
01r203	145.98	917.24	1294.29	879.40	917.24	11.54	887.35	917.24	4.72	880.40	917.24	35.82
01r204	145.98	905.07	1800.00*	879.40	905.07	5.94	887.35	905.07	3.45	880.39	905.07	9.47
01r205	207.51	920.43	56.89	890.56	920.43	0.50	899.51	920.43	0.94	882.62	920.43	35.78
01r206	145.98	905.07	1800.03*	879.40	905.07	2.14	887.35	905.07	1.64	880.40	905.07	4.20
01r207	145.98	905.07	1800.02*	879.40	905.07	2.09	887.35	905.07	1.53	880.40	905.07	4.23
01r208	145.98	908.29	1800.00*	879.40	905.07	3.18	887.35	905.07	1.88	880.39	905.07	3.02
01r209	171.23	905.07	259.68	885.47	905.07	1.31	890.19	905.07	0.83	881.07	905.07	7.46
01r210	145.98	917.24	1800.00*	879.40	917.24	15.10	887.35	917.24	6.25	880.62	917.24	57.57
01r211	145.98	905.07	1800.00*	879.40	905.07	13.65	895.46	905.07	2.22	880.56	905.07	4.73
01rc101	738.93	1485.25	3.12	1283.29	1485.25	1.58	1458.60	1485.25	0.91	1276.75	1485.25	570.4
01rc102	110.89	1477.05	630.63	1271.01	1477.05	693.08	1448.31	1477.05	2.91	1272.23	1477.05	1800*
01rc103	110.89	1477.05	634.56	1271.01	1477.05	687.11	1448.31	1477.05	2.67	1272.23	1477.05	1800*
01rc104	42.39	1471.88	1800.00*	1270.33	1471.88	1800.00*	1430.21	1471.88	2.97	1272.23	1471.88	1800*
01rc105	127.27	1519.52	103.29	1272.08	1519.52	420.09	1449.49	1519.52	9.44	1271.34	1523.41	1800*
01rc106	53.60	1477.05	28.03	1277.21	1477.05	104.50	1432.93	1477.05	1.97	1273.04	1477.05	1800*
01rc107	42.39	1471.88	397.21	1267.94	1471.88	375.89	1430.13	1471.88	4.31	1272.10	1471.88	1800*
01rc108	42.39	1471.88	1800.02*	1267.94	1471.88	1800.00*	1430.13	1471.88	2.52	1272.01	1471.88	1807*
01rc201	147.67	1480.13	52.26	1283.29	1480.13	132.74	1454.43	1480.13	0.83	1274.95	1480.13	1785
01rc202	110.54	1474.91	1800.00*	1271.01	1474.91	1800.00*	1448.31	1474.91	2.20	1270.07	1474.91	1800*
01rc203	110.54	1474.91	1800.00*	1271.01	1474.91	1800.00*	1448.31	1474.91	1.66	1270.07	1474.91	1800*
01rc204	42.39	1471.88	1800.00*	1270.33	1471.88	1800.02*	1430.21	1471.88	3.55	1269.62	1471.88	1800*
01rc205	112.81	1474.91	413.39	1272.08	1474.91	671.02	1448.59	1474.91	1.66	1271.18	1475.83	1800*
01rc206	57.33	1477.10	196.76	1281.46	1477.10	222.57	1435.34	1477.10	1.75	1272.34	1477.10	1801*
01rc207	48.14	1471.88	836.29	1272.08	1471.88	1271.05	1432.78	1471.88	2.05	1270.34	1471.88	1803*
01rc208	42.39	1471.88	1800.00*	1267.94	1471.88	1800.06*	1430.13	1471.88	2.89	1268.78	1471.88	1800*
Avg.			729.70			457.06			16.72			881.76

As the 10-customer instances do not seem to be promising for F1 and F3 formulations, we only solved the 15-customer instances with F1 with valid inequality, and with F2 formulation. The results of the 112 instances can be seen in Tables 3 and 4. These instances could be solved within a reasonable time by the two formulations. On the average, F2 formulation was better in terms of lower bound and CPU time. On low variance instances, both formulations have found optimal solutions. However, on high variance instances, F1 with valid inequality has found only 11 optimal solutions out of 56. On the other hand, F2 has found 37 optimal solutions. Similar to the results of the 10-customer instances, F2 has found near-optimal solutions for the linear relaxation at the root node.

Finally, we solved 20-customer instances by using only F2 formulation. The results of these 112 instances can be seen in Tables 5 and 6. In the tables, “LB” indicates the best lower bound and “GAP” indicates the gap in percentage between the value of the linear relaxation at the root node and the optimal value. F2 has found optimal solutions in 26 out of 56 instances with low variance. For the instances reaching the time limit, the gaps are very small. A majority, namely 28 of the 30 instances is less than %10 far from the optimum, and the average gap is %3.1. The number of instances that cannot be solved increases in high variance instances, as expected. Only 10 of 56 instances have been solved to optimality, where the largest gap from the optimum is %31.11, and the average gap is %6.46. Although the performance of F2 is much better than the other formulations, the results show that solving even the small sized instances is hard. Therefore, to solve larger instances, a heuristic algorithm is evidently crucial. In the next section, we compare the results of the F2 formulation with the solutions by our heuristic approach. Then, we analyze the quality of the proposed heuristic for larger sized instances.

Table 3: 15 Customer Requests with Low Variance

15 customers	F1 with Covering Type Inequality			F2		
No:	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)
00c101	2641.29	2691.37	3.85	2659.36	2691.37	4.73
00c102	2604.58	2661.17	19.06	2650.84	2661.17	7.19
00c103	2604.58	2661.17	19.19	2650.23	2661.17	11.68
00c104	2604.58	2660.61	28.52	2649.52	2660.61	45.43
00c105	2641.29	2690.68	5.83	2657.53	2690.68	6.05
00c106	2641.29	2691.37	6.41	2659.36	2691.37	5.82
00c107	2637.10	2689.65	56.94	2657.33	2689.65	8.14
00c108	2629.54	2662.40	12.03	2653.09	2662.40	10.87
00c109	2604.58	2661.17	43.07	2650.56	2661.17	20.44
00c201	2674.28	2716.94	3.15	2704.21	2716.94	1.16
00c202	2632.58	2715.22	160.06	2696.47	2715.22	17.38
00c203	2632.58	2715.22	112.17	2696.47	2715.22	18.21
00c204	2632.58	2715.22	255.56	2695.14	2715.22	80.04
00c205	2627.23	2715.22	12.31	2694.33	2715.22	6.52
00c206	2627.23	2715.22	42.88	2694.33	2715.22	10.25
00c207	2627.23	2714.01	121.49	2695.36	2714.01	6.97
00c208	2627.23	2714.01	20.83	2694.24	2714.01	6.57
00r101	2359.56	2440.48	0.44	2360.44	2440.48	0.98
00r102	2266.60	2372.00	400.21	2283.31	2372.00	122.98
00r103	2255.39	2372.00	591.95	2276.82	2372.00	155.02
00r104	2251.84	2339.90	50.28	2272.74	2339.90	64.86
00r105	2328.49	2386.08	0.69	2329.83	2386.08	1.36
00r106	2261.96	2344.12	33.06	2281.10	2344.12	37.19
00r107	2255.39	2344.12	26.44	2276.82	2344.12	62.68
00r108	2251.84	2339.90	95.91	2272.74	2339.90	89.51
00r109	2261.40	2344.12	9.83	2279.89	2344.12	5.79
00r110	2251.84	2339.90	57.97	2273.14	2339.90	62.68
00r111	2255.39	2339.90	72.10	2276.82	2339.90	39.47
00r112	2251.84	2339.90	165.20	2272.74	2339.90	218.56
00r201	2311.48	2370.49	4.31	2314.73	2370.49	4.07

