HEURISTIC APPROACHES FOR MULTI DEPOT VEHICLE ROUTING PROBLEMS WITH HETEROGENEOUS VEHICLE FLEET

FATİH KOCATÜRK

Ph.D. Thesis

Graduate School

Izmir University of Economics

İzmir

2022

HEURISTIC APPROACHES FOR MULTI DEPOT VEHICLE ROUTING PROBLEMS WITH HETEROGENEOUS VEHICLE FLEET

FATİH KOCATÜRK

A Thesis Submitted to

The Graduate School of Izmir University of Economics

Ph.D. Program in Applied Mathematics and Statistics

İzmir

2022

ABSTRACT

HEURISTIC APPROACHES FOR MULTI DEPOT VEHICLE ROUTING PROBLEMS WITH HETEROGENEOUS VEHICLE FLEET

Kocatürk, Fatih

Ph.D. Program in Applied Mathematics and Statistics

Advisor: Prof. Dr. Gözde Yazgı TÜTÜNCÜ

March, 2022

In this thesis, we investigated the Multi-Depot Heterogeneous Vehicle Routing Problems (VRP) with Backhauls. Though the problem is a generalisation of three existing routing problems: Multi-depot VRP, the heterogeneous vehicle fleet problem and the routing problem with backhauls, this is the first time this combined routing problem was investigated. A mathematical formulation was first presented followed by some tightening. A powerful and novel unified hybridisation of Variable Neighbourhood Search (VNS) with the Greedy Randomized Adaptive Memory Programming Search (GRAMPS), VNS-GRAMPS for short, was proposed. As there are no problem instances available for bench-marking, we generated data sets by combining those from existing VRPs. The proposed meta-heuristic obtained a number of optimal solutions

for small instances and yields about 13% gap from the lower bounds compared to nearly 40% and 20% average gap values for our CPLEX implementation and the VNS without hybridisation, respectively. Moreover, a Decision Support System (DSS) was developed that can solve VRPs with single-depot, multi-depot, heterogeneous vehicle fleet and backhaul features with VNS-GRAMPS meta-heuristic. The developed DSS is a visual interactive solution tool called as ADVISER2. The user can save the problems to be solved and the solutions obtained into the database of DSS, and print the report of the saved solution. After selecting the problem to be solved, the user can solve the problem with VNS-GRAMPS and can make interactive changes on the solution. The developed DSS was tested on the problem instances suggested in the literature for 5 different problems, and it was able to find solutions in a short CPU time with a maximum average gap value of 5.87% from the best known solutions, except for the Multi-depot VRP (MVRP) with backhauls problem. Moreover, new best solutions of one problem for fleet size and mix VRP and 3 problems for MVRP were found with VNS-GRAMPS.

Keywords: multi depot vehicle routing problem, heterogeneous vehicle fleet, backhaul, decision support system, VRP, DSS.

ÖZET

ÇOK DEPOLU HETEROJEN ARAÇ FİLOLU ARAÇ ROTALAMA PROBLEMLERİ İÇİN SEZGİSEL YAKLAŞIMLAR

Kocatürk, Fatih

Uygulamalı Matematik ve İstatistik Doktora Programı

Tez Danışmanı: Prof. Dr. Gözde Yazgı TÜTÜNCÜ

Mart, 2022

Bu tezde, Çok-depolu Geri-toplamalı Heterojen ARP (ÇGHARP) araştırılmıştır. Problem, mevcut üç rotalama probleminin (çok-depolu ARP, heterojen araç filolu problem ve geri-toplamalı rotalama problemi) bir genellemesi olmasına rağmen, bu birleştirilmiş rotalama problemi ilk kez araştırılmıştır. Önce matematiksel bir model sunulmuş, ardından bazı kısıtları daraltılmıştır. Değişken Komşuluk Araması (DKA) ile Aç gözlü Rastsal Adaptif Hafıza Programlama Araması (ARAHPA), kısaca DKA-ARAHPA, birleştirilerek güçlü ve özgün bir birleşik meta-sezgisel önerilmiştir. Kıyaslama ve değerlendirme yapabilmek için herhangi bir problem örneği bulunmadığından, mevcut araç rotalama problem örneklerini birleştirerek veri setleri oluşturulmuştur. Önerilen meta-sezgisel, küçük örnekler için bir dizi optimal çözüm

elde edebilmiş ve sırasıyla CPLEX ve temel DKA ile elde edilen yaklaşık %40 ve %20 ortalama aralık değerlerine kıyasla alt sınırlardan yaklaşık %13 aralık değeri elde etmiştir. Tek depo, çok depo, heterojen araç filosu ve geri toplama özelliklerine sahip araç rotalama problemlerini DKA-ARAHPA meta-sezgiseli ile çözebilen bir Karar Destek Sistemi (KDS) geliştirilmiştir. Geliştirilen KDS, ADVISER2 olarak adlandırılan görsel etkileşimli bir çözüm aracıdır. Kullanıcı, çözülmesi gereken problemleri ve elde ettiği çözümleri KDS veritabanına kaydedebilmekte ve kaydedilen çözümün raporunu yazdırabilmektedir. Kullanıcı, çözülecek problemi seçtikten sonra DKA-ARAHPA ile problemi çözebilmekte ve çözüm üzerinde interaktif değişiklikler yapabilmektedir. Geliştirilen KDS, 5 farklı problem için literatürde önerilen problem örnekleri üzerinde test edilmiş ve Geri toplamalı Çok-depolu ARP hariç en iyi bilinen çözümlerden maksimum %5,87 ortalama aralık değeri ile kısa bir CPU süresinde çözüm bulabilmiştir. Ayrıca, DKA-ARAHPA ile Sabit Filolu ve Karma ARP için bir problemin ve çok-depolu ARP için 3 problemin yeni en iyi çözümleri bulunmuştur.

Anahtar Kelimeler: çok depolu araç rotalama problemi, heterojen araç filosu, geri toplama, karar destek sistemi, ARP, KDS.

This thesis work is dedicated to my wife, Ayşenur Kocatürk, who has been a constant source of support and encouragement during the challenges of graduate school and life. I am truly thankful for having you in my life. This work is also dedicated to my wonderful son Egemen Efe for his patience, love and cuteness throughout the entire doctorate program, and to my parents, Mehmet and Raziye Kocatürk, who have always loved me unconditionally and whose good examples have taught me to work hard for the things that I aspire to achieve.

This thesis is dedicated to my adviser, Prof. G. Yazgı Tütüncü and her husband Tolga Osman Aşçı, who guided me in this process, took care of all my troubles patiently and was with me in my private moments. This work is also dedicated to the thesis monitoring committee members, Assoc. Prof. Özgür Özpeynirci, Assoc. Prof. Umay Uzunoğlu Koçer, who kept me on track. I also give special thanks to Assoc. Prof. Özgür Özpeynirci for taking care of all my troubles patiently and for his encouragement to finish my dissertation.

I dedicate this work and give special thanks to my friends at Norm Cıvata R&D Center, especially Dr. Doğuş Zeren and Dr. M. Burak Toparlı, for their encouragement to finish my dissertation.

ACKNOWLEDGEMENT

I would like to express my gratitude to my thesis advisor, Prof. G. Yazgı Tütüncü, for the abilities such as mathematical modeling, project management, article writing she taught me, and the support, motivation and invaluable guidance she gave me, starting from my undergraduate education and throughout my thesis. I would also like to thank my thesis monitoring committee members Assoc. Prof. Özgür Özpeynirci and Assoc. Prof. Umay Uzunoğlu Koçer for their encouraging words, helpful guidance and patiently reading the reports I prepared.

A special thanks go to Emeritus Prof. Said Salhi, Operational Research / Management Science at Kent Business School, for his contributions and invaluable guidance in the editing of our published article.

I would like to express my gratitude to Assoc. Prof. Özgür Özpeynirci for the various academic research skills such as heuristic algorithm coding, article writing that he taught me during the project we worked together. Finally, I would like to thank Tolga Osman Aşçı, who supported my learning for the C# programming language, which I used to develop the decision support system proposed within the scope of the thesis.

I would like to thank my Ph.D. exam jury members, who so generously took time out of their schedules to participate in my thesis defense and make this thesis possible.

I would like to thank İzmir University of Economics for the Doctorate Scholarship and the services it provided for me to do my Ph.D.

I acknowledge the support of the Scientific and Technological Research Council of Turkey (TÜBİTAK), grant number 1001 - 213M438.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Vehicle Routing Problem (VRP) has a wide range of applications in both public and private sector having transportation, logistic and supply chain problems. VRP is firstly introduced by Dantzig and Ramser (1959) and various exact and heuristic solution approaches were proposed in the last 60 years. The classical VRP consists of a set of customers with their respective demand, a depot as the supply center and a fleet of vehicles having the same capacity. The aim of VRP is to find the route combination that gives the minimum travel distance and meets customers' demand with the vehicles in a depot. In real life, VRP and its variants contain additional constraints and challenges beyond the classical VRP such as multiple depots, heterogeneous fleet and pickup customers. It is important to define the characteristics of the problems while working with the extensions of VRP. In real life, identifying the type of the problem is so important for focusing on the future research and saving time. One can reach detailed information about the models and solution methods of VRP and its extensions in Toth and Vigo (2002).

In this chapter, we introduced the VRPs examined in the scope of this thesis and presented the related literature review for these problems. Several single-depot and multi-depot VRPs having heterogeneous vehicle fleet and backhaul properties were studied including Heterogeneous Fixed Fleet VRP (HFFVRP), Fleet Size and Mix VRP with Backhauls (FSMVRPB), Multi-depot VRP (MVRP), MVRP with Backhauls (MVRPB), Multi-depot Heterogeneous VRP (MDHFVRP) and the newly proposed MDHFVRP with Backhauls (MDHFVRPB).

1.1 Overview of Vehicle Routing Problems

In this section, single-depot and multi-depot VRPs examined in this thesis were introduced in the following sub-sections.

1.1.1 The Heterogeneous Fixed Fleet Vehicle Routing Problem (HFFVRP)

By using a homogeneous vehicle fleet, classical VRP aims to dispatch the loads taken from a single depot to customers with the least number of vehicles and travel distance. In this problem, it is assumed that there is an enough amount of load in the depot to meet the demands of the customers and the number of vehicles is unlimited. On the other hand, the HFFVRP proposed by Taillard (1999) includes both a heterogeneous fleet of vehicles with different fixed and variable costs and capacities, and a certain number of vehicles in each type. Using these constraints, HFFVRP aims to find the solution that gives the least cost comprising vehicle fixed cost and travel cost in order to meet customer demands. A route included in a proposed solution for HFFVRP must meet the following constraints:

- Each route starts and ends at the same depot after distributing the demands of the customers,
- Each customer's demand must be distributed with only one vehicle and at a single visit,
- The total demand of the customers on the route cannot be more than the capacity of the vehicle,
- The total travel distance or service time cannot exceed the maximum route distance or duration,
- The number of k-type vehicles used in the solution cannot exceed the total number of vehicles of that type, n_k.

1.1.2 The Fleet Size and Mix Vehicle Routing Problem with Backhauls (FSMVRPB)

The problem consisting of pick-up and delivery customers served by a heterogeneous vehicle fleet from a depot was defined as FSMVRPB. This problem is a combination of Fleet Size and Mix VRP (FSMVRP) introduced by Golden et al. (1984) and VRP with Backhauls (VRPB) defined by Golden et al. (1985). In this problem,

there is a heterogeneous fleet of vehicles that varies according to fixed cost, variable cost or capacity, but it is assumed that there is an unlimited number of each vehicle type in the fleet. In some VRPB problems, it is assumed that there is a fixed number of vehicles in the fleet (Toth and Vigo, 1997). A vehicle that starts the route from a depot to the first delivery customer must return to the same depot at the end of the route. Due to the backhaul nature of the problem, backhaul customers only have pick-up loads and are visited only once by one vehicle. In addition, a vehicle that starts the route from the depot can visit the backhaul customers after visiting all of the delivery customers, and the vehicle is not allowed to visit a delivery customer again after visiting a backhaul customer. The total of the demands of the distribution customers visited by the vehicle along the route and the total of the loads collected from the backhaul customers should not exceed the capacity of the vehicle. Additionally, a route with only delivery customers is allowed, while a route with only backhaul customers is not.

1.1.3 Multi-depot Vehicle Routing Problem (MVRP)

The routing problem, which aims to distribute the demands of the customers with the least travel cost with a homogeneous vehicle fleet from multiple depots, is defined as MVRP. In this problem, a capacity constraint is applied to the vehicles and each vehicle must start the route from a depot and return to the same depot at the end of the route. It is also assumed that there is an unlimited number of vehicles in the depots and there is enough supply to meet the demands of the customers.

1.1.4 Multi-depot Vehicle Routing Problem with Backhauls (MVRPB)

MVRPB includes pick-up customers where products are collected as well as delivery customers. The collected loads of pick-up customers are transported back to the depot. In this problem, the backhaul method defined by Toth and Vigo (1997) was applied. According to this method, pick-up customers on a route can be visited after all delivery customers are visited on the route. In other words, after a vehicle visits the last delivery customer on the route, it can visit the pick-up customers, if any, otherwise

it has to return to the depot where it was originated. In this problem, vehicle capacity constraint is applied and it is assumed that there are unlimited number of vehicles in the depots.

1.1.5 Multi-depot Heterogeneous Vehicle Routing Problem (MDHFVRP)

In most of the transportation problems encountered in real life, it is aimed to distribute customer demands with the least cost with vehicles of different capacities and costs moving from more than one depot. In the VRP literature, the problem in which loads are distributed to customers with a heterogeneous vehicle fleet from multiple depots is defined as MDHFVRP or multi-depot vehicle fleet mix problem. It is an NP-Hard combinatorial optimization problem, commonly faced in real life vehicle routing applications. In MDHFVRP, a capacity constraint is applied to the vehicles and it is assumed that there are unlimited numbers of each vehicle type in each depot. At the end of the route, each vehicle must return to the depot where it started the route. In addition, in some problem instances, the maximum travel distance constraint is applied to the routes.

1.1.6 Multi-Depot Heterogeneous Vehicle Routing Problem with Backhauls (MDHFVRPB)

The problem presented in this section is a commonly faced decision problem in real-life logistic systems which include for example the case of beer or coke distribution companies. Here, full bottles need to be delivered to customers, empty ones to be collected, not necessary an homogeneous vehicle fleet is always used and also not all customers are necessarily served from one depot only. This problem can be considered as the integration of three complex but related routing problems which are commonly studied in the literature. These include the MVRP, the heterogeneous vehicle fleet problem and the routing problem with backhauls. We refer to this integrated routing problem as the MDHFVRPB for short.

In real life, VRP and its variants contain additional constraints and challenges beyond the classical VRP such as multiple depots, heterogeneous fleet and pickup

customers. In recent years, there is an increasingly environmentally-conscious public awareness resulting in more collection of recyclable goods. Moreover, in the current severe economic situation, companies are also paying more attention than ever to financial savings obtained by combining deliveries and pickups. Within reverse logistics for example, efficient solution methods for the VRP with Deliveries and Pickups (VRPDP) contributes considerably to reducing waste in terms of time and energy consumption which systematically leads to a reduction of CO_2 emission and consequently health benefits.

There are different types of VRPDP concerning the order of visiting delivery (linehaul) and pickup (backhaul) customers. In this study, we deal with an extension of VRPB in which pickup customers in a route are served after delivery customers are completed only. This is one of the simplest pick up and delivery problem where no reshuffle is required during the deliveries or collections. This is contrary to the VRP with Mix Deliveries and Pickups (VRPMDP) where deliveries and pickups may occur in any order as long as the maximum vehicle capacity constraint remains satisfied along each arc of the route (Wassan and Nagy, 2014). A real routing case study in China Post of Guangzhou that incorporates fleet heterogeneity, backhaul mixed-loads, and time windows was solved by Wu et al. (2016) using a multi-attribute label-based ant colony system. It was shown that the use of an heterogeneous fleet of vehicles can lower the service cost up to 9.2% than relying on a homogeneous fleet. Most recently, a VRP with two-dimensional loading constraints and mixed linehauls and backhauls were investigated in Pinto et al. (2020). It is interesting to note that in this case, backhaul customers do not need to be postponed in a route when it is possible to pick up items earlier and without rearrangements of the items. Three variants of Variable Neighbourhood Search (VNS) were proposed using 10 neighbourhood structures including new ones that take into account the selection of customers of linehauls and backhauls.

In this thesis, the MDHFVRPB which is a new extension of the VRP was defined. In the classical VRP, there is an homogeneous vehicle fleet (unlimited number of vehicles) and a set of customers with known demands. The objective of the problem is to find the vehicle routes starting and finishing at the same depot, resulting in

the minimum cost while satisfying capacity and travel distance constraints if any. MDHFVRPB is a more complicated version of the classical VRP. This problem is modelled by combining earlier formulations for the FSMVRPB proposed by Salhi et al. (2013) and the MDHFVRP also presented by Salhi and Sari (1997). The properties of the MDHFVRPB are summarized as follows:

- Customers are divided into two groups: delivery (linehaul) and pickup (backhaul) customers.
- There are more than one depot and there is a heterogeneous vehicle fleet (unlimited number of vehicles in each type) which has fixed costs varies according to capacity and variable costs, at each depot.
- Backhaul customers cannot be visited unless all linehaul customers are visited.
- While a route consisting of only backhaul customers is not allowed, a route includes only linehaul customers is allowed.
- Vehicle capacity constraint is applied. Travel distance and the vehicle number constraint for each vehicle type can be added, but these constraints were not used here.

1.2 Literature Review

In this chapter, we reported the literature for the VRPs introduced in the former section.

1.2.1 Literature Review of MVRP

MVRP is more advanced and challenging variation compared to single depot VRP, but many real-life problems can be modelled and analysed as MVRP more conveniently. NP-Hardness of MVRP was proved by Lenstra and Kan (1981) as a combinatorial optimization problem. Exact solution algorithms including symmetric and asymmetric cases were developed firstly in Laporte et al. (1984) and Laporte et al. (1988). In order to solve MVRP, generally two methods are used: Clustering method

in which customers are assigned to depots, and routing method in which the minimum cost route is found in each cluster. The problem can be solved by using these methods in two different orders: Route First Cluster Second (RFCS), Cluster First Route Second (CFRS).

Most of the proposed algorithms for MVRP in the literature were heuristic algorithms. Tillman (1969) proposed the first heuristic that uses Clarke and Wright savings algorithm for MVRP. Wren and Holliday (1972) developed a heuristic that uses sweep algorithm. Gillett and Johnson (1976) grouped the customers around the nearest depot to form disjoint sets and then applied sweep algorithm for each depot to construct the routes. Golden et al. (1977) proposed two heuristic methods for MVRP. Saving algorithm defined by Yellow (1970) was used in the first method and CFRS approach was utilised in the second method to solve large-sized problem instances. Raft (1982) and Ball et al. (1983) used RFCS approach. A modular approach that decomposes the problem into smaller sub problems was also proposed in Raft (1982). First, customers were assigned to vehicle routes, then the routes were assigned to the nearest depot. Chao et al. (1993) defined a multi-phase heuristic. This heuristic firstly assigns customers to the nearest depot, then assigns customers to the routes by using savings algorithm in each depot and lastly improves the solution by changing the routes of customers. Potvin and Rousseau (1993) improved the heuristic proposed by Chao et al. (1993) by adding a few new ideas to assign customers to the depots. Renaud et al. (1996) developed a new heuristic using Tabu Search (TS) algorithm to solve MVRP having route and capacity constraints. Cordeau et al. (1997) also used TS algorithm to solve MVRP. Salhi and Sari (1997) proposed a three-phase heuristic. In the first phase, a feasible initial solution was constructed, the routes in each depot were improved in the second phase and the routes in all depots were improved in the last phase. Thangiah and Salhi (2001) developed genetic clustering heuristic.

The first notable study on the MVRP was by Ho et al. (2008), who proposed two Hybrid Genetic Algorithms (HGA) with different initial solutions for MVRP. The initial solutions of HGA were constructed randomly, combining Clarke and Wright savings algorithm with the nearest neighbour heuristic. Mirabi et al. (2010) proposed three hybrid heuristics to solve MVRP. These heuristics use deterministic, stochastic

and SA improvement methods, respectively. Liu et al. (2010) developed two-phase greedy algorithm minimizing the empty vehicle changes to solve applicable large size MVRP. In the first phase of the proposed algorithm, the initial routes were constructed by using Genetic Algorithm (GA), the initial solution was improved by using local search methods in the second phase. Geetha et al. (2012) applied CFRS method to solve MVRP and used meta-heuristics such as GA, Particle Swarm Optimization (PSO). In this study, Hybrid PSO meta-heuristic was also proposed. A bi-level Voronoi diagram-based meta-heuristic was proposed by Tu et al. (2014) to solve very large-scale real-world MVRPs. In order to improve MVRP solution quality, they extended the one-level Voronoi diagrams to bi-level Voronoi diagrams, creating an efficient strategy of reallocating customers among the depots. Montoya-Torres et al. (2015) reviewed the literature in detail for many kinds of MVRP such as MVRP with time windows, heterogeneous vehicle fleet, pickups and deliveries, split deliveries and periodic deliveries.

Aras et al. (2011) defined the collecting cores problem of firms in the durable goods industry as a MVRP and designed two mixed integer linear programming models for this problem. They proposed a TS algorithm to solve medium and large-sized problems. Gulczynski et al. (2011) introduced a new problem called multi-depot split delivery VRP, combining two different VRPs, and also proposed a heuristic based on integer programming for this new problem. Yücenur and Çetin Demirel (2011) developed a new type of geometric shape-based genetic clustering algorithm for MVRP. They developed GA based on the proposed clustering technique in order to be used in the solution process of the problem. Kuo and Wang (2012) proposed VNS heuristic for MVRP with loading cost. VNS consisted of three phases: in the first phase, a stochastic method was used to construct the initial solution; in the second, four operators were randomly chosen to search for neighbour solutions; and finally, a criterion as in TS was used to choose the neighbour solution.

1.2.2 Literature Review of MDHFVRP

MDHFVRP was first defined by Salhi and Sari (1997), who proposed a multi-level composite heuristic for solving the problem and designed two reduction tests to

enhance its efficiency. Then, two mathematical models of MDHFVRP were presented and the lower and upper bounds of the problems were found by solving the model by CPLEX in Salhi et al. (2014). They also applied VNS effectively to the problem and found the new best solutions for 23 problem instances out of 26. Bettinelli et al. (2011) firstly proposed an exact solution approach for MDHFVRP with time windows. The proposed method was branch-cut-price method and different pricing and cutting techniques were applied. Xu et al. (2012) proposed VNS heuristic to solve MDHFVRP with time windows in their study. Xu and Jiang (2014) proposed VNS heuristic by improving the method in the previous study. They applied the improved VNS heuristic given good results to the large water project in China. Adelzadeh et al. (2014) proposed bi-objective mathematical model for MDHFVRP with fuzzy time windows and the objectives were minimizing the total cost by reducing the total travelled distance and increasing the service level. They developed a multi-phase method using SA heuristic for the defined problem. Benslimane and Benadada (2013) applied MDHFVRP to the distribution problem in which a large amount of single type products are delivered to the customers. They applied ant colony systems algorithm to solve that problem. Mancini (2016) defined and modelled the multi period MDHFVRP and proposed a math-heuristic approach based on Adaptive Large Neighborhood Search (ALNS) for that problem.

For a better understanding of methods proposed for MDHFVRP, the reader should refer to a comprehensive review published by Montoya-Torres et al. (2015), providing details of many kinds of MVRP such as MVRP with time windows, heterogeneous vehicle fleet, pick-ups and deliveries, split deliveries, and periodic deliveries.

1.2.3 Literature Review of VRPs with Pickups and Deliveries

In this section, we present an extensive literature review for the three related routing problems that constitute our new variant. These include the VRP with Deliveries and Pickups (VRPDP), the VRPB and the MVRPB.

The VRPDP is an extension of the classical VRP. Here, a vehicle picks up a predefined amount of products from customers besides delivering some other products and transports these delivered products to the depot. Since VRPDP is an extension of

VRP, it is an NP-Hard problem (Nagy and Salhi, 2005) as the VRPDP can easily be reduced to the VRP.

Most of the researchers assumed that vehicles can visit pickup (backhaul) customers after visiting all delivery (linehaul) customers (Nagy and Salhi, 2005). They showed the difficulty of arranging the picked up and delivery goods in the vehicle while visiting. There are two different cases when this assumption is relaxed: Simultaneous Pickups and Deliveries (SPD), Mixed Pickups and Deliveries (MPD). In the former, customers can receive and dispatch goods at the same time (in one visit) whereas in the latter, customers are either delivery or pickup locations but not both. In the case a customer happens to be both, this customer is considered as a linehaul and a backhaul customer and hence visited twice by the vehicle. Note that in the MPD, the vehicle can visit linehaul and backhaul customers in any order.

The VRPDP can be divided into three categories, namely, (i) SPD, (ii) MPD and (iii) Deliver First Pickup Second (DFPS) (Salhi and Nagy, 1999). When the MPD and DFPS are combined, the problem is called the VRPB. Three different strategies with respect to backhauling are investigated by Reimann and Ulrich (2006) using Ant Colony Optimization (ACO) for the VRPB with time windows. It was empirically found that the most cost reduction (in terms of fleet size and travel time) was obtained by allowing pickups once the remaining load of the vehicle is less than approximately 25%.

Min (1989) was the first to introduce the SPD where he solved the transportation of books between libraries (one depot, two vehicles and 22 customers). First, customers were divided into two classes, then two Travelling Salesman Problems (TSP) were solved. In order to form a feasible solution from infeasible ones, a penalty was assigned infeasible arcs and then infeasible TSPs was resolved. Dethloff (2001) introduced an application of VRPDP with SPD in reverse logistics and proposed an insertion based heuristic that uses the idea of remaining load in the vehicle. Crispim and Brandao (2005), Chen and Wu (2006) and Montane and Galvao (2006) developed meta-heuristics (generally TS) for the VRPDP with SPD. DellAmico et al. (2006) proposed an exact solution method based on branch and price procedure while Bianchessi and Righini (2007) applied TS algorithm based on complex and variable neighbourhood

heuristics. Ganesh and Narendran (2007) proposed a multi-phase constructive heuristic approach enhanced by a GA, to study for the first time a routing problem where both DFPS and SPD are considered simultaneously. The closest to this work is Gajpal and Abad (2009) who proposed an ACO algorithm for VRPDP with SPD with two multi-route local search methods based on sub-paths operations. The other study is by Avci and Topaloglu (2016) who studied the heterogeneous VRP with SPD and developed an adaptive local search integrated with TS.

There are however a few studies that focus on the VRPDP with MPD in the literature. For instance, Deif and Bodin (1984) adapted Clarke and Wright savings heuristic to solve VRPDP with MPD and DFPS using a backhaul customer insertion which is then improved by Golden et al. (1985). In order to postpone the insertion of a backhaul customer to the route, they calculated the savings between linehaul and backhaul customers by adding a penalty coefficient. Cosco et al. (1988) solved the same problem by combining the savings method and a load dependent insertion method. Remaining load dependent insertion cost was calculated in this method. Golden et al. (1985), Cosco et al. (1988) and Salhi and Nagy (1999) all combined the Clarke and Wright savings heuristic with an insertion based heuristic. In these studies, a Capacitated VRP which includes the linehaul customers only was solved and then backhaul customers are added to the routes. These studies differ in terms of the customer insertion method. While Golden et al. (1985) consider the number of remaining linehaul customers after adding a backhaul customer, Cosco et al. (1988) take into account the total remaining load to be delivered after adding a backhaul customer while Salhi and Nagy (1999) applied a similar method, but they introduced the concept of group insertion (in that work a group is made up of two backhaul customers). A practical solution approach was proposed in Royo et al. (2016) for pallet and package delivery companies by considering a mixed delivery system.

Toth and Vigo (1996) applied DFPS method to the VRPDP and developed Cluster First Route Second (CFRS) heuristic algorithm. Wade and Salhi (2004) designed an ACO algorithm for the VRPDP based on DFPS and MPD. Toth and Vigo (1997) proposed a branch and bound algorithm using Lagrangian relaxation method for VRPB for lower bounds which are then strengthened by the cutting plane approach. Mingozzi

et al. (1999) found the exact solutions of the benchmark instances up to 100 customers of the same problem by using branch and bound algorithm. Many heuristic solution approaches were proposed for the VRPB in the last years. Halse (1992) analyzed many VRP extensions including the VRPB and the VRPDP where solutions up to 100 customers for VRPDP and up to 150 customers for VRPB were reported.

Goetschalckx and Jacobs-Blecha (1989) developed a two-phase heuristic based on a space-filling curve approach for the VRPB. Mosheiov (1998) proposed a heuristic based on a tour destruction approach. Brandao (2006) developed a multi-phase TS algorithm and Wassan (2007) proposed a reactive TS algorithm which is then improved by adaptive memory programming. Osman and Wassan (2002) introduced reactive TS algorithm in which tabu duration is updated during the search. Zachariadis and Kiranoudis (2012) proposed a local search heuristic that extends the size of the search space while Cuervo et al. (2014) developed an iterated local search algorithm consisting of two basic components: In the first component, the algorithm searches larger search space in each iteration by saving the found solutions in the memory, in the second component, it visits the feasible and infeasible parts of the solution space regularly by assigning penalty to infeasible solutions. Furthermore, the new best solutions of two benchmark problem instances were found in that study. An interesting visual interactive solution approach that allows the user (decision maker) to take part in the solution process was designed by Tütüncü et al. (2009) for the VRPB. This was extended to a Decision Support System (DSS) to solve the VRPB and its extensions. Some of the ideas in this approach were adapted accordingly into this thesis.

Taillard (1999) introduced HFFVRP which can be defined as a special case of the Heterogeneous Vehicle Fleet VRP (HVFVRP) with the addition that the number of vehicles in each type is fixed instead. In other words, the HFFVRP aims to find the best routes for the given vehicles, while HVFVRP aims to find the best vehicle fleet combination. Tütüncü (2010) adapted her earlier visual interactive method to solve the HFFVRP and the HFFVRP with Backhauls (HFFVRPB). Here, the user is allowed to take alternative decisions by using his/her knowledge and experience about the HFFVRP and modify the decisions accordingly through a visual DSS.

There are few studies about MVRPB in the literature. Salhi and Nagy (1999)

developed an insertion based heuristic that uses cluster-insertion method for VRPB and adapted this heuristic to the multi-depot problem. They also analyzed SPD and MPD versions of VRPDP. Nagy and Salhi (2005) developed an effective compound heuristic approach for the VRPDP with SPD and MPD versions and applied this heuristic to the multi-depot problem. The proposed heuristics obtained high quality solutions in a few seconds for the VRPDP problem instances with 1 to 5 depots and 50 to 249 customers. Li et al. (2015) proposed a meta-heuristic based on an iterated local search method for the MVRP with Pickups and Deliveries (MVRPPD) with simultaneous pickup and delivery approach where better results were discovered. An optimization algorithm that solves the rich VRP problem including pickups and delivery using VNS and TS algorithms was presented in Sicilia et al. (2016). Irnich (2000) introduced the multi-depot pickup and delivery problem with a single-hub and heterogeneous vehicles which is a special case of the MVRPPD. This problem differs from MVRPPD as the pickup requests are first collected to the hub location, then delivery requests are then dispatched from the hub by a vehicle, of a given heterogeneous fleet, departed from one of the request locations. In addition, every request location served as depots of vehicles and all vehicles starting at a location have to return to the same location at the end of the planning period. It is also worth noting that in their study their primary concern is the assignment of requests to vehicles rather than the routing itself as the trips are short due to narrow time windows and large quantities to deliver as they base their experiments on a real life case study. That is why they opted for a set covering type formulation. In brief, their problem does not have the same structure of MVRPPD in terms of depot definition.

Koç and Laporte (2018) conducted an informative literature review which includes models such as exact and heuristic algorithms, industrial applications and case studies of the VRPB and its extensions. Very recently, an easy to read review on the simultaneous pickup and delivery and its related versions was given by Koç et al. (2020).

The reader will also find the review paper by Berbeglia et al. (2007) who extensively surveyed the VRPDP and presented a classification scheme to be useful and very complementary.

The only work that is closer to ours is the recent study by Penna et al. (2019). In their study, they addressed a family of rich VRPs including the use of heterogeneous fleet with other attributes such as backhauls, multiple depots, among others. Although, they proposed a unified algorithm that is capable of solving VRPs having some extensions, they did not introduce a mathematical model of these MDHFVRPB extensions and also did not provide any data set to test the proposed algorithm on MDHFVRPB. They used a two phase approach where in phase one a pool of promising routes are constructed using an Iterated Local Search with a Randomized Variable Neighbourhood Decent. Phase two uses this set of routes to solve a corresponding set partitioning problem with a commercial solver. Our study differs from theirs in producing a formal mathematical formulation and also in the construction of the initial solution, the use of adaptive learning and the VNS as will be shown in the subsequent sections.

To the best of our knowledge, there is no study that integrates the MVRPB and heterogeneous vehicle fleet which we refer to as the MDHFVRPB. We presented the mathematical models of MDHFVRPB in Section 2.1. Moreover, we can also note that this is also the first time where VNS and Greedy Randomised Adaptive Memory Programming Search (GRAMPS) meta-heuristics are hybridised. We called this new meta-heuristic as hybrid unified VNS based GRAMPS, which we refer to as the VNS-GRAMPS, and reported the details in Section 2.2.

1.2.4 Literature Review of DSS

A DSS was defined as an interactive software tool in Ferreira et al. (2015) that collects data from various sources and presents the user with useful information, displayed on graphical interfaces, that helps solving decision-making problems. DSSs are used to support complex decision-making processes and problem solving by utilising computer technologies. The benefits of using a DSS in operations research or transportation management problem can be summarized as follows: Transport cost reduction, reduction in fuel consumption and environmental impact, improved customer service, effective strategic planning, less reliance on individual skills and tighter control of distribution. The evolution of Computerised Vehicle Routing

and Scheduling (CVRS) systems was reviewed from the perspective of the users' experience, investigating the software capabilities along with the perceived barriers to its future development in Rincon-Garcia et al. (2018). A survey of logistics providers operating in the U.K. was conducted by taking special attention to the VRP software and the set of models that support the decision-making processes. Survey results suggested that companies required improved route optimisation to tackle congestion based on time dependent data and models, and greater accuracy in the representation of the road network. It was observed that operational research techniques are available to solve problems that represent real-world conditions in the literature, but research into the relative merits of using time-dependent models needs to be undertaken. It was suggested that data might be improved by cooperation between government and private sector. Rincon-Garcia et al. (2018) also examined the properties of the algorithms used in different commercial CVRS software. Solution methods for commercial routing software should be able to solve large examples as stated in Drexl (2012). It has been reported that most of the commercial software can solve large-sized problems with unlimited number of customers and vehicles in less than 15 minutes in Rincon-Garcia et al. (2018). Most developers in the industry prefer heuristic methods over meta-heuristic scientific approaches, which can take a long time to solve large-sized problem instances (Sörensen et al., 2008). In addition, it is useful to use heuristic approaches to solve different VRP variants with different logistical constraints available in the industry, as they suggest near-optimal solutions in reasonable CPU time. Although commercial software can solve large problem instances in a short time (minutes or even seconds), the quality (accuracy) of the solution is not consistent. The accuracy of a solution can be defined by calculating the percent difference between the cost of the current solution and the solution applied in the practice or the best solution obtained with the related commercial software. In a test conducted by using different CVRS providers, significant differences in solution quality were found by up to 10% between best and worst schedules in cases involving only 100 customer requests, and this difference increased in larger instances (Bräysy and Hasle, 2014).

Mendoza et al. (2009) proposed a DSS integrated with commercial systems and based on a custom-made distance-constrained routing module. A modified Clarke

and Wright savings heuristic and two memetic algorithms were used in this module, along with two integer-programming clustering models whose function was to balance the workload. It was tested on ten real-world distance-constrained vehicle routing instances consisting of nodes up to 601. Tütüncü (2010) proposed the visual interactive method based on GRAMPS algorithm to solve HFFVRP and HFFVRPB which was defined in the study as an extension of HFFVRP. The user was allowed to take alternative decisions by using his/her knowledge and experience about HFFVRP and to be able to edit the taken decisions by applying the proposed approach to a visual DSS in the study. An integrated transportation solver for MVRP with distance constraints was developed in Tlili and Krichen (2015) by designing a DSS based on the integration of Geographical Information System (GIS) and the iterated local search. The DSS architecture as well as its performance were checked using a real-world case. A new variant of VRP considering a set of goods to be picked up and delivered was proposed in Hsieh and Huang (2015). Each good to be picked up from a source address has a destination address in that problem, and each node has to be visited once in which delivery and pickup operations are performed. Moreover, a DSS based on a discrete PSO method was developed to support the decision of vehicle routes. A model-based DSS was proposed by Min and Melachrinoudis (2016) in order to determine working hours, rest periods of the truck driver, and the schedules and routes of the truck assigned to the driver. A mixed-integer programming model and a simulated annealing meta-heuristic were developed for solving the problem and embedded into the DSS. The developed model was also integrated with a GIS and relational database management system to enhance interfaces between the model and its parametric data using spatial, graphical displays. An open-source Excel based solver called as VRP Spreadsheet Solver was introduced by Erdoğan (2017) to solve many variants of VRP. Two real-world case studies of the solver from the healthcare and tourism sectors were investigated. The solver was capable of solving Capacitated VRP and Distance-Constrained VRP instances with up to 200 customers within 1 h of CPU time. Another open-source, spreadsheet based DSS was introduced for facility location problems (FLP) by Erdoğan et al. (2019). This solver was able to solve capacitated and distance constrained versions of the four basic FLPs: p-median,

p-centre, maximum coverage location problem, and un-capacitated FLP. P-median problems with up to 600 vertices and 200 facilities to near optimality, as well as capacitated p-median problems with up to 100 vertices and 10 facilities could be solved with the solver. Leyerer et al. (2019) claimed that most of the DSSs developed for VRPs were idealized and focus on single problem-tailored routing applications, and they presented a customizable VRP for optimized road transportation embedded into a multi-attribute vehicle routing DSS by addressing this research gap. Various model attributes were integrated to handle a multitude of real-world routing problems. The DSS was evaluated with computational benchmarks and real-world simulations based on the design science research methodology. Computational experiments indicated that the developed DSS can compete with problem-tailored algorithms and contribute to an enhanced economic and environmental sustainability in urban logistic applications. A DSS based on clustering and routing methodology were presented, incorporating the driver's experience, the company's historical data and Google map's data by Zhao et al. (2020). It was claimed that the proposed heuristic performs as well as k-means algorithm while having other notable advantages, and the superiority of the proposed approach was illustrated through numerical examples.

A DSS for route planning of vehicles performing waste collection for recycling was presented in Ferreira et al. (2015). The proposed DSS consisted of three modules: route optimization, waste generation prediction, and multiple-criteria decision analysis (MCDA). A cellular genetic algorithm was developed to solve the route optimization problem. SMART, ValueFn and Analytic Hierarchy Process were the methods employed for the MCDA module. Bi-objective Dynamic Hazardous Materials VRP with Time Windows (DHVRP), aiming to optimize the total transportation cost and the total travel risk under dynamic environments, was introduced in the reference Ouertani et al. (2022). In order to generate the best routes, based on two new meta-heuristics: a bi-population GA and a hybrid approach combining the GA and the VNS, a DSS was developed. An experimental investigation was carried out using 56 benchmarks instances of Solomon and through several performance measures. It was shown that the new approaches were highly competitive with regards to two state-of-the-art algorithms. Ercan and Gencer (2018) introduced a heuristic method to solve the

dynamic heterogeneous unmanned aerial systems routing problems without causing the initial tour to be completely changed. A dynamic routing DSS based on both fuzzy clustering and leveraged cheapest insertion neighbourhood method was developed for the solution of a new target for the strategic level heterogeneous unmanned aerial vehicles, capable of taking into account dynamic time as well as dynamic position and priority.