Table 3: 15 Customer Requests with Low Variance (cont.)						
15 customers	F1 with Covering Type Inequality			F2		
No:	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)
00r202	2255.39	2339.90	48.91	2276.82	2339.90	9.03
00r203	2255.39	2339.90	54.41	2276.82	2339.90	20.51
00r204	2251.84	2339.90	137.09	2272.74	2339.90	29.00
00r205	2261.40	2344.12	55.68	2280.71	2344.12	48.45
00r206	2255.39	2339.90	86.56	2276.82	2339.90	27.78
00r207	2255.39	2339.90	104.08	2276.82	2339.90	108.05
00r208	2251.84	2339.90	117.14	2272.74	2339.90	304.39
00r209	2261.28	2339.90	31.00	2277.94	2339.90	53.91
00r210	2255.39	2339.90	29.36	2276.82	2339.90	31.59
00r211	2251.84	2339.90	172.35	2272.74	2339.90	58.91
00rc101	3594.82	3639.95	1.09	3604.89	3639.95	2.96
00rc102	3586.31	3632.09	22.46	3598.72	3632.09	14.37
00rc103	3586.31	3632.09	93.38	3598.05	3632.09	78.53
00rc104	3586.31	3631.74	135.61	3596.63	3631.74	51.57
00rc105	2950.00	3636.37	25.02	3599.79	3636.37	16.15
00rc106	2950.00	3637.05	11.47	3598.02	3637.05	15.36
00rc107	3583.59	3636.77	97.24	3595.83	3636.77	74.30
00rc108	2950.00	3631.74	189.03	3595.83	3631.74	27.56
00r201	2950.00	3639.95	5.83	3604.89	3639.95	5.77
00rc202	3586.31	3632.09	84.40	3598.72	3632.09	13.90
00rc203	3586.31	3632.09	69.55	3598.05	3632.09	128.48
00rc204	2950.00	3631.74	127.22	3596.63	3631.74	16.89
00rc205	3586.31	3637.12	45.90	3600.87	3637.12	48.95
00rc206	3590.64	3639.53	40.09	3598.69	3639.53	44.83
00rc207	3584.69	3636.77	109.56	3596.27	3636.77	86.38
00rc208	2950.00	3631.74	201.01	3595.83	3631.74	198.65
Avg.			80.84			47.30

Table 4: 15 Customer Requests with High Variance

15 customers	F1 with Covering Type Inequality			F2		
No:	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)
01c101	1329.40	1635.30	0.89	1591.94	1635.30	3.12
01c102	1329.76	1635.30	1800.19*	1591.38	1635.30	41.91
01c103	1329.76	1632.72	1800.27*	1591.24	1632.72	53.16
01c104	1329.63	1633.12	1800.32*	1592.35	1632.72	84.29
01c105	1329.63	1635.30	64.03	1592.02	1635.30	8.68
01c106	1329.40	1635.30	47.41	1592.64	1635.30	7.34
01c107	1329.53	1634.43	1276.14	1592.49	1634.43	7.87
01c108	1329.40	1635.30	1801.99*	1595.18	1634.43	38.12
01c109	1368.49	1630.97	1800.07*	1592.96	1630.97	36.43
01c201	1354.75	1682.86	5.23	1641.16	1682.86	1.11
01c202	1354.01	1679.90	1800.41*	1638.05	1679.90	23.83
01c203	1355.64	1678.15	1800.04*	1639.66	1678.15	22.75
01c204	1368.49	1698.66	1800.08*	1640.39	1678.15	68.35
01c205	1354.01	1678.15	1104.4	1638.77	1678.15	7.63
01c206	1352.00	1678.15	1800.02*	1640.14	1678.15	4.78
01c207	1349.97	1678.15	1800.08*	1641.16	1678.15	6.09
01c208	1282.08	1678.15	1800.02*	1637.41	1678.15	6.34
01r101	1287.09	1683.83	2.02	1361.38	1683.83	0.77
01r102	1287.30	1523.99	1801.61*	1361.46	1548.55	1802.27*
01r103	1289.81	1643.38	1802.22*	1360.83	1601.63	1800.05*
01r104	1279.77	1491.89	1800.05*	1361.40	1491.89	1802.28*
01r105	1287.27	1566.94	22.78	1361.46	1566.94	178.78
01r106	1287.58	1496.11	1800.02*	1361.46	1496.11	1803.63*
01r107	1289.81	1496.11	1800.63*	1361.46	1583.06	1804.84*
01r108	1286.56	1447.52	1800.10*	1361.23	1445.08	1800.05*
01r109	1286.67	1449.07	897.54	1361.23	1449.07	464.56
01r110	1285.35	1491.89	1800.05*	1361.16	1504.67	1801.32*
01r111	1283.12	1491.89	1800.12*	1361.39	1509.93	1803.47*
01r112	1274.59	1564.88	1800.10*	1361.46	1481.21	1813.03*
01r201	1288.91	1460.80	496.23	1360.99	1460.80	63.34

Table 4: 15 Customer Requests with High Variance (cont.)						
15 customers	F1 with Covering Type Inequality			F2		
No:	LP Relax	UB	CPU (sec.)	LP Relax	UB	CPU (sec.)
01r202	1288.91	1465.76	1800.09*	1360.76	1483.35	1822.08*
01r203	1287.88	1491.89	1800.04*	1361.27	1491.89	1808.49*
01r204	1282.78	1429.33	1800.05*	1361.04	1442.57	1805.28*
01r205	1286.67	1494.32	1800.07*	1361.06	1449.05	1800.52*
01r206	1287.09	1471.47	1800.05*	1361.39	1439.45	1807.68*
01r207	1289.40	1499.27	1800.04*	1360.51	1471.47	1808.51*
01r208	1289.40	1508.64	1800.05*	1361.24	1451.58	1802.21*
01r209	1280.14	1507.79	1800.05*	1361.27	1449.07	1803.47*
01r210	1287.44	1491.89	1800.04*	1361.38	1504.38	1807.81*
01r211	1281.34	1491.89	1800.05*	1360.82	1444.96	1801.46*
01rc101	1702.66	2153.27	934.30	2105.60	2153.27	4.17
01rc102	1689.92	2147.78	1800.02*	2106.03	2144.01	29.86
01rc103	1692.25	2144.01	1800.02*	2105.89	2144.01	10.60
01rc104	1702.66	2141.02	1800.07*	2106.33	2141.02	17.02
01rc105	1702.20	2144.93	1802.35*	2105.92	2144.01	30.49
01rc106	1702.66	2141.02	1800.10*	2106.35	2141.02	6.71
01rc107	1687.15	2139.91	1800.02*	2106.59	2139.91	71.66
01rc108	1702.66	2139.91	1800.02*	2105.40	2139.91	120.75
01rc201	1693.24	2153.27	907.10	2106.12	2153.27	13.08
01rc202	1692.97	2144.01	1800.04*	2106.21	2144.01	15.50
01rc203	1695.35	2144.01	1800.41*	2106.32	2144.01	43.90
01rc204	1702.66	2141.32	1800.37*	2106.89	2141.02	23.83
01rc205	1694.91	2144.01	1800.07*	2106.32	2144.01	9.76
01rc206	1689.58	2146.24	1800.04*	2106.35	2146.24	39.10
01rc207	1702.44	2139.91	1800.10*	2106.41	2139.91	19.38
01rc208	1696.77	2139.91	1800.18*	2106.31	2139.91	27.60
Avg.			1517.36			641.26

Table 5: 20 Customer Requests with Low Variance

20 customers	F2				
	No:	LP Relax.	LB	UB	GAP (%)
00c101	3327.52	3337.58	3337.58	-	5.76
00c102	3320.94	3278.71	3337.58	-	583.18
00c103	3308.30	3329.56	3337.58	0.24	1802.42*
00c104	3307.08	3334.55	3337.54	0.09	1800.11*
00c105	3317.81	3337.58	3337.58	-	17.56
00c106	3327.52	3337.58	3337.58	-	4.10
00c107	3313.83	3337.58	3337.58	-	12.96
00c108	3309.39	3334.74	3334.74	-	223.39
00c109	3308.20	3334.74	3334.74	-	369.75
00c201	3376.18	3395.20	3395.20	-	3.40
00c202	3355.92	3390.79	3390.79	-	560.06
00c203	3353.81	3377.16	3390.79	0.40	1800.08*
00c204	3352.71	3378.24	3390.79	0.37	1800.70*
00c205	3355.93	3395.20	3395.20	-	230.51
00c206	3355.76	3395.20	3395.20	-	864.09
00c207	3362.63	3395.20	3395.20	-	873.03
00c208	3355.61	3395.20	3395.20	-	1230.11
00r101	2691.25	2890.37	2890.37	-	345.31
00r102	2486.57	2510.03	2790.25	10.04	1801.06*
00r103	2453.68	2470.04	2824.52	12.55	1819.06*
00r104	2453.68	2469.87	2814.61	12.55	1800.56*
00r105	2509.83	2620.95	2620.95	-	1372.73
00r106	2467.73	2487.77	2677.23	7.08	1818.52*
00r107	2453.68	2493.30	2580.38	3.73	1800.73*
00r108	2453.68	2467.10	2829.43	12.81	1800.53*
00r109	2470.43	2487.64	2778.74	10.48	1800.07*
00r110	2453.70	2471.12	2668.16	7.38	1800.72*
00r111	2454.93	2471.64	2646.49	6.61	1800.86*
00r112	2453.68	2509.19	2684.23	6.52	1800.94*
00r201	2495.07	2520.84	2686.16	6.15	1800.93*
00r202	2458.14	2478.03	2726.14	9.10	1800.11*

Table 5: 20 Customer Requests with Low Variance (cont.)

20 customers	F2				
No:	LP Relax.	LB	UB	GAP (%)	CPU (sec.)
00r203	2453.68	2470.09	2603.79	5.13	1800.43*
00r204	2453.68	2471.08	2673.36	7.57	1800.52*
00r205	2468.64	2486.31	2666.11	6.74	1800.76*
00r206	2454.93	2472.94	2665.68	7.23	1800.15*
00r207	2453.68	2468.94	2692.89	8.32	1800.02*
00r208	2453.68	2468.42	2685.00	8.07	1800.88*
00r209	2456.59	2474.68	2636.87	6.15	1803.15*
00r210	2454.93	2471.04	2721.70	9.21	1800.45*
00r211	2453.68	2468.19	2652.45	6.95	1800.02*
00rc101	4286.36	4373.20	4456.54	1.87	1800.47*
00rc102	4228.63	4253.70	4253.70	-	185.14
00rc103	4224.44	4246.97	4246.97	-	591.81
00rc104	4224.40	4243.10	4246.97	0.09	1800.05*
00rc105	4229.60	4253.63	4253.63	-	412.54
00rc106	4229.79	4255.16	4255.16	-	715.03
00rc107	4222.76	4243.40	4246.97	0.02	1800.02*
00rc108	4222.76	4238.15	4246.97	0.21	1800.05*
00rc201	4238.24	4290.38	4290.38	-	6.43
00rc202	4228.56	4253.63	4253.63	-	390.35
00rc203	4224.44	4246.97	4246.97	-	886.28
00rc204	4224.40	4246.97	4246.97	-	869.55
00rc205	4229.37	4253.63	4253.63	-	366.81
00rc206	4231.39	4255.44	4255.44	-	48.43
00rc207	4223.59	4243.67	4246.97	0.08	1800.02*
00rc208	4222.76	4236.67	4246.97	0.24	1804.47*
Avg.				3.10	1196.91

Table 6: 20 Customer Requests with High Variance

20 customers	F2				
No:	LP Relax.	LB	UB	GAP (%)	CPU (sec.)
01c101	1666.22	1695.93	1695.93	-	63.22
01c102	1658.46	1669.42	1693.40	1.42	1800.24*
01c103	1645.77	1660.65	1824.41	8.98	1804.50*
01c104	1644.73	1659.90	1890.49	12.20	1805.05*
01c105	1665.89	1689.33	1689.33	-	1142.49
01c106	1666.22	1695.93	1695.93	-	213.79
01c107	1654.56	1689.33	1689.33	-	353.35
01c108	1644.99	1672.61	1689.33	0.99	1800.10*
01c109	1643.96	1661.78	1690.23	1.68	1803.10*
01c201	1712.24	1729.49	1729.49	-	5.59
01c202	1698.61	1719.74	1729.49	0.56	1800.26*
01c203	1688.59	1711.60	1730.16	1.07	1800.15*
01c204	1686.76	1708.69	1730.89	1.28	1800.05*
01c205	1699.51	1729.49	1729.49	-	36.04
01c206	1696.21	1724.51	1729.49	0.29	1800.05*
01c207	1694.28	1727.8	1727.80	-	922.98
01c208	1694.15	1726.04	1729.88	0.22	1800.18*
01r101	2113.63	2119.33	2119.33	-	1.11
01r102	1388.63	1407.94	2043.84	31.11	1819.93*
01r103	1343.59	1355.75	1700.13	20.26	1809.35*
01r104	1343.58	1354.57	1801.20	24.30	1814.11*
01r105	1443.46	1626.39	1760.77	7.63	1800.10*
01r106	1361.34	1379.50	1799.53	23.34	1805.19*
01r107	1343.59	1356.19	1601.63	15.32	1803.20*
01r108	1343.58	1354.65	1690.20	19.85	1804.30*
01r109	1371.68	1386.11	1703.37	18.63	1817.28*
01r110	1345.92	1356.44	1677.51	19.14	1810.29*
01r111	1345.71	1357.89	1585.08	14.33	1812.88*
01r112	1343.58	1353.66	1417.74	4.52	1800.05*
01r201	1408.23	1456.4	1456.40	-	991.00
01r202	1348.47	1368.82	1635.15	16.29	1803.94*

Table 6: 20 Customer Requests with High Variance (cont.)

20 customers	F2				
	No:	LP Relax.	LB	UB	GAP (%)
01r203	1343.59	1355.30	1455.95	6.91	1812.41*
01r204	1343.58	1354.17	1376.46	1.62	1803.44*
01r205	1368.36	1379.15	1607.48	14.20	1807.06*
01r206	1345.78	1357.99	1557.81	12.83	1807.48*
01r207	1343.59	1354.82	1659.93	18.38	1805.61*
01r208	1343.58	1357.15	1376.46	1.40	1802.94*
01r209	1351.62	1364.10	1421.37	4.03	1804.36*
01r210	1345.71	1358.85	1431.83	5.10	1809.26*
01r211	1343.58	1354.57	1384.84	2.19	1804.80*
01rc101	2175.69	2327.57	2327.57	-	809.21
01rc102	2123.02	2253.43	2316.38	2.72	1801.97*
01rc103	2100.47	2203.11	2304.34	4.39	1820.46*
01rc104	2099.76	2230.35	2298.16	2.95	1800.18*
01rc105	2123.43	2220.08	2322.61	4.41	1802.75*
01rc106	2110.33	2243.74	2310.46	2.89	1800.07*
01rc107	2106.63	2234.00	2307.64	3.19	1805.42*
01rc108	2096.38	2204.92	2298.16	4.06	1802.80*
01rc201	2132.30	2287.06	2321.90	1.50	1803.75*
01rc202	2117.94	2226.45	2311.33	3.67	1802.81*
01rc203	2100.47	2204.03	2311.34	4.64	1816.73*
01rc204	2099.76	2203.59	2309.58	4.59	1824.27*
01rc205	2121.52	2265.52	2313.45	2.07	1800.46*
01rc206	2112.86	2220.74	2308.77	3.81	1802.88*
01rc207	2110.25	2227.72	2302.21	3.24	1802.10*
01rc208	2096.38	2212.87	2300.09	3.79	1803.67*
Avg.				6.46	1564.29

5.2 Heuristic Results

The proposed algorithm uses 11 different parameters in its execution as shown in Table 7, of which values are to be determined. To determine good values for the algorithm parameters, a pilot experiment was performed. Only one parameter's value was changed while the others were fixed. The ALNS algorithm was executed for 10 replications for each setting on all instances that contain 50 customer requests, and the best combination of parameters was chosen. Table 7 lists the best values.

The most important parameter is r , which determines the number of customers to be removed from the current solution at each iteration. As shown by Pisinger and Ropke (2006, 2007), removing a large number of requests is not necessary to find good solutions. If r is too large, the heuristic may try to solve the problem from the scratch at each iteration. If r is too small on the other hand, the heuristic is not able to move to different solutions, and most probably trap in local optima. The value of this parameter is therefore chosen as instance-dependent. If the total number of customer requests is below 100 ($n \leq 100$), r is chosen at random in the interval $[0.1n, 0.4n]$, and for larger instances r is chosen at random in the interval $[20, 40]$.

Table 7: Parameters of ALNS

Parameter	Definition	Best Value
r	Number of customers removed at each iteration	$r \in [0.1n, 0.4n]$ or $r \in [20, 40]$
μ	Penalizes the solution if there is a request in request bank	10^{-5}
α	Effect of distance value at Shaw removal	0.2
β	Effect of difference of demands value at Shaw removal	0.5
T_{start}	Initial value of temperature	Cost of the initial solution
c	Cooling rate	0.99975
σ^1	Score of finding the new best solution	50

σ^2	Score of finding a better solution than current one	20
σ^3	Score of finding a non-improving solution which is accepted	10
φ	Number of consecutive iterations in a segment	50
ρ	Reaction factor	0.01

ALNS algorithm was executed for 10 replications for each instance. The best and the average solutions are reported. The maximum number of iterations for ALNS is fixed to 5000 iterations. This means that, if only one vehicle is removed from the solution at the end of the algorithm, 10000 ALNS steps were executed.