The rapid development of technology and the conditions of the Covid-19 pandemic in recent years have changed the shopping habits of people around the world. E-commerce and home service sectors were most affected by this drastic change. In both sectors, many new customers place orders dynamically during the day and expect their orders to be delivered as soon as possible. The transportation problems of these companies often include multiple warehouses, heterogeneous vehicle fleets and pick-up loads. Especially e-commerce companies have to collect the product returns of the customers as well as the product distribution to their customers during the day. In order for companies to respond to customer requests quickly and cost-effectively, they need software that can provide fast solutions to dynamic customer demands. Thanks to the interactive tools offered by the DSS proposed within the scope of this thesis, decision makers can add or remove new customers, new vehicles to the problem, and make changes on the proposed vehicle routes by changing the vehicle type or adding-removing customers from the route. In addition, the information about the solved problem such as customers, vehicles, depots and the routes of the proposed solution can be saved in the database, so that it is accessible at all times and the decision maker does not need to load this information into the DSS every time it is used. In addition to these, the decision maker can call the solution of a problem saved in the database whenever he wants and make simple changes on the solution by using interactive tools, and then the problem can be solved again and the best solution can be reached. It was observed that the proposed DSS can solve several variants of VRP and MVRP for up to 360 customers in a short time, which will provide great advantages and convenience for companies with transportation problems in e-commerce and home service sectors.

To sum up, we introduced a modified DSS to solve single and multi depot VRPs with heterogeneous fleet and/or backhaul properties quickly and effectively in this

thesis. The name of the modified DSS was defined as "ADVISER2" by referencing the papers Tütüncü et al. (2009) and Tütüncü (2010). The aim of the developed DSS is to provide user convenience in order to produce new ideas and analyse alternative ones. We also aimed to provide flexibility, user convenience and efficiency, which are necessary for a DSS, to the users. With the visual interactive DSS, the user can make instant changes on the solution by using interactive tools such as adding / deleting customers, adding / deleting vehicles on a solution suggested for a problem.

The central aim of the thesis has been proposed as developing a visual interactive DSS which is able solve single and multi depot VRP extensions with heterogeneous vehicle fleet and backhauls utilising newly developed hybrid unified VNS-GRAMPS meta-heuristic. In order to achieve this aim, we firstly introduced the VRPs that can be solved with VNS-GRAMPS meta-heuristic embedded into the DSS. Then, we reviewed the literature of these problems. In Chapter 2, we firstly defined two mathematical models of the new MVRP type called as MDHFVRP with Backhauls, then we introduced the VNS-GRAMPS meta-heuristic based on the GRAMPS and VNS meta-heuristics with the explanation of main steps. We also explained the modules and the interactive tools of DSS. Computational experiment results were reported in Chapter 3 and the performance of VNS-GRAMPS was discussed.

The contribution of this thesis is five-folds;

(i) The multi-depot routing problem with backhauls and heterogeneous vehicles was studied,

(ii) A new formulation was proposed which is then enhanced by introducing tightening,

(iii) A novel unified hybridisation of VNS and GRAMPS adopting a two stage approach was developed,

(iv) New data sets, based on the commonly used instances from related routing problems, were generated and interesting results obtained for comparison and benchmarking purposes.

(v) A modified visual interactive DSS was developed that is able to solve single and

multi depot VRPs with heterogeneous fleet and/or backhaul properties quickly and effectively by utilising VNS-GRAMPS.

CHAPTER 2: METHODOLOGY

In this chapter, we firstly defined the mathematical model and some tightening for Multi-Depot Heterogeneous Vehicle Routing Problem (VRP) with Backhauls (MDHFVRPB), then we introduced the hybrid unified VNS-GRAMPS meta-heuristic based on the Greedy Randomized Adaptive Memory Programming Search (GRAMPS) and Variable Neighbourhood Search (VNS) meta-heuristics. Finally, we represented the main modules and interactive tools of the Decision Support System (DSS) developed in the scope of this thesis.

2.1 Mathematical Model

We first provide an overview of the problem, necessary notation and the corresponding mathematical formulation. This is then followed using some tightening of the formulation by introducing valid inequalities and new variables.

2.1.1 Overview and Notation

In this section, the MDHFVRPB was modelled by combining earlier formulations for the Fleet Size and Mix VRP with Backhauls (FSMVRPB) proposed by Salhi et al. (2013) and the MDHFVRP also presented by Salhi and Sari (1997). The properties of the MDHFVRPB were summarized as follows: Customers are divided into two groups, namely, delivery (linehaul) and pickup (backhaul) customers. There are more than one depot in the system and there is a heterogeneous vehicle fleet (unlimited number of vehicles in each type) with fixed and variable costs varying according to the vehicle type. Backhaul customers cannot be visited unless all linehaul customers are visited. While a route consisting of only backhaul customers is not allowed, a route may include linehaul customers only if necessary. The vehicle capacity constraint is imposed.

Parameters:

n : Number of customers, $(1,\ldots,n)$,

m : Number of depots, $(n+1,\ldots,n+m)$,

l : Number of linehaul customers, $(1,\ldots,l)$,

b : Number of backhaul customers, $(l+1,\ldots,n)$,

All customers and depots are considered as node $(1,\ldots,n+m)$, where m depots are represented as $(n+1,\ldots,n+m)$, l linehaul customers are represented as $(1,\ldots,l)$, and $b=n-l$ backhaul customers are represented as $(l+1,\ldots,n)$.

q_i : Demand of customer i $(i=1,\ldots,l)$ and $q_i=0$ for $i=l+1,\ldots,n+m$,

p_i : Supply of customer i $(i=l+1,\ldots,n)$ and $p_i=0$ for $i=1,\ldots,l$ and $i=n+1,\ldots,n+m$,

K : Number of vehicle types,

Q_k : Capacity of vehicle type k $(k=1,\ldots,K)$,

f_k : Fixed cost of vehicle type k $(k=1,\ldots,K)$,

α_k : Variable cost of vehicle type k $(k=1,\ldots,K)$,

D_{ij} : Distance between customers i and j $(i,j=1,\ldots,n+m)$.

Decision Variables:

$$x_{ijdk} = \begin{cases} 1, \text{if the vehicle } k \text{ originating from depot } d \text{ and travelling along} \\ \text{arc } (i,j) \text{ is chosen;} \\ 0, \text{otherwise.} \end{cases}$$

where $i,j=1,\ldots,n+m;\ k=1,\ldots,K;\ d=n+1,\ldots,n+m.$

y_{ij} = The total remaining load on the vehicle travelling along arc (i,j) before reaching customer j.

2.1.2 The Initial Mathematical Formulation

$$Min\ Z = \sum_{d=n+1}^{n+m}\sum_{k=1}^{K} f_k \sum_{i=n+1}^{n+m}\sum_{j=1}^{l} x_{ijkd} + \sum_{d=n+1}^{n+m}\sum_{k=1}^{K}\sum_{i=1}^{n+m}\sum_{j=1}^{n+m} \alpha_k D_{ij} x_{ijkd} \qquad (1)$$

Subject to

$$\sum_{d=n+1}^{n+m}\sum_{k=1}^{K}\sum_{i=1}^{n+m} x_{ijkd} = 1, \qquad j = 1,\ldots,n, \tag{2}$$

$$\sum_{d=n+1}^{n+m}\sum_{k=1}^{K}\sum_{j=1}^{n+m} x_{ijkd} = 1, \qquad i = 1,\ldots,n, \tag{3}$$

$$\sum_{i=1}^{n+m} x_{ijkd} = \sum_{i=1}^{n+m} x_{jikd}, \qquad k = 1,\ldots,K;\ j = 1,\ldots,n+m;\ d = n+1,\ldots,n+m, \tag{4}$$

$$\sum_{i=n+1}^{n+m}\sum_{j=1}^{l} y_{ij} = \sum_{j=1}^{l} q_j, \tag{5}$$

$$\sum_{i=l+1}^{n}\sum_{j=n+1}^{n+m} y_{ij} = \sum_{i=l+1}^{n} p_i, \tag{6}$$

$$\sum_{i=1}^{l} y_{ij} + \sum_{i=n+1}^{n+m} y_{ij} = \sum_{i=1}^{n+m} y_{ji} + q_j, \quad j = 1,\ldots,l, \tag{7}$$

$$\sum_{i=l+1}^{n} y_{ji} + \sum_{i=n+1}^{n+m} y_{ji} = \sum_{i=1}^{n} y_{ij} + p_j, \ j = l+1,\ldots,n, \tag{8}$$

$$y_{ij} \le \sum_{d=n+1}^{n+m}\sum_{k=1}^{K} Q_k x_{ijkd}, \qquad i \ne j = 1,\ldots,n+m, \tag{9}$$

$$y_{ij} = 0, \qquad i = 1,\ldots,l,\ and\ j = l+1,\ldots,n+m, \tag{10}$$

$$y_{ij} = 0, \qquad i = n+1,\ldots,n+m,\ and\ j = n+1,\ldots,n+m, \tag{11}$$

$$y_{ii} = 0, \qquad i = 1,\ldots,n, \tag{12}$$

$$x_{d_1ikd_2} = 0, \qquad i = 1,\ldots,n;\ k = 1,\ldots,K;\ d_1 \ne d_2 = n+1,\ldots,n+m, \tag{13}$$

$$x_{id_1kd_2} = 0, \qquad i = 1,\ldots,n;\ k = 1,\ldots,K;\ d_1 \ne d_2 = n+1,\ldots,n+m, \tag{14}$$

$$x_{djkd} = 0, \qquad j = l+1,\ldots,n;\ k = 1,\ldots,K;\ d = n+1,\ldots,n+m, \tag{15}$$

$$x_{ijkd} = 0, \qquad i = l+1,...,n;\ j = 1,...,l;\ k = 1,...,K;\ d = n+1,...,n+m, \tag{16}$$

$$x_{ijkd} \in \{0,1\}, \qquad i,j = 1,\ldots,n+m;\ k = 1,\ldots,K;\ d = n+1,\ldots,n+m, \tag{17}$$

$$y_{ij} \ge 0, \qquad i,j = 1,\ldots,n+m, \tag{18}$$

The objective function (1) aims to minimize the total cost. In the first part of the objective function, the fixed cost of each used vehicle is added and the multiplication of variable cost and travelled distance of each used vehicle is also added to the total cost in the second part. While constraint sets (2) and (3) ensure that each customer must be visited by only one vehicle and only once, constraint set (4) ensures the continuity of each route and completion by one vehicle. Constraint (5) equates the

total load send from depots to linehaul customers with the total sum of the demands of all linehaul customers. The total load coming from backhaul customers to depots and the total supplies of backhaul customers are equated in constraint (6). Constraints (7) and (8) control the entering and leaving load flow for linehaul and backhaul customers, respectively. The upper bound of the load carried along each arc is equated to the capacity of the vehicle travels along that arc in constraint (9). Constraints (10) guarantee that there is no carried load from linehaul customers to backhaul customers and depots. Also, the carried load amount between depots is not allowed in constraint (11) and the carried load from customer to itself is also not permitted in constraint (12). Constraint (13) and (14) impose that a vehicle departs and returns to the same depot. While constraint (15) avoids travelling of the vehicles from depots to backhaul customers, constraint (16) avoids travelling of the vehicles from backhaul customers to linehaul customers. Binary decision variables are defined in constraint (17) and continuous, non-negative decision variables are given in constraint (18). It is worth noting that as the type and originating depot of the vehicle travelling along arc (i, j) are determined by the binary decision variable x_{ijdk}, it is not necessary to include indices k and d for the continuous, non-negative decision variable y_{ij}.

2.1.3 Maximum travel distance constraint

In case a maximum travel distance constraint, i.e., a route length constraint, is imposed (TD), the following constraint set, adapted from Kara (2011), were added with the new positive continuous variable T_{ijd}. T_{ijd} represents the shortest length travelled to customer j from depot d with i being the predecessor of j.

$$\sum_{j\neq i}^{n+m} T_{ijd} - \sum_{j\neq i}^{n+m} T_{jid} = \sum_{j=1}^{n+m} D_{ij} \sum_{k=1}^{K} x_{ijkd},\ i = 1,\ldots,n;\ d = n+1,\ldots,n+m, \tag{19}$$

$$T_{ijd} \leq (TD - D_{jd}) \sum_{k=1}^{K} x_{ijkd}, \qquad i \neq j = 1,\ldots,n;\ d = n+1,\ldots,n+m, \tag{20}$$

$$T_{idd} \leq TD \sum_{k=1}^{K} x_{idkd}, \qquad i = 1,\ldots,n;\ d = n+1,\ldots,n+m, \tag{21}$$

$$T_{ijd} \geq (D_{ij} + D_{di}) \sum_{k=1}^{K} x_{ijkd}, \qquad i \neq j = 1,\ldots,n;\ d = n+1,\ldots,n+m, \tag{22}$$

$$T_{did} = D_{di} \sum_{k=1}^{K} x_{dikd}, \qquad i = 1,\ldots,n;\ d = n+1,\ldots,n+m, \tag{23}$$

$$T_{d_1 d_2 d_3} = 0, \qquad d_1, d_2, d_3 = n+1,\ldots,n+m, \tag{24}$$

$$T_{id_1 d_2} = 0, \qquad i = 1,\ldots,n;\ d_1 \neq d_2 = n+1,\ldots,n+m, \tag{25}$$

$$T_{ijd} \geq 0, \qquad i, j = 1,\ldots,n+m;\ d = n+1,\ldots,n+m. \tag{26}$$

2.1.4 Some tightening of the formulation

Salhi et al. (2013) found important to restrict the mathematical model by decomposing some constraints of VRP with Backhauls (VRPB) for linehaul and backhaul customers and they generally obtained better Upper Bound (UB) and Lower Bound (LB) values in less CPU time with the restricted model. The same restriction method is adopted to the proposed model. We removed the parts in which the decision variables take value zero from the constraints (2)-(4). Furthermore, we redefined the carried load on each arc separately for linehaul and backhaul customers in constraint (9).

1- Redefining constraint (2)

Constraint (2) is redefined as two constraints as follows, the first part stands for linehaul customers and the second part for backhaul customers. This formulation has the same number of constraints as the previous one, but it has a fewer number of decision variables.

$$\sum_{d=n+1}^{n+m}\left(\sum_{k=1}^{K}\sum_{i=1}^{l}x_{ijkd}+\sum_{k=1}^{K}\sum_{i=n+1}^{n+m}x_{ijkd}\right)=1,\ j=1,\ldots,l, \qquad (2a)$$

$$\sum_{d=n+1}^{n+m}\sum_{k=1}^{K}\sum_{i=1}^{n}x_{ijkd}=1, \qquad j=l+1,\ldots,n. (2b)$$

2- Redefining constraint (3)

Constraint (3) is redefined similarly as two constraints. This new formulation has also a fewer decision variables.

$$\sum_{d=n+1}^{n+m}\sum_{k=1}^{K}\sum_{j=1}^{n+m}x_{ijkd}=1, \quad i=1,\ldots,l, \qquad (3a)$$

$$\sum_{d=n+1}^{n+m}\sum_{k=1}^{K}\sum_{j=l+1}^{n+m}x_{ijkd}=1,\ i=l+1,\ldots,n. (3b)$$

3- Redefining constraint (4)

Constraint (4) can be divided into three constraints for linehaul customers, backhaul customers and depots. This formulation has less number of constraints and less number of decision variables.

$$\sum_{i=1}^{l}x_{ijkd}+\sum_{i=n+1}^{n+m}x_{ijkd}=1, \qquad k=1,\ldots,K,\ j=1,\ldots,l,\ d=n+1,\ldots,n+m, \qquad (4a)$$

$$\sum_{i=1}^{n}x_{ijkd}=\sum_{i=l+1}^{n+m}x_{jikd}, \qquad k=1,\ldots,K,\ j=l+1,\ldots,n,\ d=n+1,\ldots,n+m, (4b)$$

$$\sum_{j=n+1}^{n+m}\sum_{i=1}^{l}x_{jikd}=\sum_{i=1}^{n+m}\sum_{j=n+1}^{n+m}x_{ijkd},\ k=1,\ldots,K,\ d=n+1,\ldots,n+m. \qquad (4c)$$

4- Redefining constraint (9)

Constraint (9) can be replaced with four constraints as shown below:

$$y_{dj} \leq \sum_{k=1}^{K} Q_k x_{djkd}, \qquad j = 1,\ldots,l,\ d = n+1,\ldots,n+m, \tag{9a}$$

$$y_{id} \leq \sum_{k=1}^{K} Q_k x_{idkd}, \qquad i = l+1,\ldots,n,\ d = n+1,\ldots,n+m, \tag{9b}$$

$$y_{ij} \leq \sum_{k=1}^{K} (Q_k - q_i) x_{ijkd}, \ i \neq j = 1,\ldots,l,\ d = n+1,\ldots,n+m, \tag{9c}$$

$$y_{ij} \leq \sum_{k=1}^{K} Q_k x_{ijkd}, \qquad i \neq j = l+1,\ldots,n,\ d = n+1,\ldots,n+m. \tag{9d}$$

Some Observations

The original model has $(n+m)(Km+n+m)+n-m$ constraints, $O(Km^2+Kmn+n^2)$, whereas the restricted model, using the substitution of $n = l+b$ for simplicity, has $2n+Km(n+1)+m(n^2-2l(n-l))$ constraints, $O(mn^2+Kmn)$. A similar calculation is performed for the decision variables. Our modifications have therefore resulted in a reduction of $2mn-n+2lm(n-l)+(m-1)(Km-n^2+m)$ constraints, $O(Km^2+n^2-mn^2)$, and $2m^2-5l^2+5mn-2lm+6ln-n-m$ decision variables, $O(m^2+n^2+mn)$. The restricted model generally obtained better LB and UB values while requiring less or the same amount of CPU time (3 hours) as shown by the interesting and convincing results in the computational experiments section.

The proposed model can easily be adapted to the other VRPB problems studied in the literature. For example, a route formed by only backhaul customer is not allowed in our model, but this constraint can be relaxed by extracting the constraint (15) from the model. In the restricted model, using the original constraints will be sufficient instead of defining restricted constraints for constraint sets (4) and (9).

In this model, it is assumed that the number of vehicles in each type is unlimited. If the number of vehicles in each vehicle type, K_k, is known, these constraints, $\sum_{d=n+1}^{n+m}\sum_{j=1}^{l} x_{djkd} \leq K_k$, need to be added to the above model. In the case the number of all vehicles, NV, is known, $\sum_{d=n+1}^{n+m}\sum_{j=1}^{l}\sum_{k=1}^{K} x_{djkd} \leq NV$ constraint must be added to the model.

2.2 Hybrid Unified Variable Neighbourhood Search with GRAMPS Algorithm (VNS-GRAMPS)

In this section, firstly an overview of the hybrid unified VNS-GRAMPS meta-heuristic algorithm which is embedded into the DSS was provided. This algorithm was proposed to solve the problems of single depot Heterogeneous Fixed Fleet VRP (HF-FVRP), FSMVRPB, Multi-depot VRP (MVRP), MVRP with backhauls (MVRPB), MDHFVRP and MDHFVRPB. In this context, the algorithm was developed to find the solution with the minimum cost using a given vehicle fleet or by finding a vehicle fleet combination automatically.

2.2.1 Overview

The GRAMPS algorithm consists of two stages both using the Greedy Randomized Adaptive Search Procedure (GRASP) algorithm. In the first stage, the Reactive GRASP (RGRASP) algorithm, in which the parameter of the Restricted Candidate List (RCL) and neighbourhood size parameter used in the local search heuristics are automatically set. In this stage, the best solution and the appropriate parameter values that yield the best solution are recorded in the memory. In other words, the first stage acts as a training stage whose chosen parameters will be used in stage two. Here, the best solution found so far is used as the initial solution for the GRASP algorithm to search for new solutions based on the parameters identified earlier in stage one. Within the search we adapted techniques to produce initial solutions, a learning process to identify the most appropriate parameters values for the RCL and neighbourhood sizes as well as some guidance on how to implement the local searches based on their respective performances. In this study, we only considered increasing the RCL and neighbourhood size parameters. However, adding flexibility for these parameters by allowing both the increase and the decrease by adopting a certain rule could be worth examining in the future.

In stage 1, RCL length and neighbourhood size parameters are re-actively increased by one in every 10 iterations and both are initialized to 5. The parameter values that result in the best solution are used in the second stage.