The comparison between F2 formulation and ALNS on 20-request instances is shown in Tables 8 and 9. In the tables, “GAP” indicates the percentage gap between the solution of F2 (UB_{F_2}) formulation and the best solution of the ALNS ($UB_{ALNS_{Best}}$), which can be computed as $GAP = \%100 * \frac{UB_{ALNS_{Best}} - UB_{F_2}}{UB_{ALNS_{Best}}}$.

Our first observation is that the proposed algorithm works better on harder instances such as r instances, which have short planning horizon and low demands. Even when a better solution is obtained by F2, the resulting gap is very small. The largest gap is %8.95 whereas the smallest is %-10.30. On the average, ALNS finds better solutions, and the average gap is %-0.90 for low-variance instances. For high variance instances, the largest gap is %29.74, which may be due the tightness of the problem. F2 can solve the problem very quickly when the constraints are very tight. But it should be noted that the fleet is heterogeneous, which means the selection of vehicles to be used is extremely important. If ALNS starts by selecting the wrong vehicles, at the end of the search, the algorithm may not be able to remove any vehicles, leading to a poor solution. For other instances, the gap is never so high, and on the average ALNS finds better solutions. The average gap for the high-variance instances is %-0.66. The algorithm performs very robustly in terms of computation time for both settings. The longest computation times of ALNS are less than 12 seconds for low variance, and less than 11 seconds for high variance instances. On the average, the CPU time of low and high variance instances are 6.03 and 5.17 seconds respectively, which are significantly shorter than the computation time of F2.

Table 8: Comparison of F2 and ALNS

20 customers No:	F2		ALNS (Best over 10 reps.)		ALNS (Avg. over 10 reps.)	
	UB	CPU (sec.)	UB	GAP (%)	UB	CPU (sec.)
00c101	3337.58	5.76	3337.58	-	3342.338	6.28
00c102	3337.58	583.18	3337.58	-	3343.673	7.04
00c103	3337.58	1802.42*	3337.58	-	3340.713	8.29
00c104	3337.54	1800.11*	3337.54	-	3339.334	5.04
00c105	3337.58	17.56	3337.58	-	3341.491	6.99
00c106	3337.58	4.10	3337.58	-	3340.946	6.76
00c107	3337.58	12.96	3337.58	-	3343.145	6.65
00c108	3334.74	223.39	3339.78	0.1509	3442.925	2.08
00c109	3334.74	369.75	3339.78	0.1509	3380.974	6.89
00c201	3395.20	3.40	3420.17	0.7300	3422.299	7.01
00c202	3390.79	560.06	3390.79	-	3393.333	7.33
00c203	3390.79	1800.08*	3390.79	-	3498.989	2.20
00c204	3390.79	1800.70*	3394.5	0.1092	3508.923	6.83
00c205	3395.20	230.51	3494.76	2.8488	3509.003	2.11
00c206	3395.20	864.09	3424.33	0.8506	3438.826	4.36
00c207	3395.20	873.03	3425.18	0.8752	3434.987	6.29
00c208	3395.20	1230.11	3425.63	0.8883	3432.439	4.75
00r101	2890.37	345.31	3174.82	8.9595	3186.163	11.47
00r102	2790.25	1801.06*	2860.68	2.4620	2873.066	6.35
00r103	2824.52	1819.06*	2582.43	-9.3745	2629.694	6.36
00r104	2814.61	1800.56*	2569.65	-9.5328	2617.651	6.58
00r105	2620.95	1372.73	2854.07	8.1679	2862.523	7.00
00r106	2677.23	1818.52*	2611.51	-2.5165	2624.882	6.76
00r107	2580.38	1800.73*	2581.18	0.0309	2624.3	8.83
00r108	2829.43	1800.53*	2565.06	-10.3066	2581.041	10.36
00r109	2778.74	1800.07*	2633.39	-5.5195	2654.709	6.66
00r110	2668.16	1800.72*	2595.77	-2.7887	2664.775	7.18
00r111	2646.49	1800.86*	2582.97	-2.4591	2646.359	7.35
00r112	2684.23	1800.94*	2571.46	-4.3854	2579.301	7.64
00r201	2686.16	1800.93*	2618.08	-2.6003	2636.086	7.44
00r202	2726.14	1800.11*	2588.68	-5.3100	2604.568	6.70

Table 8: Comparison of F2 and ALNS (cont.)						
20 customers	F2		ALNS (Best over 10 reps.)		ALNS (Avg. over 10 reps.)	
No:	UB	CPU (sec.)	UB	GAP (%)	UB	CPU (sec.)
00r203	2603.79	1800.43*	2571.19	-1.2679	2611.109	8.27
00r204	2673.36	1800.52*	2555.18	-4.6251	2567.057	9.23
00r205	2666.11	1800.76*	2606.31	-2.2944	2614.862	7.78
00r206	2665.68	1800.15*	2577.19	-3.4335	2591.104	7.23
00r207	2692.89	1800.02*	2574.59	-4.5949	2582.942	7.79
00r208	2685.00	1800.88*	2568.00	-4.5560	2601.478	7.68
00r209	2636.87	1803.15*	2572.64	-2.4966	2593.964	7.35
00r210	2721.70	1800.45*	2577.95	-5.5761	2617.763	8.10
00r211	2652.45	1800.02*	2551.78	-3.9450	2588.63	8.54
00rc101	4456.54	1800.47*	4353.76	-2.3607	4363.898	3.71
00rc102	4253.70	185.14	4353.70	2.2968	4396.669	3.66
00rc103	4246.97	591.81	4355.90	2.5007	4398.88	3.78
00rc104	4246.97	1800.05*	4349.71	2.3619	4398.838	3.88
00rc105	4253.63	412.54	4253.63	-	4400.344	3.79
00rc106	4255.16	715.03	4255.16	-	4364.01	4.02
00rc107	4246.97	1800.02*	4254.89	0.1869	4360.423	3.79
00rc108	4246.97	1800.05*	4253.99	0.1650	4399.379	3.62
00rc201	4290.38	6.43	4290.38	-	4368.33	4.01
00rc202	4253.63	390.35	4258.69	0.1188	4363.545	4.43
00rc203	4246.97	886.28	4257.28	0.2421	4362.83	4.08
00rc204	4246.97	869.55	4249.88	0.0684	4363.956	3.76
00rc205	4253.63	366.81	4358.89	2.4148	4364.031	4.11
00rc206	4255.44	48.43	4355.85	2.3051	4402.851	3.35
00rc207	4246.97	1800.02*	4249.71	0.0644	4357.168	4.24
00rc208	4246.97	1804.47*	4255.14	0.1920	4361.284	4.30
Avg.		1196.91		-0.9071		6.03

Table 9: Comparison of F2 and ALNS

20 customers No:	F2		ALNS (Best over 10 reps.)		ALNS (Avg. over 10 reps.)	
	UB	CPU (sec.)	UB	GAP	UB	CPU (sec.)
01c101	1695.93	63.22	1695.93	-	1696.514	4.43
01c102	1693.40	1800.24*	1693.40	-	1694.888	5.33
01c103	1824.41	1804.50*	1688.98	-8.01	1692.375	5.92
01c104	1890.49	1805.05*	1689.38	-11.9	1691.023	5.38
01c105	1689.33	1142.49	1689.33	-	1691.016	5.31
01c106	1695.93	213.79	1695.93	-	1697.751	4.68
01c107	1689.33	353.35	1689.33	-	1691.081	4.58
01c108	1689.33	1800.10*	1689.33	-	1692.017	5.63
01c109	1690.23	1803.10*	1688.5	-0.10	1689.633	5.55
01c201	1729.49	5.59	1729.49	-	1729.49	4.62
01c202	1729.49	1800.26*	1729.49	-	1729.756	5.59
01c203	1730.16	1800.15*	1729.49	-0.03	1730.406	5.39
01c204	1730.89	1800.05*	1727.80	-0.17	1729.246	6.22
01c205	1729.49	36.04	1729.49	-	1730.171	5.30
01c206	1729.49	1800.05*	1729.49	-	1730.65	4.64
01c207	1727.80	922.98	1727.80	-	1728.008	5.29
01c208	1729.88	1800.18*	1729.49	-0.02	1730.129	4.99
01r101	2119.33	1.11	3016.62	29.74	3016.651	7.91
01r102	2043.84	1819.93*	2634.91	22.43	2634.91	5.81
01r103	1700.13	1809.35*	1685.85	-0.84	1688.21	4.85
01r104	1801.20	1814.11*	1428.55	-26.08	1459.4	5.46
01r105	1760.77	1800.10*	1860.77	5.37	1912.941	10.54
01r106	1799.53	1805.19*	1784.65	-0.83	1784.65	8.33
01r107	1601.63	1803.20*	1453.34	-10.20	1483.382	6.39
01r108	1690.20	1804.30*	1409.32	-19.93	1411.92	6.96
01r109	1703.37	1817.28*	1776.87	4.13	1776.87	8.63
01r110	1677.51	1810.29*	1662.25	-0.91	1669.558	6.37
01r111	1585.08	1812.88*	1475.22	-7.44	1581.594	4.90
01r112	1417.74	1800.05*	1413.67	-0.28	1418.558	6.82
01r201	1456.40	991.00	1493.48	2.48	1504.113	5.39
01r202	1635.15	1803.94*	1430.37	-14.31	1437.463	6.01

Table 9: Comparison of F2 and ALNS (cont.)