2.2.2 The Algorithm VNS-GRAMPS

The GRAMPS algorithm was proposed by Ahmadi and Osman (2005) for the capacitated clustering problem, and then by Tütüncü et al. (2009) and Tütüncü (2010) for the visual interactive DSS in order to solve VRPB, and Heterogeneous Vehicle Fleet VRP (HVFVRP), respectively. The GRAMPS algorithm consists of running two GRASP algorithm extensions respectively. The GRASP algorithm in the first phase is actually the RGRASP algorithm, in which the length of RCL is automatically set. In this phase, the length of RCL which provides the best solution and the best solution itself are stored in memory by using adaptive memory. The GRASP algorithm used in the second phase of the GRAMPS algorithm starts with the best solution found in the first phase and constructs new solutions using the length of RCL where the best solution was found in the first phase. GRAMPS algorithm was given in Algorithm 1. The GRASP is a flexible meta-heuristic which was introduced by Feo and Resende (1995). The GRASP procedure has proven its success for a wide range of combinatorial optimization problems, from scheduling and routing problems (Festa and Resende, 2009b) to graph colouring and satisfiability problems (Laguna and Martí, 2001; Felici et al., 2017). The versatility in solving NP-Hard problems and the simplicity of the application made GRASP meta-heuristic an excellent solution procedure for VRP. Carreto and Baker (2002), Chaovalitwongse et al. (2003), Prins et al. (2006), Villegas et al. (2011) and Haddadene et al. (2016) showed the effectiveness of GRASP heuristic in solving complex VRP problems in their studies. Detailed analysis of the theoretical framework of the GRASP procedure and its most popular applications can be found in Festa and Resende (2009a) and Festa and Resende (2009b). Festa et al. (2018) analyzed an extension of the classical GRASP meta-heuristic to solve VRP with stochastic demands. In that study, they hybridized bi-random GRASP with a two-stage Monte Carlo simulation capable of generating robust and competitive solutions. RGRASP was described intuitively by Prais and Ribeiro (2000) and Cantu-Funes et al. (2018) proposed the RGRASP method for an extension of MVRP. The value of the quality parameter which is used to restrict the candidate list in the RGRASP procedure was automatically adjusted during the

solution process according to the problem instance at hand. Therefore, the user does not need to empirically determine the quality parameter value.

Algorithm 1 GRAMPS Algorithm

Start:
User selects the initial solution;
The best feasible cost $f_{c_{best}} = \infty$;
The best infeasible cost $i_{c_{best}} = \infty$;
The total feasible cost $f_{c_{total}} = 0$;
Number of iterations $Iter_{number} = 0$;
While(The termination condition is not satisfied)
{
Construct the initial solution with AISCA;
Construct the greedy randomized solution, x, by using REDSA heuristic;
Improve the solution x to x' by using one-node and two-node heuristics;
If (x' is feasible)
{
Improve the solution x' to x'' by using one-node and two-node heuristics;
Improve x'' by applying 3-Opt heuristic;
$f_{c_{total}} = f_{c_{total}} + f(x'')$;
$Iter_{number} = Iter_{number} + 1$;
If($f(x'') < f_{c_{best}}$)
{
$f_{c_{best}} = f(x'')$;
$x_f^* = x''$;
}
}
Else If ($f(x') < i_{c_{best}}$)
{
Improve x' by applying 3-Opt heuristic;
$i_{c_{best}} = f(x')$;
$x_i^* = x'$;
}
}
$f_{c_{average}} = \frac{f_{c_{total}}}{Iter_{number}}$;
Show the best feasible solution and the best feasible cost, if any. Otherwise, show the best infeasible ones;
Update the Adaptive Memory including the best solution with its parameters and average feasible cost in order to use in second phase;
End.

The classical GRASP algorithm consists of two main phases: the construction phase and the local search phase. In the construction phase, a solution is created repeatedly by adding one element from the candidate list to the solution each time. In each iteration, the element to be inserted to the solution under construction is selected uniformly at random from the elements in the RCL. The RCL stores the best insertion candidates based on some greedy score function. There are many ways of defining such a score function, including a certain number from the best, a percentage deterioration from the best among others. One commonly used approach which we

adopt here is the latter where all insertion candidates whose values are within such a threshold (i.e., a percentage of the best score). This can be set, adjusted dynamically during the search or determined by the user. This adaptive thresholding process is controlled by a parameter which adjusts the greediness ratio of the construction phase. A feasible solution is obtained after the completion of the first phase which is then used a starting solution for the local search. The two phases are applied repeatedly and the best solution found is considered as the final solution. For more information on this issue and heuristic search in general, see Salhi (2017).

In this study, we solve the single-depot and multi-depot VRP variants by developing a new hybrid unified GRAMPS meta-heuristic which applies VNS procedure in the local search step. We refer to this hybrid GRAMPS meta-heuristic as VNS with GRAMPS, or VNS-GRAMPS for short.

The first phase of VNS-GRAMPS procedure is a RGRASP algorithm and it consists of a solution construction and local search steps. The solution construction step starts with the selection of the initial seed solution, i.e., determination of the vehicle combination in each depot. This is constructed using the newly proposed Initial Seed Solution Construction Algorithm (ISSCA) which is given in section 2.3.1. Then, the modified Relative Distance Search Algorithm (REDSA) (Tütüncü et al., 2009; Tütüncü, 2010) which is described in section 2.3.2 constructs the initial solution by inserting unassigned customers to the routes of the generated initial seed solution. In the local search step, a simple implementation of the VNS algorithm is applied to find an improved solution if possible. The details of the applied VNS algorithm is presented in Section 2.3.3.

The pseudo-code of the first RGRASP stage of the VNS-GRAMPS meta-heuristic is given in Algorithm 2. The best feasible solution of this first stage is defined as x_f^* and the best infeasible solution is defined as x_i^* in this algorithm.

At the beginning of the GRASP phase, all routes of the best solution, recorded in the first RGRASP stage, are sorted in decreasing order with respect to the cost. Then the GRASP phase starts with the first route having the maximum cost and iterates for all routes. At the beginning of each iteration, the Seed Improvement Algorithm (SIA) is applied to obtain the initial seed solution by marking the customers of the related

Algorithm 2 RGRASP Phase of VNS-GRAMPS Algorithm

Start:
Generate the initial seed solution with ISSCA, see Section 2.3.1;
Initialize the best feasible cost, infeasible cost and the total feasible cost, respectively, $f_{c_{best}} = \infty,\ i_{c_{best}} = \infty,\ f_{c_{total}} = 0$;
Number of iterations $Iter_{number} = 0$;
While(The termination condition is not satisfied) {
Start with the initial seed solution and construct the greedy randomized solution, x, by using REDSA, see Section 2.3.2;
Use the solution x as the initial solution of VNS as briefly described below while its details including the neighbourhood structures and the local searches are given in Section 2.3.3;
While(The maximum # iterations is not reached){
k = 1;
Repeat (Until $k = k_{max}$){
(a) *Shaking:* Generate a random solution, x' from the k^{th} neighbourhood of x ($x' \in N_k(x)$);
(b) *Local search:* Find the local optimum solution, x'', around x' by using the related local search heuristic;
(c) *Movement:* If $f(x'') < f(x)$, then change the incumbent solution ($x = x''$) and return ($k = 1, N_1$) and continue to search from there. Otherwise, increase the neighbourhood size ($k = k + 1$). }
}
If (x is feasible){
$f_{c_{total}} = f_{c_{total}} + f(x)$;
$Iter_{number} = Iter_{number} + 1$;
If($f(x) < f_{c_{best}}$)
$f_{c_{best}} = f(x)$; $x^*_f = x$; }
Else If ($f(x) < i_{c_{best}}$)
$i_{c_{best}} = f(x)$; $x^*_i = x$;
}
$f_{c_{average}} = \frac{f_{c_{total}}}{Iter_{number}}$;
If there is at least one feasible solution, record the best feasible solution and the best feasible cost. Otherwise, record the best infeasible one;
Record the best feasible solt. with its parameter and the average feasible cost.
Go to the GRASP phase as given in Algorithm 2.
End.

route and one of its adjacent route as unrouted. Only two customers, one from each route, remain as routed on these two adjacent routes by using a method defined in Section 2.3.4. The other routes except these two adjacent routes remain unchanged. After obtaining the initial seed solution, an initial solution is constructed with REDSA by assigning the unrouted customers. The aim of the SIA given in Section 2.3.4 is to construct better initial solutions by considering past information. The local search step of GRASP phase also uses the same VNS algorithm defined in Section 2.3.3 with the best neighbourhood size parameter saved in the memory during the RGRASP phase. The GRASP phase continues until a feasible solution that costs less than the

average feasible cost saved in memory of the RGRASP phase is found. However, if no solution can be obtained after a certain number of infeasible solutions, the search terminates. Note that the search also stops after a certain number of feasible solutions are identified. The pseudo-code of the GRASP phase of VNS-GRAMPS is given in Algorithm 3 where the best feasible solution is defined as x_f^{**} and the best infeasible solution as x_i^{**}.

Algorithm 3 GRASP Phase of VNS-GRAMPS Algorithm

Start:
Set the best solution found in RGRASP phase as the initial solution;
The best feasible cost $f_{c_{best}} = f(x_f^*)$;
The best infeasible cost $i_{c_{best}} = i(x_i^*)$;
The average feasible cost $f_{c_{average}}$ = The average feasible cost found in RGRASP phase;
Until (All routes are visited in decreasing order with respect to the cost) {
Construct a new initial seed solution from the best solution using the Seed Improvement Algorithm (SIA) as described in Section 2.3.4.
While($f_{c_{best}} > f_{c_{average}}$ || termination condition is not satisfied) {
Start with the changed initial seed solution;
Construct a greedy randomized initial solution, x, using REDSA;
Improve the solution x to x' by using VNS;
If(x' is feasible && $f(x') < f_{c_{best}}$) {
$f_{c_{best}} = f(x')$;
$x_f^{**} = x'$; }
Else If (x' is infeasible && $f(x') < i_{c_{best}}$) {
$i_{c_{best}} = f(x')$;
$x_i^{**} = x'$; }
}
If ($x_f^{**} \neq \emptyset || x_i^{**} \neq \emptyset$){
Update the initial solution with the best solution found; }
}
Record the best solution found;
End.

For completeness, a flow chart describing the overall algorithm of VNS-GRAMPS meta-heuristic is also provided in Figure 1.

2.3 Explanation of the Main Steps

In this section we describe the main ingredients that constitute algorithms 2 and 3. This includes the way the initial solutions are found and the VNS platform with its neighbourhood structures and the various local searches used.

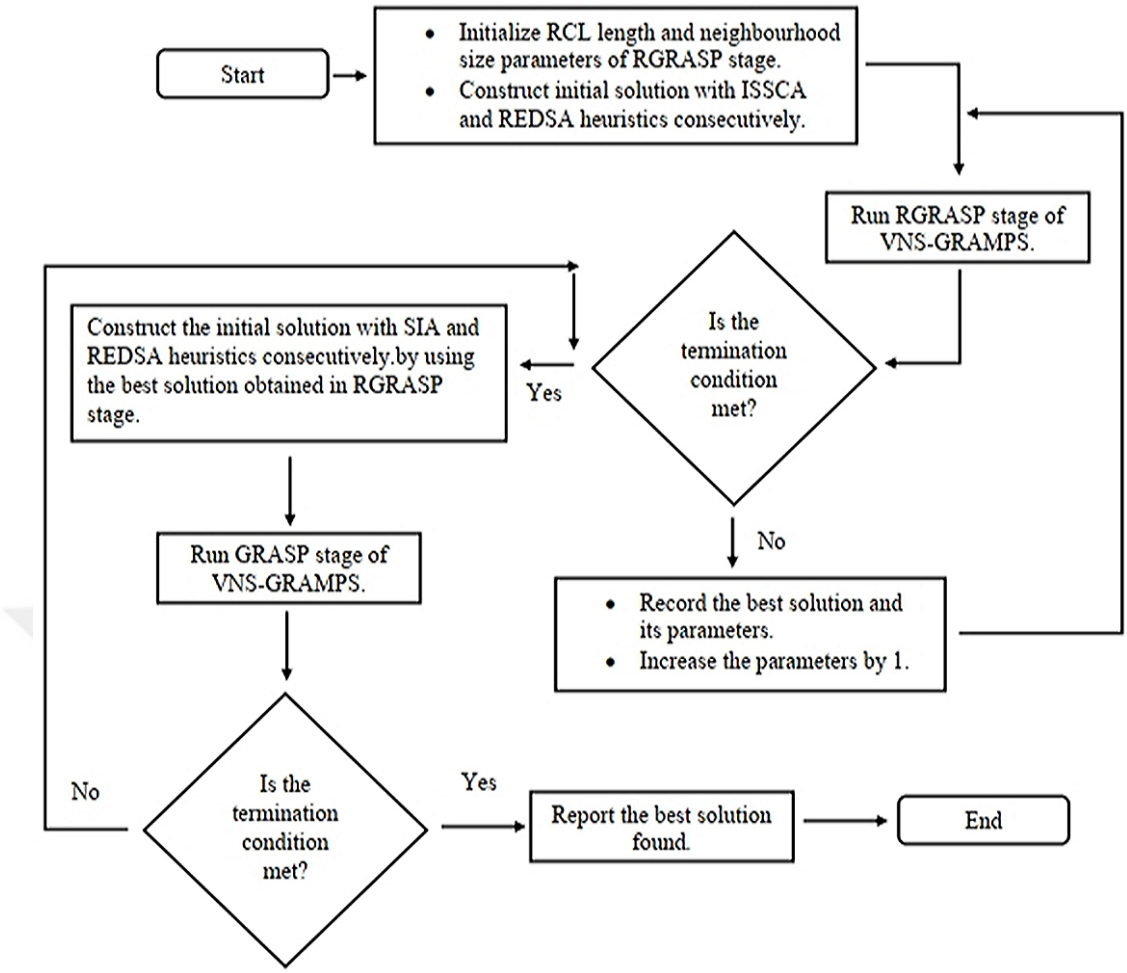


Figure 1. The flow chart of the overall VNS-GRAMPS meta-heuristic (Source: Kocatürk et al., 2021)

2.3.1 Initial Seed Solution Construction Algorithm (ISSCA)

GRAMPS algorithm requires an initial seed solution determining the vehicle fleet information to start. The quality of the solution highly depends on the used vehicle combination in heterogeneous VRPs. Therefore, we have developed ISSCA, which is not only fast, but it can also provide a good initial seed solution with a good vehicle combination. This procedure consists of three main steps.

The ISSCA Procedure

Step 1- ISSCA starts by clustering customers around the depots by using the clustering method given by de Oliveira et al. (2016). Two rules are defined to cluster customers around the depots: (a) The closest depot of customer j and (b) The closest

depot of customer k which is the closest to customer j. If the results of these two rules result in having the same depot, then j is assigned to such a depot. However, If the closest depot of j is not the one assigned to customer k, then j can be assigned to both (i.e., customer j will be included in the clusters of both depots). In other words, customer j is known as a borderline customer of these two depots as defined in Salhi and Sari (1997).

Step 2- The total demand for each depot/cluster is calculated, based on the demands of the customers clustered around the depot.

Step 3- An initial seed solution is generated by constructing routes in each depot based on the customer clusters around each depot. This guides the search to determine the order of the depots when constructing the routes while respecting the total demand information of each depot. To provide extra flexibility, we adopt three initial seed solutions to order the depots. These include (i) the depots are sorted in increasing order with respect to the total demand, (ii) same as (i) but in a decreasing order and (iii) using a random depot order. ISSCA then selects the solution having the minimum cost among these three solutions as the initial seed solution. The routes are obtained following the scheme given below.

The Construction of the Routes

This is a cluster first route second type heuristic. In each depot, the routes are constructed as follows:

1. The first route r is generated by determining the vehicle type k randomly among the possible vehicle types.
2. Customer i closest to the depot is assigned as the first customer of the first route r.
3. The unrouted customer j closest to customer i is assigned to route r.
4. We continue to assign the unrouted customers closest to the last assigned customer of route r until the capacity of the vehicle type k is exceeded.
5. A complete potential route is formed by joining the last assigned customer to the depot.

6. At this stage the information of the cost of the complete route, the configuration of the route, the last assigned customer and the vehicle type are recorded.

7. The next larger vehicle if any, is then considered and the search continues in the same way to assign the customer closest to the last saved assigned customer. It is worth noting that in the case the vehicle type k has a greater capacity than a pre-determined threshold which we will define later, we assign customer j that has the minimum distance per unit of demand instead, (i.e., the ratio $\frac{D_{ij}}{d_j}$). We introduced this rule to favour the assignment of larger customers to those vehicles with a larger capacity.

8. If the total load of route r violates the largest vehicle capacity, the search stops assigning customers to route r, and determines the vehicle type k^*, where the last added customer j_{last} is the closest to the depot using the stored information. In other words, route r is constructed by using vehicle k^* and containing all customers from i to j_{last}.

9. The next route is then constructed using a random vehicle type, and the customers are assigned following the same procedure. These steps are repeated until all customers allocated to the depot are assigned. The search starts again with the next chosen depot and its corresponding customers. This process is repeated until all depots are considered.

Note that the customers included in two clusters (borderline customers) are assigned to the routes originated from their two corresponding depots. This flexibility provides more chance for the local search operators to find better routes and better vehicle types. Since ISSCA will find a vehicle fleet combination whose total capacity is larger than the total demand of the customers, empty routes are incorporated into the search but are obviously deleted from the final solution.

GRAMPS algorithm is an iterative search procedure and it needs an initial solution construction algorithm (REDSA) to start the search in each iteration. In REDSA, where its details are given next, the solution is constructed based on a combination of greediness and randomness. This specification constitutes the greedy randomized

nature of GRAMPS. To obtain different initial solutions by REDSA in each iteration of GRAMPS, we therefore opted to leave only one linehaul customer located in the middle position on each route generated by ISSCA.

Moreover, the initial solution can also be constructed by using the interactive tools of the DSS or by selecting a saved solution/initial solution saved into the database. If a saved solution from the database is used as the initial solution, the solution can easily be modified by using the interactive tools to obtain a better initial solution.

2.3.2 RElative Distance Search Algorithm (REDSA)

The aim of the REDSA heuristic is to construct an initial solution by assigning the unrouted customers in the initial seed solution generated by ISSCA. Here, each customer is assigned to one route only. The construction of the routes is based on the following three steps.

Step 1- The insertion cost and the insertion position are evaluated according to the approach proposed by Baker (1992) for each unassigned customer. This insertion cost is calculated as follows: c_{rj} represents the insertion cost of customer j to route r, and this is calculated only for the customers included in the cluster of the depot that the route r is originated from. This is performed for each route r.

$$c_{rj} = \alpha_r \min_{0 \le l \le |I_r|+1} \{D_{i_l j} + D_{j i_{l+1}} - D_{i_l i_{l+1}}\}, \tag{27}$$

where $I = \{1, \ldots, n+m\}$ is the set of nodes (n customers and m depots), I_r is the set of customers assigned to route r, i_0 and $i_{|I_r|+1}$ represent the depot, and α_r represents the variable cost of the route r. The problem constraints are checked during the calculation, and the position, p_{rj}, of the customer j in the route r that gives the minimum insertion cost is saved.

Step 2- The unassigned customers are individually assigned to the routes. The rule of customer insertion into the routes prioritizes customers with respect to the number of possible route assignments and the least insertion cost. The customers

that can only be assigned to a single route are prioritized according to this rule. Then the first and the second least insertion costs are calculated for the customers who can be assigned to multiple routes. This is commonly known as the regret cost or opportunity cost. If there are more than one customer that can only be assigned to a single route, $R_j = 1$, where R_j is the number of routes that customer j can be assigned to, the one with the highest demand is chosen. Each iteration includes a recalculation of the number of routes, R_j. If there is at least one customer which cannot be assigned to any route with $R_j = 0$, there is no feasible solution. In the case of getting an infeasible solution, the heuristic continues to assign customers until all possible customer assignments are completed.

The minimal first and second insertion costs of customer j are represented as c_j^{1st} and c_j^{2nd}, respectively. The insertion priority coefficient of customer j is represented as $IP_j = c_j^{2nd} - c_j^{1st}$, i.e., IP_j coefficient is equal to the difference between the minimal first and second insertion costs of customer j. The customer with the highest insertion priority coefficient is given priority during the assignment.

If there are no customers with $R_j = 1$, a customer is selected randomly from the RCL including the customers that can be assigned to multiple routes. RCL is constructed by selecting a certain number of customers with the highest insertion priority coefficients. In the pseudo-code of the REDSA algorithm, given in Algorithm 4, R^1 represents the number of customers that can only be assigned to a single route.

Here in this step, all insertion costs c_{rj}, respective positions p_{rj} and insertion priority coefficients IP_j are recalculated for each unassigned customer when a customer is assigned to a route. This step is repeated until all possible customer assignments are completed.

Step 3- The 3-Opt heuristic with the best-improvement strategy is applied to all constructed routes. In this procedure, the heuristic calculates all possible 3-arc swaps in a route and applies the 3-arc swap with the maximum saving.

It is worth stressing that although REDSA aims to satisfy the problem constraints,

we may still obtain an infeasible solution because of the existence of unassigned customers.

Algorithm 4 REDSA Solution Construction Heuristic

```
Start:
    Calculate c_rj and p_rj for each unassigned customer j and route r;
    Find c_j^1st and c_j^2nd for each unassigned customer j, where c_j^1st is the least
       insertion cost of customer j among c_rjs, and c_j^2nd is the second least cost;
    Calculate IP_j and R_j for each unassigned customer j;
    R^1 = 0;
    While (There is a possible customer assignment) {
        If (R^1 > 0) {
            j* = the customer j with the highest demand and R_j = 1;}
        Else {
            Construct RCL by using IP_j coefficients of related customers;
            Select customer j* from RCL randomly;
            Select r* with least insertion cost (c_rj) of the customer j*;
            Insert the customer j* to the position p_r*j*;
            Mark the customer j* as assigned;
            Update remaining capacity and distance for the route r*;
            Update the parameters c_r*j, p_r*j, c_j^1st, c_j^2nd, IP_j, and R_j for
            each unassigned customer j;
            Update R^1 value; }
    }
    Apply 3-Opt heuristic to all routes;
End.
```

In stage one of VNS-GRAMPS, we set the initial length of the RCL to 5 and we increase it by one in every 10 iterations. These two parameters, namely 5 and 10, are empirically identified to be reliable after preliminary experiments. The dynamic updating of the length of the RCL during the first stage led to the final length of the RCL which is then used throughout the second stage of VNS-GRAMPS.

2.3.3 Variable Neighbourhood Search Algorithm

As part of the local search used in GRASP in both stages, we adopt a commonly used meta-heuristic, namely, the VNS, originally developed by Mladenović and Hansen (1997). This simple but powerful technique avoids entrapment in a local optimum by adopting a systematic change of neighbourhoods within a local search algorithm. VNS searches in increasing distant neighbourhoods of the current incumbent solution and moves to a new solution if and only if an improvement is obtained instead of following a normal trajectory as tabu search or simulated annealing.

A basic VNS starts by generating an initial solution, x, a set of defining neighbourhood structures $N_k,\ k = \{1,\ldots,k_{max}\}$, adopting a local search and a stopping criterion. The main steps of the VNS include the use of the shaking, the local search and whether or not to move to the next neighbourhood. In the shaking step, the algorithm randomly generates a new solution, x', in the k^{th} neighbourhood of the solution x $(x' \in N_k(x))$, and then it applies a local search to find the corresponding local optimum solution, x'', around x'. In the movement step, if the solution is improved (i.e., $f(x'') < f(x)$), the new solution becomes the current incumbent solution ($x = x''$) and the search returns to the first neighbourhood ($k = 1, N_1$), otherwise, the next larger neighbourhood is explored i.e., ($k = k+1$). The pseudo code of the VNS was given in Algorithm 5.

Algorithm 5 VNS Algorithm

Start:
Initialization: Generate an initial solution x, set the neighbourhood structures $N_k, k = \{1,\ldots,k_{max}\}$ and define a stopping criterion;
Main Step: Repeat (Until the maximum # non-improvement iterations is reached);
(1) k = 1;
(2) Repeat (Until $k = k_{max}$);
(a) *Shaking:* Generate a random solution, x' from the k^{th} neighbourhood of x $(x' \in N_k(x))$;
(b) *Local search:* Find the local optimum solution, x'', around x' by using the related local search heuristic;
(c) *Movement:* If $f(x'') < f(x)$, then change the incumbent solution by setting $x = x''$ and $k = 1$ and go to step 2. Otherwise, explore the next larger neighbourhood by setting $k = k+1$ and go back to step 2.
End.

The initial solution constructed by using ISSCA and REDSA heuristics is used as the initial solution of VNS in the local search step of VNS-GRAMPS. In the shaking step, a random solution is created around the respective neighbourhood of the initial solution with respect to some procedures defined in Section 2.3.3.2. Then, the local optimum of the randomly generated solution is found using the respective local search operators. Finally, the best solution obtained with VNS is updated in the improvement step. These steps are repeated until the maximum number of iterations is reached, and the best solution obtained is compared against the global best solution within the GRASP framework.

2.3.3.1 Neighbourhood Structures

In this study, we used five neighbourhood structures which are described in the next subsection as part of the local searches. As the order in which these will be used in the VNS is critical, at this stage we do not explicitly denote the k^{th} neighbourhood $N_k(x), k = 1, \ldots, K_{max} = 5$. This will be defined at the end of the next section where an empirical experiment is conducted. The 5 neighbourhood structures are the following:

One-node interchange,

Two-node interchange,

Two-shift type 1,

Two-shift type 2 and

Two-one node interchange.

2.3.3.2 Local Search Operators

In this section, we explained the five local search operators which are based on the five neighbourhood structures mentioned earlier, namely, one-node interchange, two-node interchange, two-shift type 1, two-shift type 2 and two-one node interchange.

One-node Interchange: In this local search, a customer is selected from a route, and then it is checked for insertion to another route.

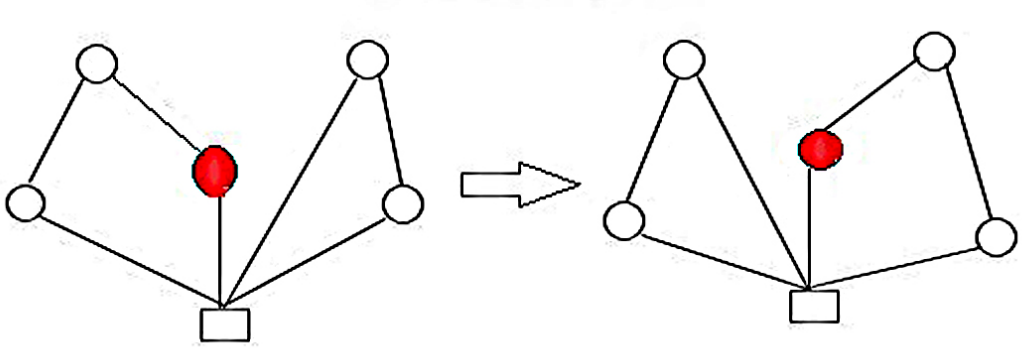


Figure 2. One-node interchange

In the shaking step of VNS algorithm, the random solution x' is generated from the current incumbent solution x as follows: First, a random route is selected with respect to the controlled randomized function given in Equation (28), and a random customer

is chosen from the selected route with respect to the controlled randomized function given in Equation (29). The controlled randomness feature of the Equations 28 and 29, which is obtained by multiplying the desired value with a uniform random number, was obtained according to the method used in the study of Kocatürk and Özpeynirci (2014). In equation (28), we give importance to the route having the highest cost, and we focus on the customer that will result in the highest saving when it is moved to another route. Then, the selected customer, say j, is checked for insertion in the routes that are in the ρ neighbourhoods of j. These are the routes of the closest ρ customers or depots of j. This is an important neighbourhood reduction scheme that cuts the unnecessary computations of the non-promising moves. This aspect is strongly demonstrated in Salhi and Sari (1997) and recently in Sze et al. (2016). If one of the closest nodes of j is a depot, then all routes originating from that depot are also included into the neighbouring routes of j. While checking customer j for insertion to a neighbouring route r, we adopt the best improvement strategy (i.e., we move customer j to the best position in the route r, if there exists a cost saving). We apply the customer interchanges if it is a feasible movement. If the algorithm could not find a feasible position that provides a cost saving in route r, the search continues with the next neighbouring route.

In the local search step, we check all routes by starting from customer j that is in the first position of the first route. In other words, this is an exhaustive application of the shaking in $N_1(.)$ as explained above. As before, we check customer j for insertion to the ρ neighbour routes of j. If it finds a feasible move having a cost saving, it moves the customer j, and continues to search with the next customer.

$$r\_rand_r = (f_r + \alpha_r * Distance_r) * U(0,1), \forall r \in \{1, \ldots, r_{max}\} \tag{28}$$

where f_r is the fixed cost, α_r is the variable cost, and $Distance_r$ is the total travel distance of the route r, $U(0,1)$ is a uniform random number in the interval $[0,1]$, and r_{max} is the total number of routes.

$$pos\_rand_i = (D_{i-1,i} + D_{i,i+1} - D_{i-1,i+1}) * U(0,1), \forall i \in I_r = \{1, \ldots, |I_r|\} \qquad (29)$$

where $D_{i,i+1}$ is the distance between customers i and $i+1$, $U(0,1)$ is a uniform random number in the interval $[0,1]$, and I_r is the set of customers in the route r. Additionally, the customers with indices 0 and $|I_r|+1$ represent the originated depot of the route r.

Two-Node Interchange: In this local search, a random customer is selected from a route, and then the selected customer is swapped with an another customer from a different route. This application is applied to all customers and to all routes and the interchange that results in the overall best saving is selected.

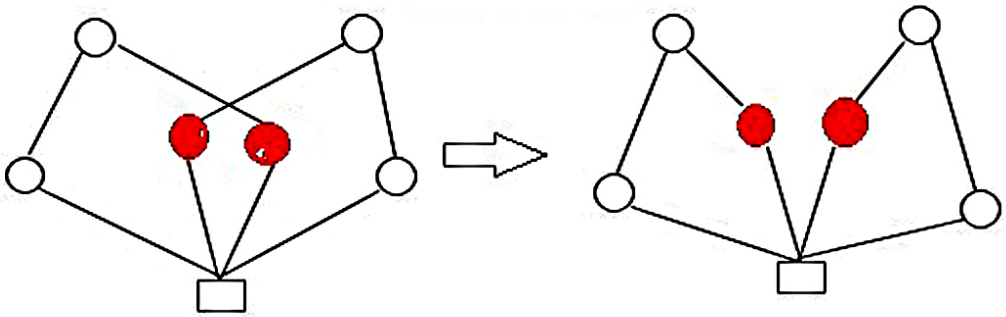


Figure 3. Two-node interchange

In the shaking step which represents one move of the local search, is defined as follows. A random solution x' is generated from the current incumbent solution x as follows: First, a random route is selected with respect to Equation (28), and then a customer is chosen randomly from the selected route with respect to Equation (29). Then, the selected customer, say j_1, is checked for insertion to its ρ neighbour routes. The first possible neighbour route that is not violating the problem constraints is selected as the second route to which the customer change is applied. A customer, say j_2, is picked randomly from the selected route with Equation (29), and these customers are swapped.

Two-Shift Type 1: In this local search, two adjacent customers are selected from a

route, and then the selected customers are checked for insertion to another route in the same order. This is applied for all customers and all routes and the one that yields the best saving is selected. The random solution x' is generated from the current incumbent solution x as follows in the shaking step: First, a random route is selected with respect to Equation (28), and then two adjacent customers are chosen randomly from the selected route with respect to Equation (30). The controlled randomness feature of the Equation 30 was also obtained according to the method used in Kocatürk and Özpeynirci (2014). In this equation, we focus on the customers having the minimum distance between them and having the maximum saving when they are moved from the route by minimizing the obtained ratio. Then, the selected customers, say j_1 and j_2, are checked for insertion to the ρ neighbouring routes.

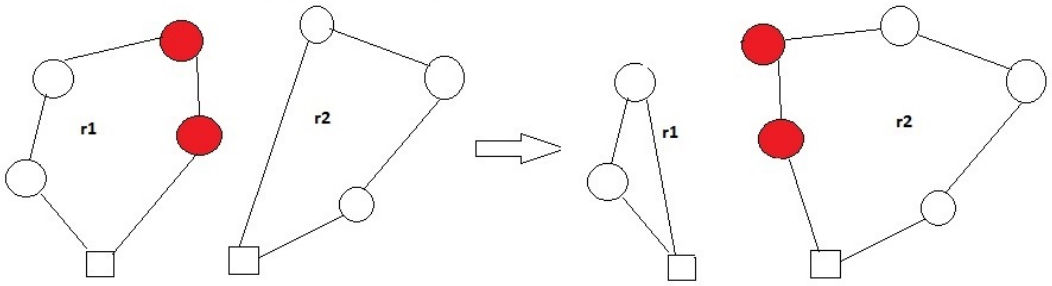


Figure 4. Two-shift type1

$$p\_rand2_i = \frac{D_{i,i+1}}{(D_{i-1,i} + D_{i,i+1} + D_{i+1,i+2} - D_{i-1,i+2})} * U(0,1), \forall i \in I_r = \{1, \ldots, |I_r|\} \tag{30}$$

Two-Shift Type 2: In this local search, two adjacent customers are selected from a random route, and these customers are inserted into two different routes. This is applied for all customers and all routes and the best one is chosen. For the shaking step, the random solution x' is generated from the current incumbent solution x as follows. First, a random route is selected, and two adjacent customers are chosen randomly by using Equation (30) from the selected route. Then, the selected customers, say j_1 and j_2, are checked for insertion to the routes that are in the ρ neighbourhoods. The insertion order of the customers is determined randomly.

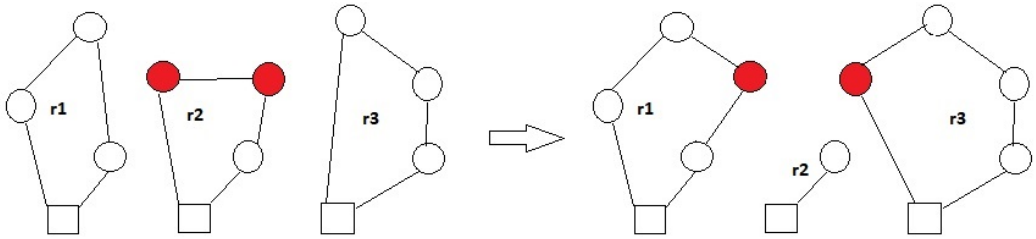


Figure 5. Two-shift type2

Two-One Node Interchange: In this local search, two adjacent customers are selected from a route, and then these customers are moved to a different route without changing the order of the customers. Moreover, a customer from the target route is moved into the origin route of the adjacent customers. This application is performed for all customers and all routes and the one producing the overall best saving is chosen. The random solution x' is generated from the current incumbent solution x as follows in the shaking step: First, a random route is selected with respect to Equation (28), and then two adjacent customers are chosen randomly from the selected route by using Equation (30). Then, the selected customers, say j_1 and j_2, are checked for insertion to the ρ neighbouring routes. The customers are moved into a route that is not violating the problem constraints, and a customer from the target route is moved into the originated route of customers j_1 and j_2.

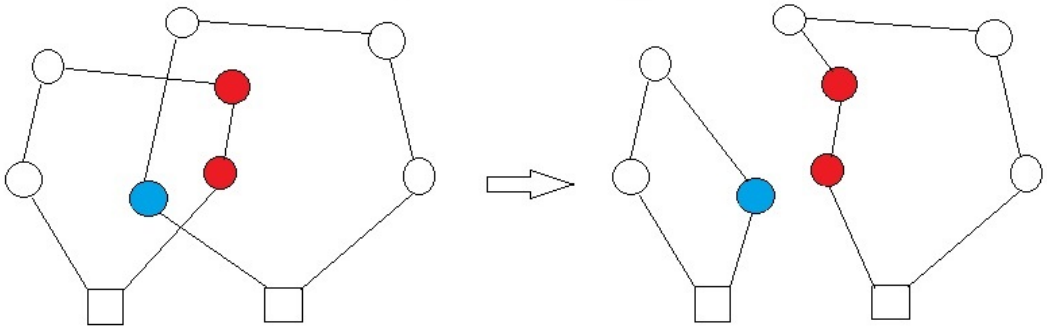


Figure 6. Two-One node interchange

2.3.3.3 Performance measures of the local searches

We determined the application order of the local search operators in the local search step of VNS algorithm with respect to the following two performance criteria. We used the success ratio and the average improvement to assess the performance of the operators.

i) $Success\_Ratio_l = \frac{\#\ iterations\ the\ operator\ l\ improved\ the\ solution}{\#\ iterations\ the\ operator\ l\ entered} \times 100,$

ii) $Average\_Improvement_l = \frac{Total\ cost\ improvement\ of\ the\ operator\ l\ (\%)}{\#\ iterations\ the\ operator\ l\ improved\ the\ solution}.$

We calculated the Effective Improvement Ratio (EIR) of a local search operator l given in Equation (31) in order to calculate the improvement performance of the operator in an iteration in which the operator improved the solution. To the best of our knowledge, this is the first time this performance measure is used to compare the local search operators. EIR represents the average percentage cost improvement achieved per percentage of iterations that improved the cost of solution. Local search operators that make more cost reductions in fewer iterations are given priority by using this metric. In other words, it is aimed to use local search operators with higher cost improvement ratio in one iteration in the first place in VNS. We reported the EIF of the local search operators for 12 problems selected from 26 MDHFVRP benchmark instances (Salhi and Sari, 1997) in Table 1, and we selected one or two instances in which VNS-GRAMPS performed better among the instances having the same number of customers. For example, an EIR of 5.26 for the Two-shift Type 2 local search operator indicates that it achieved an average cost improvement of 5.26 percent per percent of iterations that improved the cost of solution for the problem instance #1 in Table 1. According to the average effective improvement ratios, the Two-shift Type 2 obtained the best value as 1.52%, the Two-shift Type 1 obtained 0.63%, the Two-one Node Interchange got 0.23%, the One-node Interchange a 0.12% and finally the Two-node Interchange obtained 0.10% only. We then re-order the neighbourhood structures as follows:

Two-shift Type 2 as N_1,

Two-shift Type 1 as N_2,

Two-one Node Interchange as N_3,

One- node interchange as N_4,

Two-node Interchange as N_5.

$$Effective\_Improvement\_Ratio_l = \frac{Average\_Improvement}{Success\_Ratio} \tag{31}$$

Table 1. Effective improvement ratios of local search operators

Instance No	One-node Interchange	Two-node Interchange	Two-shift Type 1	Two-shift Type 2	Two-one Node Interchange
1	0.50	0.10	3.38	5.26	1.50
3	0.04	0.05	0.27	1.69	0.10
5	0.15	0.12	1.04	3.42	0.08
6	0.09	0.12	1.14	1.76	0.22
8	0.06	0.24	0.24	0.42	0.14
9	0.05	0.07	0.25	1.34	0.09
11	0.03	0.02	0.10	1.78	0.06
12	0.02	0.02	0.06	0.15	0.03
15	0.22	0.18	0.32	0.83	0.31
18	0.10	0.15	0.47	0.38	0.11
23	0.11	0.09	0.25	1.08	0.11
24	0.03	0.04	0.04	0.09	0.02
Average:	0.12	0.10	0.63	1.52	0.23

2.3.4 Seed Improvement Algorithm (SIA)

In the second stage of the VNS-GRAMPS meta-heuristic, the initial seed solution is generated based on the best solution found in the first stage. This is performed as follows:

- All routes of the best solution are ordered in decreasing value of the cost,

- The GRASP stage starts with the first route in the list (i.e., the one having the largest cost) and then iterates for all routes,

- An adjacent route of the one having the next greater route index and originating from the same depot is selected,

- Two customers, one from each route of the two selected routes, that maximize the function g_{ij} in Equation (32), are chosen as the remaining customers,

- The other customers from these two routes are deleted,

- The other routes of the solution remain unchanged.