20 customers	F2		ALNS (Best over 10 reps.)		ALNS (Avg. over 10 reps.)	
No:	UB	CPU (sec.)	UB	GAP	UB	CPU (sec.)
01r203	1455.95	1812.41*	1414.03	-2.96	1432.358	6.00
01r204	1376.46	1803.44*	1383.50	0.50	1400.609	5.82
01r205	1607.48	1807.06*	1434.40	-12.06	1442.765	6.04
01r206	1557.81	1807.48*	1412.06	-10.32	1416.223	7.38
01r207	1659.93	1805.61*	1412.06	-17.55	1417.258	6.41
01r208	1376.46	1802.94*	1383.50	0.50	1392.657	7.01
01r209	1421.37	1804.36*	1422.62	0.08	1426.418	6.46
01r210	1431.83	1809.26*	1423.09	-0.61	1427.556	6.49
01r211	1384.84	1804.80*	1370.91	-1.01	1377.767	6.75
01rc101	2327.57	809.21	2386.70	2.47	2390.769	4.88
01rc102	2316.38	1801.97*	2378.57	2.61	2379.216	5.46
01rc103	2304.34	1820.46*	2376.27	3.02	2377.4	2.62
01rc104	2298.16	1800.18*	2369.33	3.00	2370.007	2.80
01rc105	2322.61	1802.75*	2383.09	2.53	2383.783	2.75
01rc106	2310.46	1800.07*	2377.11	2.80	2377.253	2.53
01rc107	2307.64	1805.42*	2364.39	2.40	2364.72	2.70
01rc108	2298.16	1802.80*	2364.39	2.80	2365.545	2.43
01rc201	2321.90	1803.75*	2383.93	2.60	2385.75	2.65
01rc202	2311.33	1802.81*	2378.50	2.82	2380.791	2.75
01rc203	2311.34	1816.73*	2375.39	2.69	2376.956	2.95
01rc204	2309.58	1824.27*	2369.33	2.52	2371.876	3.16
01rc205	2313.45	1800.46*	2378.50	2.73	2381.313	2.74
01rc206	2308.77	1802.88*	2377.11	2.87	2377.856	2.57
01rc207	2302.21	1802.10*	2364.94	2.65	2366.77	2.47
01rc208	2300.09	1803.67*	2364.39	2.71	2364.802	2.68
Avg.		1564.29		-0.66		5.17

To see how ALNS works on larger instances, 50 and 100-request instances were used. However, F2 formulation could not obtain the optimal solutions for these larger instances. Hence, we employed a simple insertion heuristic, namely *regret-2*, and compare its results with ALNS. The results are reported in the following four tables.

The solutions for the 50-request instances are shown in Tables 10 and 11. In this case, “GAP1” columns indicate the percentage gaps between *regret-2* solution and the best ALNS solution, and “GAP2” columns indicate the percentage gaps between the average ALNS solution and the best ALNS solution. If GAP2 approaches zero, we can say that the algorithm is stable in terms of solution quality among its different replications. When we compare *regret-2* and the best ALNS solutions, huge differences exist in almost all instances, as expected. For low-variance instances the average gap is %12.97, and for high-variance the average gap is %18.08.

As *regret-2* is a simple insertion heuristic, all instances can be solved less than 1 second. Our heuristic also runs under a minute for the 50-request instances, which is quite fast.

The percentage gaps between the best and the average solutions of the ALNS are very small. In almost all instances, the gap is less than %2, and on the average it is %0.75 for low-variance instances and %1.20 for high-variance instances. This observation may indicate that the algorithm converges to a small range of solutions, which may be close to the optimal solution. Another important observation is on computation times. It seems that the computation time of ALNS does not increase significantly when the problem size is increased. This fact is interesting, as it contradicts a common disadvantage of metaheuristic algorithms.

Table 10: Results of ALNS on 50 Requests with Low Variance

50 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
	No:	UB	GAP1	UB	CPU (sec.)	UB
00c101	6434.60	11.47	5719.011	28.66	5697.03	0.38
00c102	6539.85	12.64	5730.656	33.12	5713.53	0.29
00c103	6631.71	13.84	5724.062	33.65	5714.10	0.17
00c104	6614.71	13.71	5719.818	37.47	5708.26	0.20
00c105	6373.73	10.56	5721.86	30.78	5700.81	0.36
00c106	6414.79	11.12	5722.097	31.15	5701.89	0.35
00c107	6449.23	11.60	5723.56	30.17	5701.73	0.38
00c108	6461.21	11.21	5763.817	23.70	5737.16	0.46
00c109	6685.58	14.26	5746.847	41.93	5732.27	0.25
00c201	6428.64	10.02	5794.722	35.81	5785.02	0.16
00c202	6574.89	11.74	5813.425	37.22	5803.36	0.17
00c203	6645.11	12.69	5813.573	43.88	5802.30	0.19
00c204	6830.00	15.05	5813.382	43.47	5802.75	0.18
00c205	6680.02	12.99	5826.144	28.93	5812.30	0.23
00c206	6755.78	14.00	5827.175	35.55	5810.44	0.28
00c207	6818.60	14.88	5811.685	39.53	5804.17	0.12
00c208	6759.01	14.08	5820.458	31.40	5807.83	0.21
00r101	8930.83	18.82	7265.394	64.74	7250.70	0.20
00r102	7925.30	17.01	6580.466	59.94	6577.61	0.04
00r103	5613.43	3.57	5460.041	15.22	5413.39	0.85
00r104	7023.56	24.44	5331.922	64.03	5307.18	0.46
00r105	7303.84	16.94	6092.499	58.17	6066.71	0.42
00r106	6246.13	13.15	5555.515	29.69	5425.31	2.34
00r107	6300.55	14.56	5398.414	30.31	5383.31	0.27
00r108	6404.72	17.01	5335.11	37.30	5315.85	0.36
00r109	6971.22	21.66	5735.087	43.52	5461.61	4.76
00r110	6294.81	14.23	5459.166	35.60	5399.59	1.09
00r111	6355.77	15.37	5388.535	39.94	5379.52	0.16
00r112	6362.99	16.31	5341.058	42.95	5325.58	0.28
00r201	5838.01	7.49	5415.352	18.01	5400.87	0.26
00r202	5868.45	9.27	5347.738	21.43	5324.55	0.43
00r203	5833.4	9.30	5307.229	20.67	5290.91	0.30
00r204	5919.02	10.91	5287.19	21.46	5273.61	0.25
00r205	5792.54	7.83	5354.612	22.00	5339.31	0.28
00r206	5929.60	10.72	5315.664	23.36	5294.45	0.39
00r207	5992.66	11.81	5300.895	21.89	5285.49	0.29
00r208	5843.4	9.75	5284.186	22.81	5274.18	0.18
00r209	5964.5	11.12	5314.566	17.75	5301.57	0.24
00r210	5852.74	9.16	5325.545	18.04	5317.21	0.15
00r211	5770.35	8.66	5278.649	18.73	5270.79	0.14
00rc101	7931.63	9.28	7271.59	25.43	7195.7	1.04
00rc102	8086.03	16.48	6983.331	30.69	6754.19	3.28
00rc103	7812.54	8.02	7235.913	12.03	7186.75	0.67

50 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
	No:	UB	GAP1	UB	CPU (sec.)	UB
00rc104	8037.27	11.49	7160.389	13.01	7114.1	0.64
00rc105	8470.43	18.90	7309.301	19.99	6869.89	6.01
00rc106	7613.33	6.37	7169.892	12.06	7129.05	0.56
00rc107	8404.78	21.37	6781.113	31.28	6608.89	2.53
00rc108	8634.72	23.66	6799.372	29.07	6592.25	3.04
00rc201	8094.32	10.61	7276.26	11.72	7236.08	0.55
00rc202	7752.77	7.55	7231.218	11.97	7167.71	0.87
00rc203	8116.39	11.92	7193.65	13.04	7149.35	0.61
00rc204	8382.34	15.03	7179.754	12.48	7123.21	0.78
00rc205	8058.19	11.04	7233.372	12.24	7169.18	0.88
00rc206	8264.02	13.56	7202.625	14.18	7143.47	0.82
00rc207	8307.26	14.09	7214.532	12.02	7137.31	1.07
00rc208	8122.00	12.36	7160.395	13.17	7118.62	0.58
Avg.		12.97		28.72		0.75

50 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
	No:	UB	GAP1	UB	CPU (sec.)	UB
01c101	4842.96	5.39	4583.26	17.53	4582.22	0.02
01c102	4853.58	5.65	4583.07	17.08	4579.39	0.08
01c103	5003.62	8.49	4588.54	19.96	4579.08	0.20
01c104	5162.17	11.30	4582.02	30.14	4579.22	0.06
01c105	4907.59	6.63	4584.06	16.88	4582.22	0.04
01c106	4849.43	5.52	4583.75	16.83	4582.22	0.03
01c107	5342.71	14.24	4583.80	32.34	4582.24	0.03
01c108	4898.51	6.38	4590.18	20.73	4586.06	0.08
01c109	5026.06	8.86	4591.96	24.06	4580.97	0.23
01c201	4791.95	2.79	4664.18	19.07	4658.37	0.12
01c202	4970.55	6.24	4669.16	22.72	4660.48	0.18
01c203	5021.92	7.27	4670.18	23.49	4657.30	0.27
01c204	5269.84	11.64	4664.87	29.83	4656.63	0.17
01c205	5073.71	8.16	4672.28	21.01	4659.92	0.26
01c206	5247.54	11.26	4667.01	23.37	4656.68	0.22
01c207	5314.90	12.43	4662.50	24.16	4654.69	0.16
01c208	5115.51	9.04	4669.59	23.87	4653.31	0.34
01r101	8257.10	17.64	6941.26	73.61	6800.70	2.02
01r102	6896.71	8.31	6324.19	47.99	6323.69	0.00
01r103	6040.24	10.53	5522.86	30.12	5404.74	2.13

Table 11: Results of ALNS on 50 Requests with High Variance (cont.)						
50 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
01r104	6767.22	37.25	4262.98	123.18	4246.60	0.38
01r105	6918.03	12.97	6055.12	51.98	6020.99	0.56
01r106	6068.38	18.14	5069.51	49.96	4968.10	2.00
01r107	6136.94	28.14	4757.79	73.40	4410.53	7.29
01r108	6190.92	31.58	4241.57	104.12	4236.09	0.12
01r109	6742.04	19.91	5423.29	57.28	5399.92	0.43
01r110	6072.07	19.70	5132.39	50.73	4876.18	4.99
01r111	6097.45	21.08	4830.09	71.90	4812.63	0.36
01r112	6199.35	30.73	4649.37	84.24	4294.60	7.63
01r201	5139.69	23.03	3992.72	59.44	3956.41	0.90
01r202	5020.55	23.82	3858.81	74.71	3825.11	0.87
01r203	5052.90	25.71	3771.27	103.50	3754.2	0.45
01r204	4876.69	24.32	3708.59	113.25	3690.94	0.47
01r205	5192.65	25.73	3880.94	63.86	3856.89	0.61
01r206	5115.31	26.30	3792.03	94.63	3770.42	0.56
01r207	5101.24	26.93	3756.69	84.19	3727.88	0.76
01r208	4892.79	24.26	3714.35	104.44	3705.88	0.22
01r209	5141.94	26.71	3807.65	72.88	3768.56	1.02
01r210	5008.91	24.55	3808.28	84.2	3779.71	0.75
01r211	5075.42	27.14	3734.90	84.42	3698.37	0.97
01rc101	7886.91	29.33	6065.16	72.4	5574.03	8.09
01rc102	6844.61	19.48	5679.88	34.52	5511.35	2.96
01rc103	6416.89	14.46	5511.06	28.19	5489.48	0.39
01rc104	7659.19	29.18	5499.77	57.7	5424.33	1.37
01rc105	7346.39	24.91	5807.47	47.96	5516.43	5.01
01rc106	6797.11	19.13	5709.92	35.74	5497.37	3.72
01rc107	8180.97	33.27	5574.60	79.07	5459.9	2.05
01rc108	8091.53	33.00	5445.36	78.23	5421.85	0.43
01rc201	6659.31	16.79	5577.40	14.33	5541.55	0.64
01rc202	6342.57	13.70	5532.24	18.95	5474.03	1.05
01rc203	6553.13	16.25	5522.72	16.96	5488.43	0.62
01rc204	6562.48	17.07	5470.75	21.75	5442.37	0.51
01rc205	6829.32	19.33	5546.64	16.85	5509.62	0.66
01rc206	6660.89	17.36	5536.27	16.45	5504.88	0.56
01rc207	6726.52	18.51	5525.37	15.28	5481.73	0.78
01rc208	6398.23	15.14	5474.03	16.65	5430.01	0.80
Avg.		18.08		48.43		1.20

Similar comments also apply to the 100-request instances, whose results are reported in Tables 12 and 13. ALNS finds much better solutions than *regret-2* (gaps are %12.59 and %16.02 on the average for low and high variance instances,

respectively). The percentage gap between the average ALNS solution and the best ALNS solution is reduced for both low and high variance settings. The increase in computation times is not significant. All instances are solved with ALNS less than 230 and 365 seconds for low and high variance instances, respectively.

Table 12: Results of ALNS on 100 Requests with Low Variance

100 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
	No:	UB	GAP1	UB	CPU (sec.)	UB
00c101	13177.28	4.40	12667.45	76.92	12597.86	0.54
00c102	13422.97	5.64	12690.70	94.08	12666.99	0.18
00c103	13754.55	10.94	12324.88	91.26	12250.32	0.60
00c104	13883.55	11.50	12338.89	113.46	12288.19	0.41
00c105	13253.51	7.41	12390.14	66.20	12272.47	0.94
00c106	13464.08	8.50	12420.57	100.02	12320.67	0.80
00c107	13256.35	7.64	12276.67	96.34	12243.95	0.26
00c108	13419.46	8.17	12436.08	63.94	12323.75	0.90
00c109	13556.86	9.24	12358.49	102.68	12304.33	0.43
00c201	12925.79	5.28	12300.69	63.62	12244.26	0.45
00c202	13534.78	9.02	12351.90	118.53	12315.08	0.29
00c203	13807.63	10.48	12408.70	86.35	12361.79	0.37
00c204	13979.95	11.63	12410.44	81.27	12354.56	0.45
00c205	13323.28	7.10	12487.22	95.58	12377.96	0.87
00c206	13754.10	9.49	12545.95	68.15	12449.57	0.76
00c207	13862.47	10.73	12433.39	83.08	12375.9	0.46
00c208	13813.44	10.33	12644.34	70.99	12387.05	2.03
00r101	14149.43	13.32	12680.02	146.90	12266.11	3.26
00r102	12586.33	11.01	11661.15	99.90	11201.58	3.94
00r103	11576.37	12.44	10173.04	175.25	10137.02	0.35
00r104	11859.84	16.15	9981.63	163.70	9944.57	0.37
00r105	12019.11	10.57	10907.48	95.84	10748.86	1.45
00r106	11476.47	11.05	10283.45	121.54	10208.37	0.73
00r107	11646.74	13.29	10190.84	108.49	10098.95	0.90
00r108	11835.42	16.12	9977.72	117.99	9928.47	0.49
00r109	11772.80	13.86	10402.92	102.43	10141.87	2.50
00r110	12345.62	18.35	10184.44	229.84	10080.99	1.01
00r111	11631.02	13.72	10094.93	149.00	10035.34	0.59
00r112	11812.49	15.29	10101.86	102.41	10007.12	0.93
00r201	11635.43	13.10	10201.99	134.85	10111.66	0.88
00r202	11840.55	15.36	10140.67	152.55	10022.59	1.16
00r203	12005.36	17.00	10000.63	194.03	9964.84	0.35
00r204	11928.36	16.94	9935.28	181.23	9908.85	0.26
00r205	12023.33	16.78	10060.93	165.59	10006.47	0.54