Selection Criterion

We develop the following criterion to select the two customers one from each of the two routes.

$$g_{ij} = \frac{\theta_{ij}}{\max \theta_{ij}} + \frac{(D_{0i} + D_{0j})}{2 * \max D_{ij}} \tag{32}$$

This function is used to define new initial seed solutions in order to search for different regions of the solution space. Here, θ_{ij} represents the angle between customers i and j, and D_{ij} refers to the distance between these two customers. In Equation (32), D_{0i} and D_{0j} represent the distances between the customers i, j and the depot of origin. The first part of the function given in Equation (32) is used to decrease the route overlaps. Our aim here is to select two customers forming the largest angle with the depot as the new seeds in the area spanned by the selected routes. The second part of the function prevents the selection of any new seeds that happen to be too close to the depot.

2.4 Details of Decision Support System

In this section, the three main modules of the proposed DSS are explained in detail. These sections are the Model Controller module, which explains the working mechanism of DSS, the Database module where the customer, route and solution information used in DSS are stored in a systematic way, and the User Interface module, where the user interface details are explained. The proposed DSS was developed on a computer with Intel Core i3-3110M CPU @ 2.40 GHz, 8 GB RAM and Windows 10 Prof. 64-bit operating system using C# programming language on Visual Studio Community software. Computational experiments to measure the performance of DSS were also performed on this computer.

2.4.1 Model Controller Module

Model controller module consists of the processes including visual introduction and solution of VRPs. The user can construct the initial seed solution by assigning some customers to some vehicles based on his/her experiences, predictions or wishes

after selecting a defined problem from the database of the DSS and can visualise the customers and depots on the map (Fig. 7). Then, the DSS proposes a near-optimal solution comprising all customers by starting with the generated initial seed solution and using the hybrid unified VNS-GRAMPS meta-heuristic. DSS also provides an option of automatically assignment of customers to the users. If the user makes a mistake while assigning customers, DSS pops up a warning screen and helps the user to make an accurate assignment.

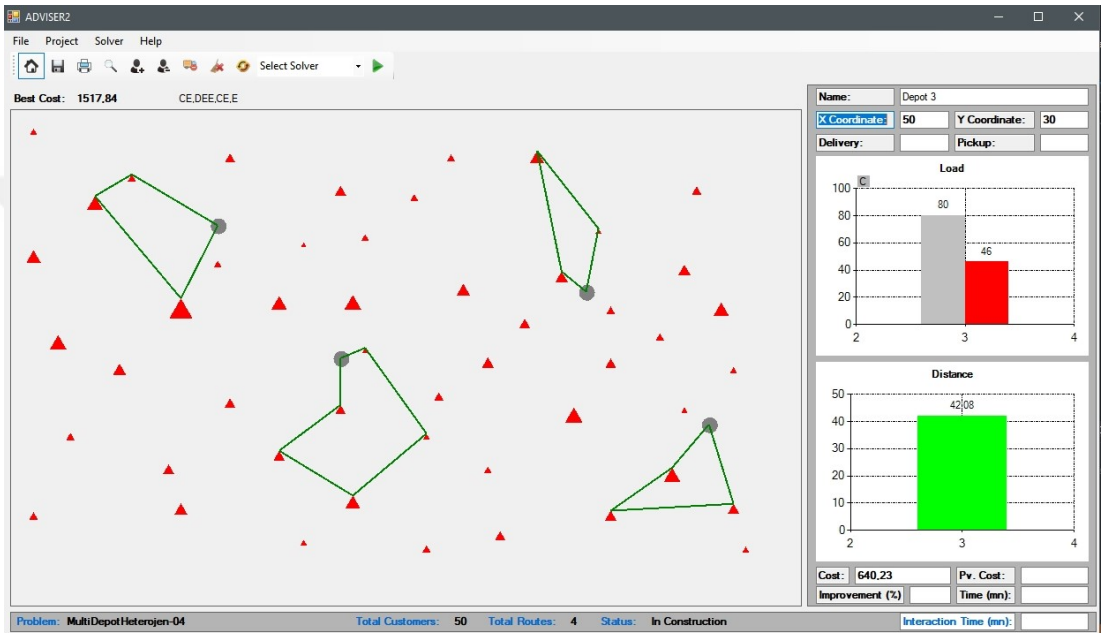


Figure 7. Main screen of DSS visualising the customers, depots, vehicles and routes

Information about the problem opened is displayed on the dashboard at the right-hand side of Fig. 7 instantly. While the mouse cursor is hovering on the map on the left, when the red triangles representing the customers are hovered over, the coordinate and demand information of that customer, and when the grey circles representing the depots are hovered, the coordinate information of that depot is displayed at the top of the dashboard instantly. When the mouse cursor is on a customer, the capacity and the travelled distance of the route where that customer is located, and when it comes over the depot, the capacity and distance information of all routes departing from that depot are instantly displayed in the middle of the dashboard with bar graphs. In this way, the user can be sure that the capacity and distance constraints for the route under consideration are not exceeded. When the capacity and distance graphics of more than

one route are displayed in the middle of the dashboard, the relevant route is drawn on the map in bolt faced when the mouse is hovered over a column in the capacity graphic so that it can be seen which graphic belongs to which route on the map. At the bottom of the dashboard, the cost of the solution obtained, the solution time, the cost of the solution before the user interaction, the percentage improvement rate on the solution after the interaction, and the interaction time are shown. At the bottom of the main screen given in Fig. 7, name, total number of customers, total number of routes and solution status information of the problem are displayed.

2.4.2 Database Module

The database of the DSS provides the data used in model controller and user interface modules. The data about the problem structure constitutes the input for the database. These inputs are imported to the database with text files. The database of the DSS consists of 8 related tables given in Fig. 8. These are "Problem" table that stores the basic parameters about VRP, "Customer" table that stores the types, demands and coordinates of the customers, "Depot" table that stores the coordinates of the depots, "Vehicle" table that stores the types, load and distance capacities, fix and variable costs of the vehicles, "SeedMaster" table that stores the seed solutions, "SolutionMaster" table that stores the generated solutions, "Route" table that stores the information of routes in the generated solutions and seed solutions, and "RouteDetail" table that stores the customer identities and positions of customers for the routes stored in "Route" table, respectively.

2.4.3 User Interface

Proposing a solution to VRPs by focusing on the user interaction is the basis of the presented DSS. Therefore, a user interactive solution procedure, that incorporates the user in the solution process with the user interface, was developed to solve various VRPs. In the top of the application window, there exist main menu bar and toolbar. Each customer is represented as a triangle icon on the map and its size is directly proportional to the demand of customer. This property guides the user to assign a

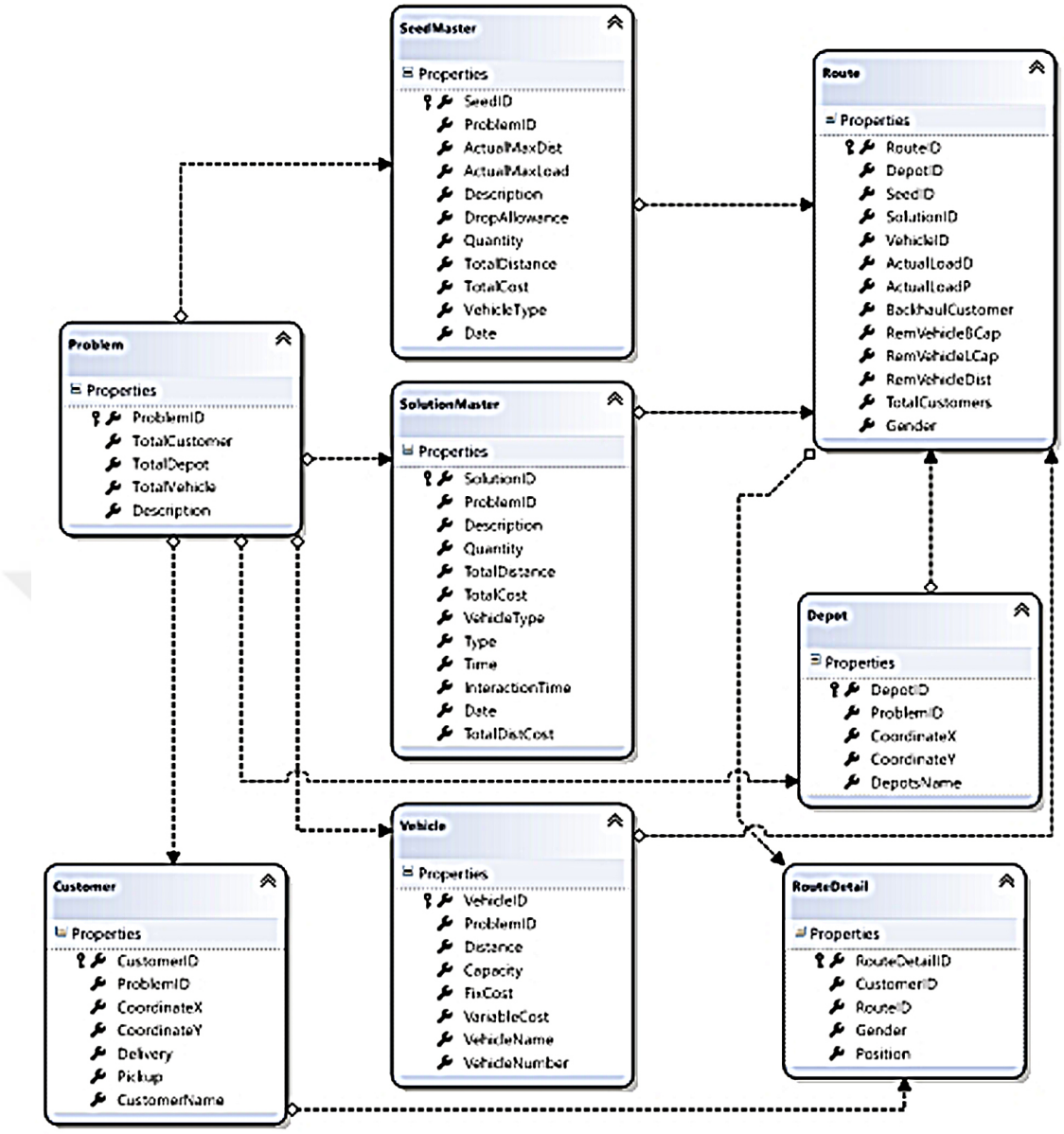


Figure 8. Database tables of the DSS

vehicle with higher capacity to the regions with higher demands. Interaction with the user is also provided with tools where the user can make changes on the solution such as adding/deleting customers, adding/deleting routes, changing vehicle types. In addition, the user is guided by warnings in cases where the problem constraints are exceeded and insufficient information is entered.

Menu Bar: There exists File, Project, Solver and Help menus in the menu bar of the DSS. There are Open Project, Import Problem and Exit menus in File menu. There are Open Project, Close Project and Save Seed Always menus in Project menu. There is Options menu that adjusts the algorithm parameters in Project menu. When the user

selects the Open Project in Project menu, the table listing the saved problems in the database is opened (Fig. 9). The selected problem in that list is visualised on the map after the user has selected the problem. The user can analyse and solve the problem by generating the initial seed solution on the map by using interactive tools or Initial Seed Solution Construction Algorithm (ISSCA) defined in Kocatürk et al. (2021). If the user clicks the depot to add a vehicle, Add Vehicle pop-up screen listing the possible vehicle types for the selected problem is opened. The user can determine the capacity and/or distance constraints, fix and variable costs by selecting the vehicle type.

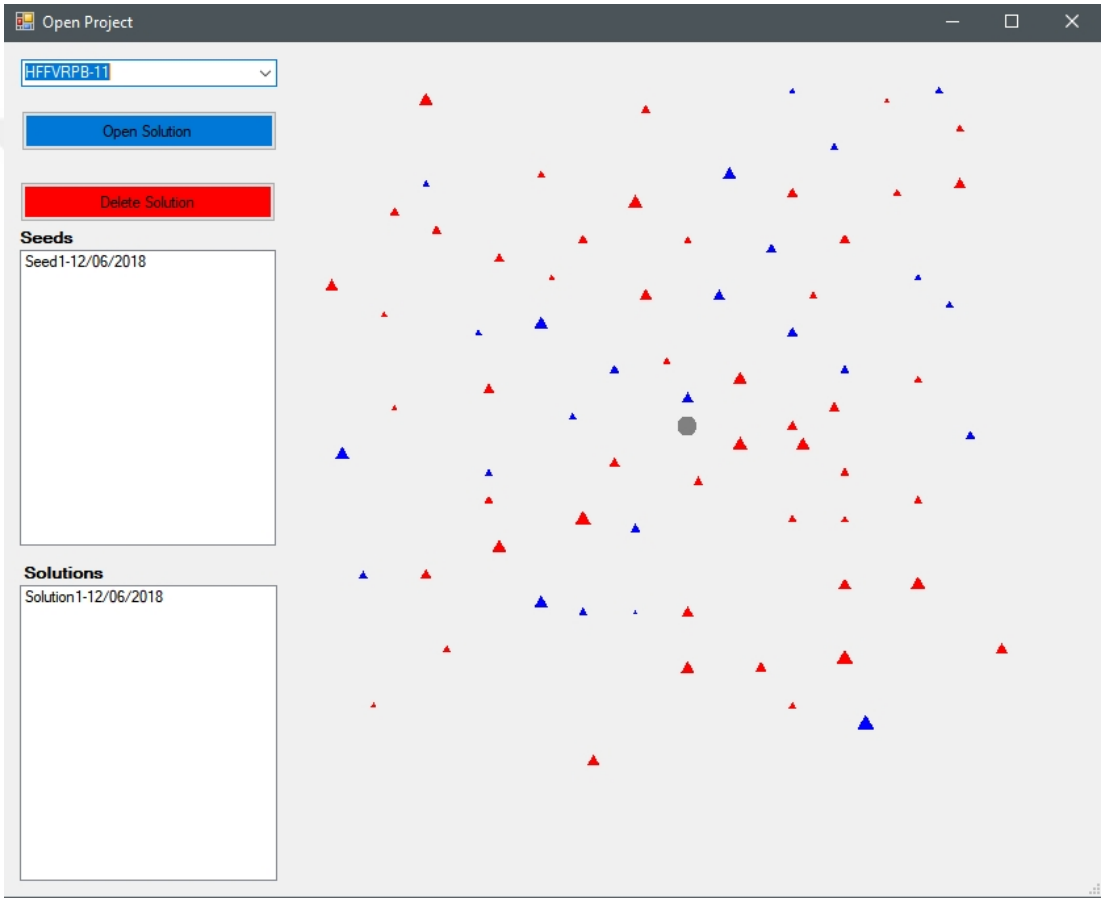


Figure 9. Open Project window of the DSS to open new problem, solution or initial seed solution

Toolbar: The toolbar of the DSS consists of three parts. The first part includes Save Solution and Print Solution buttons. The user can access and print the detailed report of the solution proposed by DSS after clicking the Print Solution button (see Fig. 10). The second part includes a drop-down list of meta-heuristics that can solve the selected VRP variant, and the button to run the selected meta-heuristic. The third

part comprises buttons playing an important role in interacting with the user. The user selects only one button each time and this selected button enables the interaction between the user and DSS. The function of each button in the third part of toolbar is explained as follows:

Default: The user generates the initial seed solution by using Default button. The user firstly clicks the depot and selects a vehicle type. After selecting the vehicle, the user clicks the customers to assign them to the route, respectively. Finally, the user must click to the depot again in order the close the route. If the user violates the capacity constraint or the preceding constraint while assigning a customer to the route, a pop-up screen warns the user.

Add Customer: Add Customer button is designed to allow the user to assign any customer he/she wants to any route, and Add Customer property becomes active after Add Customer button is selected. First, the user has to select the customer to be assigned, and then he/she has to select two adjacent customers or a customer and an adjacent depot in order to add the first selected customer between two other selected nodes. If the user makes a mistake during this adding process, a pop-up screen warns the user.

Delete Customer: Delete Customer button is designed to allow the user to delete any customer from its route. The user must select the customer to be deleted after selecting Delete Customer button.

Delete Route: The user can delete any route by selecting any customer from the route to be deleted after selecting Delete Route button.

Delete All Routes: All routes on the map are deleted after the user selects Delete All Routes button. Default button is activated after all routes are deleted.

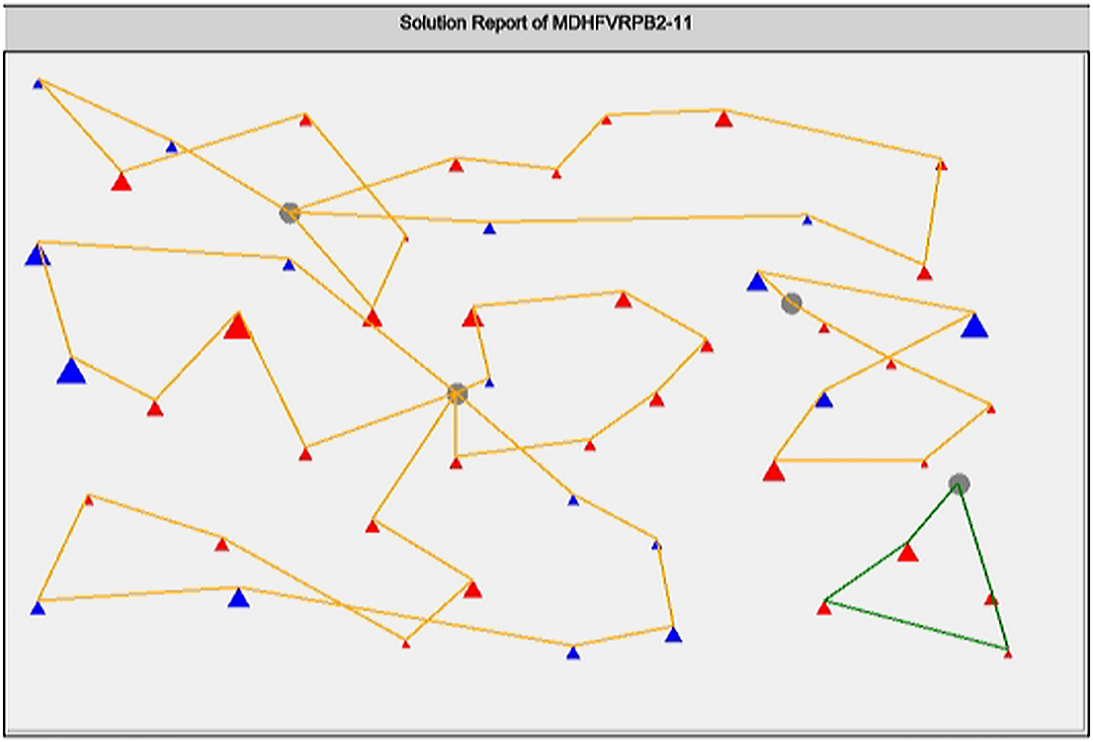
Change Vehicle: The type of the vehicle of a route can be changed by using Change Vehicle button. The user can change the type of the vehicle of a route on the window opened after clicking on any customer assigned to the route.

Auto Seed: Auto Seed button was designed to generate initial seed solution for the selected problem automatically by using ISSCA. The user can construct initial seed solution by using the Auto Seed button if he/she does not have prior information about the problem. After, the initial seed solution has constructed automatically, the user can

modify the solution by using the interactive tools and find a solution.

If the user clicks the run button after selecting the meta-heuristic, DSS generates a solution for the selected problem by using the selected meta-heuristic. The generated solution is displayed on the map and the user can accept the proposed solution or improve the solution by using the interactive tools of the DSS defined in this section.

The user can select a problem or an initial seed solution / final solution of a selected problem saved in the database from the lists appeared when he/she clicks the Open Project button. If the user selects a problem from this list to solve, the selected problem is displayed on the map in Open Project window as seen in Fig. 9. The selected problem is visualised on the map of DSS after the user clicks Open Solution button. Finally, the user can analyse the capacity and distance information from the dashboard located in the right-hand side of DSS. We aim to help the user to generate better initial seed solutions by utilizing this information dashboard and achieve better solutions for the selected problems.