Table 12: Results of ALNS on 100 Requests with Low Variance (cont.)						
100 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
No:	UB	GAP1	UB	CPU (sec.)	UB	GAP2
00r206	11846.97	15.64	10005.66	134.85	9994.91	0.10
00r207	11971.59	16.88	9972.79	221.36	9951.58	0.21
00r208	11935.48	17.33	9909.58	169.6	9867.45	0.42
00r209	11953.33	16.90	9977.06	179.33	9933.82	0.43
00r210	11928.53	16.45	10002.02	179.25	9966.95	0.35
00r211	11986.04	17.38	9934.17	144.34	9903.15	0.31
00rc101	14082.05	6.22	13424.33	41.94	13206.99	1.61
00rc102	14277.91	11.20	12915.64	83.88	12679.18	1.83
00rc103	14523.35	13.60	12642.33	116.94	12548.23	0.74
00rc104	14594.20	14.07	12588.05	187.24	12541.85	0.36
00rc105	14241.63	10.77	12920.16	89.27	12708.57	1.63
00rc106	14581.25	13.10	12829.00	134.63	12671.59	1.22
00rc107	14621.59	13.17	12732.67	130.19	12696.86	0.28
00rc108	14879.51	15.03	12739.11	113.75	12643.54	0.75
00rc201	14412.74	12.39	12672.03	197.35	12627.98	0.34
00rc202	14613.16	13.83	12623.70	206.51	12593.24	0.24
00rc203	14589.58	13.81	12608.76	182.02	12575.97	0.26
00rc204	14834.01	15.34	12587.12	173.75	12558.63	0.22
00rc205	14709.22	14.08	12664.26	191.21	12639.06	0.19
00rc206	14825.00	15.16	12653.04	166.97	12577.8	0.59
00rc207	14911.05	15.50	12645.05	126.14	12600.97	0.34
00rc208	14831.40	15.87	12529.27	172.20	12479.09	0.40
Avg.		12.59		129.58		0.77

Table 13: Results of ALNS on 100 Requests with High Variance

100 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
	No:	UB	GAP1	UB	CPU (sec.)	UB
01c101	10659.24	12.68	9412.22	163.37	9307.87	1.10
01c102	10462.86	11.60	9397.88	69.52	9249.86	1.57
01c103	10668.00	12.84	9427.05	88.39	9298.81	1.36
01c104	10754.61	13.10	9438.794	105.09	9346.36	0.97
01c105	10598.29	12.19	9384.58	103.79	9307.3	0.82
01c106	10620.25	10.77	9600.332	77.26	9477.33	1.28
01c107	10650.51	13.01	9375.26	95.36	9265.69	1.16
01c108	10586.15	9.68	9619.046	87.61	9561.51	0.59
01c109	10519.74	10.38	9589.324	90.73	9428.34	1.67
01c201	9750.37	4.84	9295.662	93.59	9278.85	0.18
01c202	10264.75	8.44	9420.092	97.15	9398.5	0.22
01c203	10741.56	12.40	9486.12	89.25	9410.34	0.79
01c204	10896.46	13.47	9456.114	157.39	9429.34	0.28
01c205	10323.55	8.29	9492.582	110.14	9468.41	0.25
01c206	10679.85	11.63	9486.678	129.64	9438.1	0.51
01c207	10727.77	12.10	9494.478	95.3	9430.21	0.67
01c208	10828.00	4.98	10357.47	91.59	10288.81	0.66
01r101	12007.19	11.02	10906.42	176.2	10684.55	2.03
01r102	11963.40	24.99	9226.05	165.78	8974.66	2.72
01r103	10821.24	24.79	8165.172	225.98	8139.26	0.31
01r104	10620.50	3.13	10357.47	91.59	10288.81	0.66
01r105	10964.85	18.49	9207.222	162.31	8937.45	2.93
01r106	10334.49	19.99	8303.926	194.09	8268.99	0.42
01r107	10440.83	25.99	7830.148	364.18	7727.96	1.30
01r108	10986.84	18.39	9125.168	325.76	8966.65	1.73
01r109	11676.14	27.40	8829.40	296.12	8477.97	3.98
01r110	11322.58	27.12	8387.13	174.55	8252.03	1.61
01r111	9890.80	17.35	8198.88	331.59	8175.56	0.28
01r112	10983.52	28.71	7859.59	97.30	7831.24	0.36
01r201	8912.59	13.29	7747.72	149.64	7728.81	0.24
01r202	9264.37	17.43	7653.68	173.03	7649.78	0.05
01r203	9177.77	17.51	7584.53	212.89	7571.32	0.17
01r204	8937.33	13.62	7733.93	142.52	7720.9	0.16
01r205	9198.98	16.77	7683.08	93.79	7657.22	0.33
01r206	9245.23	17.75	7622.45	175.28	7604.36	0.23
01r207	9003.50	16.18	7563.00	185.9	7547.59	0.20
01r208	9118.21	16.26	7672.04	141.78	7635.65	0.47
01r209	9094.57	15.78	7685.03	156.25	7659.94	0.32
01r210	9106.96	16.81	7584.57	158.6	7576.87	0.10
01r211	9130.83	16.23	7653.68	156.34	7649.78	2.32
01rc101	12007.19	12.16	10683.17	115.02	10547.16	1.27
01rc102	12233.53	19.61	9958.22	208.68	9835.72	1.23
01rc103	12150.88	20.75	9734.27	363.39	9630.61	1.06

100 customers	Regret-2		ALNS (Avg. over 10 reps.)		ALNS (Best over 10 reps.)	
	UB	GAP1	UB	CPU (sec.)	UB	GAP2
No:						
01rc104	13413.48	20.83	11076.55	177.88	10620.18	4.12
01rc105	13031.41	18.34	10876.27	236.42	10642.73	2.14
01rc106	13148.58	21.62	10676.99	231.88	10306.35	3.47
01rc107	13185.78	22.32	10385.62	351.47	10242.81	1.37
01rc108	13916.76	29.71	9807.69	117.03	9782.41	0.25
01rc201	11371.44	14.73	9726.73	117.17	9697.21	0.30
01rc202	11370.28	15.73	9637.34	124.29	9581.87	0.57
01rc203	11402.06	16.48	9571.52	130.13	9523.53	0.50
01rc204	11665.50	16.62	9784.89	96.29	9727.71	0.58
01rc205	11323.42	14.60	9721.86	103.62	9671.28	0.52
01rc206	11482.61	15.98	9680.64	110.89	9647.89	0.33
01rc207	11302.32	16.33	9569.25	115.16	9456.90	1.17
01rc208	11306.92	13.97	9784.89	172.2	9727.71	0.50
Avg.		16.02		156.75		1.00

5.3 Discussion

Solutions by the mathematical models and the ALNS are compared in this section. The experimental results show that, the flow type variables tighten the formulation significantly, which leads to much better lower bounds. Also, the covering type inequality proposed by Yaman (2006) strengthens the formulations remarkably. Based on small-size instance results, the demand flow formulation (F2) is superior to the other ones with regard to its lower bounds and computation time.

The comparison between the best formulation (F2) and ALNS shows that the proposed algorithm finds optimal solutions when the mathematical model can find optimal solutions, and finds better solutions when the model cannot obtain the optimal solutions within the limited time. On the average, ALNS finds better solutions in all test instances in less than 10 seconds when compared with the model.

Expectedly, ALNS finds better solutions than a simple insertion heuristic. When we compare ALNS and *regret-2* heuristic on large instances, the average gap is very large. Two important characteristics of the proposed heuristic are worth underlying. When problem size increases, the computation time of ALNS does not increase significantly. Besides, the solution quality of ALNS is very stable when the gaps between its different replications are considered. All in all, the proposed ALNS

algorithm seems to be very promising for the studied HVRPTWSPD problem. In Table 14, a summary of computational experiments is shown. The rows in the table show the averages over all instances in each problem size, considering low and high variances. “GAP1” indicates the percentage gap of the best result of ALNS and the result of F2, as $GAP1 = \%100 * \frac{UB_{ALNS_{Best}} - UB_{F2}}{UB_{ALNS_{Best}}}$, and “GAP2” is the percentage gap of the best result of ALNS and the result of *regret-2*, as $GAP2 = \%100 * \frac{UB_{ALNS_{Best}} - UB_{regret-2}}{UB_{ALNS_{Best}}}$. On the average, ALNS finds better solutions over all instances in each setting. The percentage gaps between ALNS and *regret-2* are considerably large, the smallest being %12.59. ALNS is extremely fast, especially in large instances, when complexity of the problem is considered.

Table 14: Comparison of Results

Instances (56 inst. in each row)	F2		<i>Regret-2</i>		ALNS
	GAP1 (%)	CPU (sec.)	GAP2 (%)	CPU (sec.)	CPU (sec.)
20 cust. – low variance	0.90	1196.91	17.21	0.003	6.03
20 cust. – high variance	0.66	1564.29	16.56	0.001	5.17
50 cust. – low variance	-	-	12.97	0.22	28.72
50 cust. – high variance	-	-	18.08	0.23	48.43
100 cust. – low variance	-	-	12.59	0.61	129.58
100 cust. – high variance	-	-	16.02	1.63	156.75

6 CONCLUSION

In this thesis, a heterogeneous vehicle routing problem with time windows and simultaneous pick-up and delivery is considered. The problem has been brought to our attention by a software company, to provide a route optimization tool to its customers. For us, the main attractions for studying this problem have been its wide application areas and the possible contribution to the academic literature. Therefore, our study hopefully brings contributions to both practitioners and the VRP literature.

We reviewed the literature on similar studies. Three formulations were proposed for the problem, as well as a heuristic to solve large real life problem instances that are encountered by the customers of the aforementioned software company. A covering type valid inequality proposed by Yaman (2006) was adapted to our problem, which strengthened our formulations. The proposed heuristic algorithm was inspired by the ALNS algorithm proposed by Ropke and Pisinger (2006). New sets of benchmark instances were generated to compare the results of the mathematical models and the proposed ALNS algorithm, as well as comparison to a well-known simple heuristic.

Computational experiments showed that the mathematical model based on demand flow variables gives the best results in terms of both the lower bounds and the computation times for small instances. When we compared the demand flow model and our ALNS, our algorithm gives near optimal solutions extremely fast. For larger instances, the proposed algorithm is stable in terms of solution quality, and is very robust in terms of computation time. Unlike other metaheuristic algorithms, the marginal increase in the computation times of our ALNS decreases as problem size increases. When the evident complexity of the studied problem is considered, we can safely state that the developed algorithm gives outstanding results.

Currently, our algorithm is in the implementation stage by the company, and will be used for route optimization in the very near future. In this respect, we are glad to have contributed to the practitioners by supplying a tool that can be used in their everyday logistics operations. The solution qualities obtained by the algorithm over the generated instances in very small computation times also indicate a significant contribution to the VRP literature for this hard problem. The implementation will also

provide us with the opportunity to observe the performance of the proposed heuristic on real life instances.

As it was stated before, the problem has many application areas. However, one natural extension of this study can include stochastic aspects. For example, the times between nodes are assumed to be known and fixed in our study, as the distances between nodes and the velocities of the vehicles are known. However, this parameter is rarely deterministic in real world. The stochastic extension of the problem may lead to more reliable results, however it will be much harder to solve.

Another realistic extension may contain multi-depots. This feature is important especially when dealing with pick-up demands, when the returned products are delivered to another depot, which may be the recycling plant. Therefore, the multi-depot extension may be worth studying.

Lastly, in real life practices, it is very common that some vehicles cannot visit some customers due to some physical limitations. Therefore, it will be better if the algorithm is compatible with site-dependent VRP. In our algorithm, it is very easy to add site dependency. Because, only the feasible insertions are accepted during search. Therefore, if a check is made whether vehicle can visit a customer or not, the algorithm will also be compatible with site-dependent VRP.

REFERENCES

- Azi, N., Gendreau, M., & Potvin, J. Y.,** 2014, An adaptive large neighborhood search for a vehicle routing problem with multiple routes. *Computers & Operations Research*, 41, 167-173
- Baldacci, R., Battarra, M., & Vigo, D.,** 2008, Routing a Heterogeneous Fleet of Vehicles. *The Vehicle Routing Problem: Latest Advances and New Challenges*, 3-27.
- Berbeglia, G., Cordeau, J. F., Gribkovskaia, I., & Laporte, G.,** 2007, Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1), 1-31.
- Cordeau, J. F., Desaulniers, G., Desrosiers, J., Solomon, M. M., & Soumis, F.** 2001, VRP with Time Windows. In *The vehicle routing problem*. Society for Industrial and Applied Mathematics, 157-193.
- Cordeau, J. F., Gendreau, M., Hertz, A., Laporte, G., & Sormany, J. S.,** 2005, New Heuristics for the Vehicle Routing Problem. *Logistics Systems: Design and Optimization*, 279-297.
- Crainic, T. G., Perboli, G., Rei, W., & Tadei, R.,** 2011, Efficient lower bounds and heuristics for the variable cost and size bin packing problem. *Computers & Operations Research*, 38(11), 1474-1482.
- Dantzig, G. B., & Ramser, J. H.,** 1959, The truck dispatching problem. *Management Science*, 6(1), 80-91.
- Demir, E., Bektaş, T., & Laporte, G.,** 2012, An adaptive large neighborhood search heuristic for the pollution-routing problem. *European Journal of Operational Research*, 223(2), 346-359.
- Dethloff, J.,** 2001, Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up. *OR-Spektrum*, 23(1), 79-96.

Hemmelmayr, V. C., Cordeau, J. F., & Crainic, T. G., 2012, An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers & operations research*, 39(12), 3215-3228.

Kallehauge, B., Larsen, J., Madsen, O. B., & Solomon, M. M., 2005, Vehicle Routing Problem with Time Windows. *Column Generation*, 67-98.

Kallehauge, B., 2008, Formulations and exact algorithms for the vehicle routing problem with time windows. *Computers & Operations Research*, 35(7), 2307-2330.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., 1983, Optimization by simulated annealing. *Science* 220 (4598), 671-680.

Laporte, G., 1992, The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3), 345-358.

Laporte, G., Gendreau, M., Potvin, J. Y., & Semet, F., 2000, Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research*, 7(4-5), 285-300.

Li, H., & Lim, A., 2003, A metaheuristic for the pickup and delivery problem with time windows. *International Journal on Artificial Intelligence Tools*, 12(02), 173-186.

Lu, Q., & Dessouky, M. M., 2006, A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows. *European Journal of Operational Research*, 175(2), 672-687.

Luo, Z., Qin, H., Zhang, D., & Lim, A., 2016, Adaptive large neighborhood search heuristics for the vehicle routing problem with stochastic demands and weight-related cost. *Transportation Research Part E: Logistics and Transportation Review*, 85, 69-89

Miller, C. E., Tucker, A. W., & Zemlin, R. A., 1960, Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4), 326-329.

Özfirat, P. M., 2008, Exact and Heuristic Algorithms for the Variant of the Vehicle Routing Problem (Doctoral thesis). Dokuz Eylul University, Izmir, Turkey.

Parragh, S. N., Doerner, K. F., & Hartl, R. F., 2008, A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, 58(1), 21-51.

Pisinger, D., & Ropke, S., 2007, A general heuristic for vehicle routing problems. *Computers & operations research*, 34(8), 2403-2435.

Pisinger, D., & Ropke, S., 2010, Large neighborhood search. In *Handbook of metaheuristics* (pp. 399-419). Springer US.

Potvin, J. Y., & Rousseau, J. M., 1993, A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3), 331-340.

Ropke, S., & Pisinger, D., 2006, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4), 455-472.

Ribeiro, G. M., & Laporte, G., 2012, An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 39(3), 728-735.

Shaw, P., 1997, A new local search algorithm providing high quality solutions to vehicle routing problems. APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK.

Shaw, P., 1998, Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming—CP98*. Springer Berlin Heidelberg.

Solomon, M. M., 1987, Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research* 254–265.

Sudalaimuthu, S., & RAJ, S. A., 2009, Logistics Management for International Business: Text and Cases. PHI Learning Pvt. Ltd.

Taşar, B., 2016, A Multi-Compartment Vehicle Routing Problem for Incompatible Products (Master's thesis). Yasar University, Izmir, Turkey.

Toth, P., & Vigo, D., 2001, The vehicle routing problem. Society for Industrial and Applied Mathematics.

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C., 2013, Heuristics for multi-attribute vehicle routing problems: a survey and synthesis. European Journal of Operational Research, 231(1), 1-21.

Yaman, H., 2006, Formulations and valid inequalities for the heterogeneous vehicle routing problem. Mathematical Programming, 106(2), 365-390.

CURRICULUM VITAE

Gökberk Özşakallı was born in İzmir. He received his BS degree in Industrial Engineering from Yaşar University. He worked in a TUBITAK project as a research assistant. His function was developing a solution approach for berth allocation and quay crane assignment problem in stochastic environment. He was an intern in Basque Center for Applied Mathematics. His job definition was applying solution algorithms for face-recognition problem. He is currently working in UNIVERA A.Ş., which is a software company, to develop algorithms for the vehicle routing problem and sales forecasting. He continues his MSc studies in Industrial Engineering at Yaşar University.