Total Cost: 1331,48

Total Customers: 50

Route ID	Route Type	Vehicle Type / Capacity	Depot	Customers	Delivery Load	Pickup Load
1	B	C / 80	D 2	C 6 -> C 18 -> C 14 -> C 25 -> C 13 -> C 4	77	60
2	B	E / 112	D 2	C 27 -> C 32 -> C 11 -> C 38 -> C 5 -> C 12 -> C 46	111	5
3	B	C / 80	D 2	C 48 -> C 8 -> C 26 -> C 23 -> C 24 -> C 43 -> C 7 -> C 31 -> C 28 -> C 22 -> C 1	73	70
4	B	C / 80	D 1	C 47 -> C 17 -> C 42 -> C 41 -> C 40 -> C 19	68	16
5	B	D / 96	D 1	C 44 -> C 15 -> C 45 -> C 33 -> C 39 -> C 30 -> C 10 -> C 37	92	14
6	B	C / 80	D 3	C 9 -> C 50 -> C 21 -> C 29 -> C 2 -> C 16 -> C 34 -> C 49	65	59
7	L	C / 80	D 4	C 35 -> C 36 -> C 3 -> C 20	67	0

Figure 10. Constructed solution report after clicking Print Solution button

CHAPTER 3: EXPERIMENTAL RESULTS

In order to ensure the generalization capability of the Decision Support System (DSS) and to compare the performance of the Hybrid unified Variable Neighborhood Search (VNS) based Greedy Randomized Adaptive Memory Programming Search (GRAMPS), VNS-GRAMPS for short, five different single-depot and multi-depot Vehicle Routing Problem (VRP) instances were solved. The benchmark instances of these problems were summarized in the following sub-sections. We also introduced three data sets of Multi-depot Heterogeneous VRP with Backhauls (MDHFVRPB) which were defined firstly in this thesis for bench-marking purpose. Then, we reported the CPLEX results of the mathematical models for MDHFVRPB and the results of VNS-GRAMPS and a basic VNS meta-heuristcs on the problem data sets of MDHFVRPB. Finally, the computational performance of the VNS-GRAMPS meta-heuristic embedded in the DSS was examined on 5 different single-depot and multi-depot VRP instances. Hybrid unified VNS-GRAMPS and VNS algorithms are coded and the developed DSS was designed using C# programming language and run on a PC having Intel(R) Core(TM) i3-3110M CPU @ 2.40 GHz CPU, 8 GB RAM and Windows 10 Prof. 64 bit operating system.

3.1 Problem Instances

3.1.1 Problem Instances of HFFVRP

Benchmark instances (Set 1) containing 8 different problems defined by Taillard (1999) and problem instances consisting of 8 different problems (Set 2) used by Gendreau et al. (1999) were solved with the VNS-GRAMPS. Except for the vehicle combinations of the problems in Set 1 and Set 2, the number of customers, vehicle capacity, fixed and variable costs are the same. The number of customers, fixed and variable costs and capacities of the vehicles belonging to the problem instances in Set 1 and Set 2 were given in Tables 2 and 3, respectively. In these tables, N, Q, f, v, n represent the total number of customers, capacity, fixed cost, variable cost and the

number of available vehicles for each vehicle type, respectively.

Table 2. Problem instances of HFFVRP in Set 1

Type	Property	Test Problems 13	14	15	16	17	18	19	20
	N	50	50	50	50	75	75	100	100
A	Q	20	120	50	40	50	20	100	60
	f	20	100	100	100	25	10	500	100
	v	1	1	1	1	1	1	1	1
	n	4	4	4	2	4	4	4	6
B	Q	30	160	100	80	120	50	200	140
	f	15	1500	250	200	80	35	1200	300
	v	1.1	1.1	1.6	1.6	1.2	1.3	1.4	1.7
	n	2	2	3	4	4	4	3	4
C	Q	40	300	160	140	200	100	300	200
	f	50	3500	450	400	150	100	2100	500
	v	1.2	1.4	2	2.1	1.5	1.9	1.7	2
	n	4	1	2	3	2	2	3	3
D	Q	70				350	150		
	f	120				320	180		
	v	1.7				1.8	2.4		
	n	4				1	2		
E	Q	120					250		
	f	225					400		
	v	2.5					2.9		
	n	2					1		
F	Q	200					400		
	f	400					800		
	v	3.2					3.2		
	n	1					1		

3.1.2 Problem Instances of FSMVRPB

For Fleet Size and Mix VRP with Backhauls (FSMVRPB), 36 problem instances defined by Salhi et al. (2013) were solved with the VNS-GRAMPS. The number of customers of the problems, fixed and variable costs and capacities of the vehicles were given in Table 4. In this table, N represents the total number of customers, L the total number of delivery customers, B the total number of pick-up customers, Q1 the capacity of the first vehicle type, f1 the fixed cost of the first vehicle type. The remaining columns represent the capacity and fixed costs of other vehicles in the related problems. In this problem, the contribution of the use of heterogeneous vehicle fleet to the objective function was calculated by adding only the fixed costs, the variable costs of the vehicles were taken as 1 for all vehicles.

Table 3. Problem instances of HFFVRP in Set 2

Type	Property	Test Problems 13	14	15	16	17	18	19	20
	N	50	50	50	50	75	75	100	100
A	Q	20	120	50	40	50	20	100	60
	f	20	100	100	100	25	10	500	100
	v	1	1	1	1	1	1	1	1
	n	0	3	7	8	0	0	5	6
B	Q	30	160	100	80	120	50	200	140
	f	15	1500	250	200	80	35	1200	300
	v	1.1	1.1	1.6	1.6	1.2	1.3	1.4	1.7
	n	1	4	3	1	2	2	5	1
C	Q	40	300	160	140	200	100	300	200
	f	50	3500	450	400	150	100	2100	500
	v	1.2	1.4	2	2.1	1.5	1.9	1.7	2
	n	1	0	1	3	3	4	0	5
D	Q	70				350	150		
	f	120				320	180		
	v	1.7				1.8	2.4		
	n	7				1	0		
E	Q	120					250		
	f	225					400		
	v	2.5					2.9		
	n	2					2		
F	Q	200					400		
	f	400					800		
	v	3.2					3.2		
	n	1					1		

3.1.3 Problem Instances of MVRP

VNS-GRAMPS was tested using two problem sets (Set 1 and Set 2) consisting of 33 Multi-depot VRP (MVRP) bench-marking instances (2 to 9 depots, 50 to 360 customers) from the literature. The first set of problems, Set 1, consists of a total of 23 problems including seven small-scale MVRP problems (P1–P7) contributed by Christofides and Eilon (1969), four medium-sized problems (P8–P11) contributed by Gillett and Johnson (1976), and twelve medium-sized problems (P12-23) defined by Chao et al. (1993). The problems in Set 2 include 10 problems (4 and 6 depots, 48 to 288 customers) that were used by Vidal et al. (2014) and defined by Cordeau et al. (1997).

Table 4. Problem instances of FSMVRPB

No	N	L	B	Q1	f1	Q2	f2	Q3	f3	Q4	f4	Q5	f5	Q6	f6
HWS1	20	10	10	20	20	30	35	40	50	70	120	120	225		
HWS2	20	13	7	20	20	30	35	40	50	70	120	120	225		
HWS3	20	16	4	20	20	30	35	40	50	70	120	120	225		
HWS4	20	10	10	60	1000	80	1500	150	3000						
HWS5	20	13	7	60	1000	80	1500	150	3000						
HWS6	20	16	4	60	1000	80	1500	150	3000						
HWS7	20	10	10	20	20	30	35	40	50	70	120	120	225		
HWS8	20	13	7	20	20	30	35	40	50	70	120	120	225		
HWS9	20	16	4	20	20	30	35	40	50	70	120	120	225		
HWS10	20	10	10	60	1000	80	1500	150	3000						
HWS11	20	13	7	60	1000	80	1500	150	3000						
HWS12	20	16	4	60	1000	80	1500	150	3000						
HWS13	50	25	25	20	20	30	35	40	50	70	120	120	225	200	400
HWS14	50	33	17	20	20	30	35	40	50	70	120	120	225	200	400
HWS15	50	40	10	20	20	30	35	40	50	70	120	120	225	200	400
HWS16	50	25	25	120	1000	160	1500	300	3500						
HWS17	50	33	17	120	1000	160	1500	300	3500						
HWS18	50	40	10	120	1000	160	1500	300	3500						
HWS19	50	25	25	50	100	100	250	160	450						
HWS20	50	33	17	50	100	100	250	160	450						
HWS21	50	40	10	50	100	100	250	160	450						
HWS22	50	25	25	40	100	80	200	140	400						
HWS23	50	33	17	40	100	80	200	140	400						
HWS24	50	40	10	40	100	80	200	140	400						
HWS25	75	37	38	50	25	120	80	200	150	350	320				
HWS26	75	50	25	50	25	120	80	200	150	350	320				
HWS27	75	60	15	50	25	120	80	200	150	350	320				
HWS28	75	37	38	20	10	50	35	100	100	150	180	250	400	400	800
HWS29	75	50	25	20	10	50	35	100	100	150	180	250	400	400	800
HWS30	75	60	15	20	10	50	35	100	100	150	180	250	400	400	800
HWS31	100	50	50	100	500	200	1200	300	2100						
HWS32	100	66	34	100	500	200	1200	300	2100						
HWS33	100	80	20	100	500	200	1200	300	2100						
HWS34	100	50	50	60	100	140	300	200	500						
HWS35	100	66	34	60	100	140	300	200	500						
HWS36	100	80	20	60	100	140	300	200	500						

3.1.4 Problem Instances of MVRPB

In order to evaluate the performance of the VNS-GRAMPS for MVRP with Backhauls (MVRPB), 33 benchmark instances proposed by Salhi and Nagy (1999) were solved. These problem instances were obtained by applying the backhaul method to 11 problem instances (2 to 5 depots, 50 to 249 customers) suggested by Gillett and Johnson (1976) for MVRP. With this method, every second, fourth and tenth customer was defined as backhaul customer for each problem instance and three new problem instances with 50%, 25% and 10% backhaul customer ratios were created, respectively.

3.1.5 Problem Instances of MDHFVRP

The performance of the proposed VNS-GRAMPS was tested on two problem sets (Set 1, Set 2) of Multi-depot Heterogeneous VRP (MDHFVRP). Set 1 includes 26 problem instances generated in Salhi and Sari (1997) by adding 5 different vehicle type information to the widely used MVRP data set. The problems in Set 1 consist of 50 to 360 customers and 2 to 9 depots. The problems in Set 2 include 10 problems that were used by Vidal et al. (2014) and defined by Cordeau et al. (1997). Heterogeneous vehicle information of the problems in Set 2 was also created according to the method proposed by Salhi and Sari (1997). According to this method, a total of 5 vehicle types, $k = 1,\ldots,5$, were created for each problem, and the capacities of these vehicles were calculated using the original capacity Q of the problem according to the equation $Q_k = (0.4 + 0.2 * k) * Q$, the variable cost is calculated according to the equation $\alpha_k = 0.7 + 0.1 * k$, and the fixed cost is calculated according to the equation $f_k = 70 + 10 * k$. The information of the problems in Set 1 and Set 2 are given in Tables 5 and 6, respectively.

3.1.6 Problem Instances of MDHFVRPB

As the the problem instances of MDHFVRPB are not available in the literature we generated three scenarios based on the data sets of related routing problems that are widely used in the literature.

(i) Scenario 1- In this scenario, we produce a new MDHFVRPB data set by combining the depot information of the MDHFVRP used in Salhi et al. (2014) with

Table 5. Problem instances of MDHFVRP in Set 1

No	N	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
1	55	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
2	85	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
3	85	3	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
4	50	4	∞	48	80	0.8	64	90	0.9	80	100	1	96	110	1.1	112	120	1.2
5	50	4	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
6	75	5	∞	84	80	0.8	112	90	0.9	140	100	1	168	110	1.1	196	120	1.2
7	100	2	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
8	100	2	∞	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
9	100	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
10	100	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
11	249	2	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
12	249	3	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
13	249	4	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
14	249	5	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
15	80	2	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
16	80	2	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
17	80	2	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
18	160	4	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
19	160	4	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
20	160	4	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
21	240	6	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
22	240	6	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
23	240	6	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
24	360	9	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
25	360	9	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
26	360	9	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2

Table 6. Problem instances of MDHFVRP in Set 2

No	N	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
1	48	4	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
2	96	4	480	117	80	0.8	156	90	0.9	195	100	1	234	110	1.1	273	120	1.2
3	144	4	460	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
4	192	4	440	111	80	0.8	148	90	0.9	185	100	1	222	110	1.1	259	120	1.2
5	240	4	420	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
6	288	4	400	105	80	0.8	140	90	0.9	175	100	1	210	110	1.1	245	120	1.2
7	72	6	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
8	144	6	475	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
9	216	6	450	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
10	288	6	425	102	80	0.8	136	90	0.9	170	100	1	204	110	1.1	238	120	1.2

the problem instances of the FSMVRPB (Salhi et al., 2013). The derived problem instances are originally based on the problem set of heterogeneous vehicle fleet VRP of Golden et al. (1984) and the problem set of VRP with backhauls of Toth and Vigo (1997).

This new MDHFVRPB data set is given in Table 7. In this table, N represents the total number of customers, L the total number of linehaul customers, B the total number of backhaul customers, and m the total number of depots. The remaining three columns represent capacity (Q), fixed cost (f) and variable cost (v) of each vehicle type, respectively.

(ii) Scenario 2- Here, the data set is generated by defining linehaul and backhaul customers for each of the 26 MDHFVRP problem instances used in Salhi and Sari (1997). Three linehaul/backhaul percentages as 50/50, 67/33 and 80/20 are generated for each problem instance. The customer locations, demands, heterogeneous vehicle fleet are all kept unchanged. As in the literature, these new MDHFVRPB instances follow also the same pattern by using the first customer of every two, three and five customers as backhaul customer for the 50/50, 67/33 and 80/20 linehaul/backhaul percentages, respectively. These new data set is given in Tables 8-11. The table format is the same as the one adopted for Table 7.

(iii) Scenario 3- This third data set is derived by defining linehaul and backhaul customers for the 10 MDHFVRP problem instances given in Vidal et al. (2014) which are based on the MVRP problem instances of Cordeau et al. (1997). Similarly to the earlier two scenarios, three linehaul/backhaul percentages as 50/50, 67/33 and 80/20 are generated here for each problem instance while the customer locations, the demands and the heterogeneous vehicle fleet are kept the same. This MDHFVRPB new data set is given in Table 12. The same table format used in the previous scenarios is also adopted here.

Table 7. MDHFVRPB Problem Instances for Scenario 1

No	N	L	B	m	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5	Q6	f6	v6
1	20	10	10	2	20	20	17.2	30	35	21.2	40	50	25.2	70	120	26.2	120	225	28.4			
2	20	13	7	2	20	20	17.2	30	35	21.2	40	50	25.2	70	120	26.2	120	225	28.4			
3	20	16	4	2	20	20	17.2	30	35	21.2	40	50	25.2	70	120	26.2	120	225	28.4			
4	20	10	10	2	60	1000	17.2	80	1500	21.2	150	3000	28.4									
5	20	13	7	2	60	1000	17.2	80	1500	21.2	150	3000	28.4									
6	20	16	4	2	60	1000	17.2	80	1500	21.2	150	3000	28.4									
7	20	10	10	2	20	20	17.2	30	35	21.2	40	50	25.2	70	120	26.2	120	225	28.4			
8	20	13	7	2	20	20	17.2	30	35	21.2	40	50	25.2	70	120	26.2	120	225	28.4			
9	20	16	4	2	20	20	17.2	30	35	21.2	40	50	25.2	70	120	26.2	120	225	28.4			
10	20	10	10	2	60	1000	17.2	80	1500	21.2	150	3000	28.4									
11	20	13	7	2	60	1000	17.2	80	1500	21.2	150	3000	28.4									
12	20	16	4	2	60	1000	17.2	80	1500	21.2	150	3000	28.4									
13	50	25	25	3	20	20	10.8	30	35	17.2	40	50	21.2	70	120	25.2	120	225	26.2	200	400	28.4
14	50	33	17	3	20	20	10.8	30	35	17.2	40	50	21.2	70	120	25.2	120	225	26.2	200	400	28.4
15	50	40	10	3	20	20	10.8	30	35	17.2	40	50	21.2	70	120	25.2	120	225	26.2	200	400	28.4
16	50	25	25	3	120	1000	17.2	160	1500	21.2	300	3500	28.4									
17	50	33	17	3	120	1000	17.2	160	1500	21.2	300	3500	28.4									
18	50	40	10	3	120	1000	17.2	160	1500	21.2	300	3500	28.4									
19	50	25	25	4	50	100	17.2	100	250	21.2	160	450	28.4									
20	50	33	17	4	50	100	17.2	100	250	21.2	160	450	28.4									
21	50	40	10	4	50	100	17.2	100	250	21.2	160	450	28.4									
22	50	25	25	4	40	100	17.2	80	200	21.2	140	400	28.4									
23	50	33	17	4	40	100	17.2	80	200	21.2	140	400	28.4									
24	50	40	10	4	40	100	17.2	80	200	21.2	140	400	28.4									
25	75	37	38	5	50	25	17.2	120	80	21.2	200	150	26.2	350	320	28.4						
26	75	50	25	5	50	25	17.2	120	80	21.2	200	150	26.2	350	320	28.4						
27	75	60	15	5	50	25	17.2	120	80	21.2	200	150	26.2	350	320	28.4						
28	75	37	38	5	20	10	10.8	50	35	17.2	100	100	21.2	150	180	25.2	250	400	26.2	400	800	28.4
29	75	50	25	5	20	10	10.8	50	35	17.2	100	100	21.2	150	180	25.2	250	400	26.2	400	800	28.4
30	75	60	15	5	20	10	10.8	50	35	17.2	100	100	21.2	150	180	25.2	250	400	26.2	400	800	28.4
31	100	50	50	4	100	500	17.2	200	1200	21.2	300	2100	28.4									
32	100	66	34	4	100	500	17.2	200	1200	21.2	300	2100	28.4									
33	100	80	20	4	100	500	17.2	200	1200	21.2	300	2100	28.4									
34	100	50	50	3	60	100	17.2	140	300	21.2	200	500	28.4									
35	100	66	34	3	60	100	17.2	140	300	21.2	200	500	28.4									
36	100	80	20	3	60	100	17.2	140	300	21.2	200	500	28.4									

Table 8. MDHFVRPB Problem Instances for Scenario 2

No	N	L	B	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
1	55	27	28	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
2	55	36	19	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
3	55	44	11	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
4	85	42	43	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
5	85	56	29	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
6	85	68	17	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
7	85	42	43	3	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
8	85	56	29	3	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
9	85	68	17	3	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
10	50	25	25	4	∞	48	80	0.8	64	90	0.9	80	100	1	96	110	1.1	112	120	1.2
11	50	33	17	4	∞	48	80	0.8	64	90	0.9	80	100	1	96	110	1.1	112	120	1.2
12	50	40	10	4	∞	48	80	0.8	64	90	0.9	80	100	1	96	110	1.1	112	120	1.2
13	50	25	25	4	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
14	50	33	17	4	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
15	50	40	10	4	∞	96	80	0.8	128	90	0.9	160	100	1	192	110	1.1	224	120	1.2
16	75	37	38	5	∞	84	80	0.8	112	90	0.9	140	100	1	168	110	1.1	196	120	1.2
17	75	50	25	5	∞	84	80	0.8	112	90	0.9	140	100	1	168	110	1.1	196	120	1.2
18	75	60	15	5	∞	84	80	0.8	112	90	0.9	140	100	1	168	110	1.1	196	120	1.2
19	100	50	50	2	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
20	100	66	34	2	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2

Table 9. MDHFVRPB Problem Instances for Scenario 2 continued

No	N	L	B	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
21	100	80	20	2	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
22	100	50	50	2	∞	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
23	100	66	34	2	∞	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
24	100	80	20	2	∞	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
25	100	50	50	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
26	100	66	34	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
27	100	80	20	3	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
28	100	50	50	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
29	100	66	34	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
30	100	80	20	4	∞	60	80	0.8	80	90	0.9	100	100	1	120	110	1.1	140	120	1.2
31	249	124	125	2	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
32	249	166	83	2	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
33	249	199	50	2	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
34	249	124	125	3	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
35	249	166	83	3	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
36	249	199	50	3	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
37	249	124	125	4	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
38	249	166	83	4	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
39	249	199	50	4	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2

Table 10. MDHFVRPB Problem Instances for Scenario 2 continued

No	N	L	B	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
40	249	124	125	5	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
41	249	166	83	5	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
42	249	199	50	5	310	300	80	0.8	400	90	0.9	500	100	1	600	110	1.1	700	120	1.2
43	80	40	40	2	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
44	80	53	27	2	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
45	80	64	16	2	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
46	80	40	40	2	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
47	80	53	27	2	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
48	80	64	16	2	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
49	80	40	40	2	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
50	80	53	27	2	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
51	80	64	16	2	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
52	160	80	80	4	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
53	160	106	54	4	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
54	160	128	32	4	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
55	160	80	80	4	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
56	160	106	54	4	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
57	160	128	32	4	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
58	160	80	80	4	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
59	160	106	54	4	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
60	160	128	32	4	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2

Table 11. MDHFVRPB Problem Instances for Scenario 2 continued

No	N	L	B	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
61	240	120	120	6	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
62	240	160	80	6	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
63	240	192	48	6	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
64	240	120	120	6	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
65	240	160	80	6	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
66	240	192	48	6	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
67	240	120	120	6	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
68	240	160	80	6	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
69	240	192	48	6	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
70	360	180	180	9	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
71	360	240	120	9	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
72	360	288	72	9	∞	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
73	360	180	180	9	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
74	360	240	120	9	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
75	360	288	72	9	200	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
76	360	180	180	9	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
77	360	240	120	9	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2
78	360	288	72	9	180	36	80	0.8	48	90	0.9	60	100	1	72	110	1.1	84	120	1.2

Table 12. MDHFVRPB Problem Instances for Scenario 3

No	N	L	B	m	TD	Q1	f1	v1	Q2	f2	v2	Q3	f3	v3	Q4	f4	v4	Q5	f5	v5
1	48	24	24	4	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
2	48	32	16	4	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
3	48	38	10	4	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
4	96	48	48	4	480	117	80	0.8	156	90	0.9	195	100	1	234	110	1.1	273	120	1.2
5	96	64	32	4	480	117	80	0.8	156	90	0.9	195	100	1	234	110	1.1	273	120	1.2
6	96	76	20	4	480	117	80	0.8	156	90	0.9	195	100	1	234	110	1.1	273	120	1.2
7	144	72	72	4	460	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
8	144	96	48	4	460	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
9	144	115	29	4	460	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
10	192	96	96	4	440	111	80	0.8	148	90	0.9	185	100	1	222	110	1.1	259	120	1.2
11	192	128	64	4	440	111	80	0.8	148	90	0.9	185	100	1	222	110	1.1	259	120	1.2
12	192	153	39	4	440	111	80	0.8	148	90	0.9	185	100	1	222	110	1.1	259	120	1.2
13	240	120	120	4	420	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
14	240	160	80	4	420	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
15	240	192	48	4	420	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
16	288	144	144	4	400	105	80	0.8	140	90	0.9	175	100	1	210	110	1.1	245	120	1.2
17	288	192	96	4	400	105	80	0.8	140	90	0.9	175	100	1	210	110	1.1	245	120	1.2
18	288	230	58	4	400	105	80	0.8	140	90	0.9	175	100	1	210	110	1.1	245	120	1.2
19	72	36	36	6	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
20	72	48	24	6	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
21	72	57	15	6	500	120	80	0.8	160	90	0.9	200	100	1	240	110	1.1	280	120	1.2
22	144	72	72	6	475	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
23	144	96	48	6	475	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
24	144	115	29	6	475	114	80	0.8	152	90	0.9	190	100	1	228	110	1.1	266	120	1.2
25	216	108	108	6	450	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
26	216	144	72	6	450	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
27	216	172	44	6	450	108	80	0.8	144	90	0.9	180	100	1	216	110	1.1	252	120	1.2
28	288	144	144	6	425	102	80	0.8	136	90	0.9	170	100	1	204	110	1.1	238	120	1.2
29	288	192	96	6	425	102	80	0.8	136	90	0.9	170	100	1	204	110	1.1	238	120	1.2
30	288	230	58	6	425	102	80	0.8	136	90	0.9	170	100	1	204	110	1.1	238	120	1.2

3.2 Computational Experiments

3.2.1 Computational Performance of VNS-GRAMPS on MDHFVRPB

In this section, the performance of the two mathematical models of MDHFVRPB were compared and followed by the results and the analysis of the hybrid unified VNS-GRAMPS and basic VNS algorithms.

3.2.1.1 Performance of the two mathematical models

In this section CPLEX results for MDHFVRPB were reported. The proposed basic and restricted models were modelled by using GAMS 23.9.4 and solved with IBM ILOG CPLEX 12.4.0.1 solver with default settings. The experiments were run on a computer having Intel (R) Core (TM) i5-2310 CPU @ 2.90 GHz CPU, 8 GB RAM and Windows 7 Prof. 64 bit operating system. We conducted 3 hours CPU time limit to solve the basic and restricted models.

(i) Case of Scenario 1- The CPLEX results for MDHFVRPB problem instances of scenario 1 are reported in Table 13. The CPU time of CPLEX was reported as seconds in CPU columns of the table. We found the optimal solutions of 10 out of 12 instances with 20 customers with both models. The optimal solutions of the remaining 2 instances with 20 customers can also be obtained if the CPU time limit is relaxed or different node search options in CPLEX such as depth-first or breadth-first are adopted instead. In general, we can note that of those 10 instances the reduced model requires relatively shorter CPU times and in the remaining 26 instances, where optimality cannot be guaranteed, it yields relatively tighter Lower Bounds (LB) except for two cases, namely, instances #12 and #31.

(ii) Case of Scenario 2- The CPLEX results for MDHFVRPB problem instances of scenario 2 are reported in Tables 14 and 15. The CPU time of CPLEX was reported as seconds in CPU columns of the tables. The basic model found tighter LB values for 12 out of 78 problem instances only, and lower UB values for 27 problems. However, the basic model found lower UB values for 5 out of 9 problems with 360 customers, while the restricted model obtained a lower UB value for the problem #77 only. On

Table 13. MDHFVRPB CPLEX Results for Scenario 1

Problem Information					CPLEX Solt. (Basic)			CPLEX Solt. (Restr.)		
No	**N**	**L**	**B**	**m**	**UB**	**LB**	**CPU**	**UB**	**LB**	**CPU**
1	20	10	10	2	**8584.87**	**8584.87**	635	**8584.87**	**8584.87**	**82**
2	20	13	7	2	**8412.76**	**8412.76**	73	**8412.76**	**8412.76**	**19**
3	20	16	4	2	**8604.8**	**8604.8**	5092	**8604.8**	**8604.8**	**941**
4	20	10	10	2	**9991.1**	**9991.1**	15	**9991.1**	**9991.1**	**4**
5	20	13	7	2	**11191.66**	**11191.66**	618	**11191.66**	**11191.66**	**61**
6	20	16	4	2	**11729.72**	11421.02	10833	**11729.72**	**11514.78**	10801
7	20	10	10	2	**8301.97**	**8301.97**	225	**8301.97**	**8301.97**	**27**
8	20	13	7	2	**8472.11**	**8472.11**	158	**8472.11**	**8472.11**	**42**
9	20	16	4	2	**8309.15**	**8309.15**	642	**8309.15**	**8309.15**	**76**
10	20	10	10	2	**9859.35**	**9859.35**	15	**9859.35**	**9859.35**	4
11	20	13	7	2	**11051.13**	**11051.13**	42	**11051.13**	**11051.13**	40
12	20	16	4	2	**11848.11**	**11609.68**	10801	**11848.11**	11604.88	10808
13	50	25	25	3	**15167**	11152.1	10804	15506.95	**11967.13**	10824
14	50	33	17	3	14869.5	11515.62	10805	**14457.46**	**12444.99**	10821
15	50	40	10	3	**15226.64**	11797.81	10801	15964.77	**12917.5**	10834
16	50	25	25	3	14919.58	13362.35	10801	**14658.83**	**13409.08**	10801
17	50	33	17	3	15460.52	14153.42	10801	**15427.15**	**14185.28**	10802
18	50	40	10	3	**16067.5**	14911.92	10801	16329.97	**14949.96**	10801
19	50	25	25	4	**11619.1**	10835.09	10802	11691.6	**10901.7**	10801
20	50	33	17	4	**12194.92**	11174.03	10801	12600.08	**11316.03**	10801
21	50	40	10	4	15200.09	11359.27	10804	**13561.53**	**11546.17**	10804
22	50	25	25	4	13078.65	11455.71	10803	**12440.13**	**11585.4**	10801
23	50	33	17	4	**12972.93**	11772.94	10802	13121.07	**11989.22**	10801
24	50	40	10	4	**13882.03**	11993.12	10813	14573.47	**12373.18**	10803
25	75	37	38	5	**43207.03**	12364.72	10802	64054.62	**13005.62**	10803
26	75	50	25	5	67120.56	12346.75	10803	**22245.36**	**12578.69**	10802
27	75	60	15	5	34268.06	12442.08	10802	**23567.46**	**12711.42**	10802
28	75	37	38	5	34002.2	11764.6	10804	**28893.98**	**12170.95**	10803
29	75	50	25	5	40039.87	11906.7	10803	**35236.64**	**12117.03**	10803
30	75	60	15	5	**36914.88**	12026.39	10803	84174.95	**12933.87**	10803
31	100	50	50	4	**22019.57**	**17687.43**	10802	22331.23	17662.38	10802
32	100	66	34	4	24402.28	18719.58	10802	**22168.34**	**18772.6**	10802
33	100	80	20	4	**23781.69**	19888.16	10803	26517.18	**19935.94**	10803
34	100	50	50	3	**21666.42**	16075.16	10802	52873.14	**16152.47**	10802
35	100	66	34	3	54898.89	16437.38	10802	**45792.06**	**16585.22**	10802
36	100	80	20	3	70885.71	17097.26	10802	**30331.06**	**17252.16**	10802

the other hand, the restricted model found tighter LB values for 8 of the problems with 360 customers, while the basic model found one only, namely, problem #71. While the basic model found lower UB values for only 17 out of 48 problem instances with 100 and fewer customers, it only found tighter LB values for 6 instances. As a result, the restricted model was able to find tighter LB values for both small and large sized problems. The basic model performed better at obtaining lower UB values for large sized problems while lower UB values for small sized problems are found by the reduced model.

(iii) Case of Scenario 3- The CPLEX results for MDHFVRPB problem instances of scenario 3 are reported in Table 16. The CPU time of CPLEX was reported as seconds in CPU columns of the table. The basic model obtained lower UB values for only 8

Table 14. MDHFVRPB CPLEX Results for Scenario 2

Problem Information						CPLEX Solt. (Basic)			CPLEX Solt. (Restr.)		
No	**N**	**L**	**B**	**m**	**TD**	**UB**	**LB**	**CPU**	**UB**	**LB**	**CPU**
1	55	27	28	4	∞	**3042.09**	**840.14**	10801	NF	839.66	10802
2	55	36	19	4	∞	4375.98	971.91	10803	**4351.09**	**972.81**	10803
3	55	44	11	4	∞	**4699.69**	1112.88	10801	5018.94	**1121.06**	10802
4	85	42	43	3	∞	NF	1228.26	10801	NF	**1242.52**	10801
5	85	56	29	3	∞	6289.60	1474.88	10802	**5680.93**	**1476.95**	10802
6	85	68	17	3	∞	7024.46	1696.82	10803	**5807.52**	**1707.70**	10802
7	85	42	43	3	∞	NF	898.43	10801	NF	**904.93**	10801
8	85	56	29	3	∞	**7071.45**	1055.25	10802	7393.68	**1056.44**	10802
9	85	68	17	3	∞	**4992.75**	1188.92	10802	6041.05	**1195.13**	10802
10	50	25	25	4	∞	NF	946.39	10806	**3447.81**	**951.42**	**5225**
11	50	33	17	4	∞	4415.31	1091.84	10803	**3725.38**	**1102.63**	10801
12	50	40	10	4	∞	3792.42	1203.57	10804	**2448.02**	**1217.03**	10801
13	50	25	25	4	∞	NF	681.98	10804	**5047.55**	**692.61**	10801
14	50	33	17	4	∞	**4484.57**	737.16	10805	5273.26	**749.55**	10802
15	50	40	10	4	∞	**5261.67**	788.84	10808	6157.13	**796.48**	10812
16	75	37	38	5	∞	NF	739.92	10806	NF	**773.34**	10802
17	75	50	25	5	∞	**6722.27**	1104.63	10810	7093.88	**1110.03**	10802
18	75	60	15	5	∞	9871.38	808.05	10810	**7732.58**	**1198.16**	10804
19	100	50	50	2	∞	NF	1118.61	10805	NF	**1184.35**	10801
20	100	66	34	2	∞	10258.20	1617.58	10811	**4681.64**	**1630.19**	10802
21	100	80	20	2	∞	9072.59	1822.14	10809	**4795.68**	**1836.20**	10803
22	100	50	50	2	∞	NF	**681.85**	10806	NF	640.05	10801
23	100	66	34	2	∞	12127.40	1080.56	10805	**5078.68**	**1101.70**	10801
24	100	80	20	2	∞	13731.41	1174.83	10804	**7264.42**	**1179.78**	10801
25	100	50	50	3	∞	NF	1051.04	10805	NF	**1087.12**	10802
26	100	66	34	3	∞	**10585.77**	**1566.77**	10809	12055.48	1560.54	10802
27	100	80	20	3	∞	13949.65	1725.85	10809	**6289.22**	**1761.92**	10802
28	100	50	50	4	∞	NF	1024.70	10807	NF	**1090.80**	10802
29	100	66	34	4	∞	7860.52	1553.42	10811	**6409.48**	**1563.01**	10803
30	100	80	20	4	∞	6425.82	1749.12	10811	**5951.27**	**1756.86**	10811
31	249	124	125	2	310	NF	4098.82	10823	NF	**4176.58**	10808
32	249	166	83	2	310	50068.22	4421.97	10832	**36266.06**	**4545.91**	10811
33	249	199	50	2	310	57463.39	4953.16	10835	**23872.02**	**5109.86**	10810
34	249	124	125	3	310	NF	2207.67	10850	NF	**3956.85**	10811
35	249	166	83	3	310	49549.93	2669.48	10910	**37802.66**	**4303.90**	10816
36	249	199	50	3	310	**28850.25**	4703.06	10817	33045.28	**4748.22**	10814
37	249	124	125	4	310	NF	3841.45	10817	NF	**3854.86**	10815
38	249	166	83	4	310	**50797.46**	4129.53	10821	51082.38	**4156.53**	10821
39	249	199	50	4	310	**56917.59**	3125.27	10878	57925.44	**4542.36**	10820

out of 30 problem instances, and 6 of them are small sized problems with 144 or fewer customers. In addition, the basic model was able to find tighter LB values for only 5 out of 30 problems, 3 of which are small sized problems. The restricted model behaves much better here where it obtained lower UB and tighter LB values for large and small sized problem instances.

3.2.1.2 Performance of the VNS-GRAMPS vs a basic VNS

The problem instances were also solved by a basic VNS meta-heuristic implementation. This was performed in order to compare the performance of our VNS-GRAMPS meta-heuristic. In the initialization step of the VNS algorithm, the initial

Table 15. MDHFVRPB CPLEX Results for Scenario 2 continued

Problem Information						CPLEX Solt. (Basic)			CPLEX Solt. (Restr.)		
No	**N**	**L**	**B**	**m**	**TD**	**UB**	**LB**	**CPU**	**UB**	**LB**	**CPU**
40	249	124	125	5	310	NF	2057.02	10824	NF	**2057.12**	10819
41	249	166	83	5	310	51238.93	2452.86	10826	**50671.50**	**2452.86**	10826
42	249	199	50	5	310	61349.91	3033.10	10827	**58203.43**	**4524.18**	10826
43	80	40	40	2	∞	5586.96	925.52	10801	NF	**966.80**	10801
44	80	53	27	2	∞	**4863.88**	1223.13	10801	7852.32	**1826.95**	10801
45	80	64	16	2	∞	11989.74	1719.69	10801	**11101.78**	**1722.40**	10801
46	80	40	40	2	200	1695.10	1562.97	10801	**1695.10**	**1565.52**	10801
47	80	53	27	2	200	2258.46	2056.51	10804	**2227.32**	**2060.53**	10801
48	80	64	16	2	200	**2147.20**	1822.50	10804	2348.81	**1896.39**	10801
49	80	40	40	2	180	**1732.48**	1664.45	10805	**1732.48**	**1694.02**	10801
50	80	53	27	2	180	**2467.20**	2163.42	10802	2474.27	**2171.40**	10801
51	80	64	16	2	180	**2220.13**	1826.33	10802	2381.91	**1953.17**	10801
52	160	80	80	4	∞	NF	**1794.07**	10806	NF	1467.32	10806
53	160	106	54	4	∞	29336.36	**1858.60**	10813	**18409.91**	1720.22	10808
54	160	128	32	4	∞	**37027.98**	**3330.79**	10810	42138.31	3241.01	10809
55	160	80	80	4	200	5877.55	2756.48	10808	**3734.24**	**2939.16**	10808
56	160	106	54	4	200	6913.90	3683.85	10809	**5340.57**	**3931.49**	10809
57	160	128	32	4	200	11838.31	3494.64	10808	**10144.85**	**3617.92**	10808
58	160	80	80	4	180	**4972.77**	2766.42	10809	NF	**3080.43**	10806
59	160	106	54	4	180	**4823.18**	3713.06	10809	5148.10	**4148.91**	10808
60	160	128	32	4	180	**5396.20**	3502.56	10809	8910.03	**3695.85**	10808
61	240	120	120	6	∞	NF	**1898.29**	10823	NF	1801.00	10821
62	240	160	80	6	∞	**39988.20**	**2387.68**	11016	67873.54	2101.03	10829
63	240	192	48	6	∞	**43581.99**	**2553.35**	11125	81233.28	2519.11	10829
64	240	120	120	6	200	NF	4244.34	10823	NF	**4255.21**	10821
65	240	160	80	6	200	NF	**5731.72**	10823	**8505.03**	5712.11	10827
66	240	192	48	6	200	18860.28	5196.15	10831	**14641.67**	**5223.25**	10827
67	240	120	120	6	180	NF	**4506.11**	10823	NF	4501.39	10821
68	240	160	80	6	180	NF	6007.27	10824	**7405.87**	**6076.72**	10827
69	240	192	48	6	180	17794.39	5468.80	10830	**13126.48**	**5472.49**	10829
70	360	180	180	9	∞	NF	2427.28	10900	NF	**2585.88**	10891
71	360	240	120	9	∞	NF	**3154.55**	10899	NF	3126.27	10887
72	360	288	72	9	∞	NF	3348.70	10900	NF	**3593.63**	10891
73	360	180	180	9	200	**29102.73**	5997.30	10922	NF	**6303.77**	10890
74	360	240	120	9	200	**41292.83**	7934.50	10923	NF	**8298.08**	10887
75	360	288	72	9	200	**41722.85**	7532.78	10920	62427.04	**7783.37**	10910
76	360	180	180	9	180	**26302.62**	6028.81	10923	NF	**6591.66**	10891
77	360	240	120	9	180	27687.07	8048.13	10921	**12536.30**	**8939.87**	10908
78	360	288	72	9	180	**58001.29**	7957.88	11444	NF	**8002.54**	10887

solution was constructed by using ISSCA and REDSA heuristics. In the shaking step, the random solution around the respective neighbourhood was obtained by using the same procedure given in Section 2.3.3. The five neighbourhoods defined in Section 2.3.3.2 were also used in the local search step of the VNS and applied in the same order, namely, two-shift type 2, two-shift type 1, two-one node interchange, one-node interchange and two-node interchange. The maximum number of iterations was used as the stopping criterion and set to 100. The VNS algorithm was run 10 times for each problem instance and the result of the best solution was reported. The CPU times reported for the VNS algorithm are the average over the 10 runs. Note that, the implementation was designed and aimed to set the total number of iterations to

Table 16. MDHFVRPB CPLEX Results for Scenario 3

Problem Information						CPLEX Solt. (Basic)			CPLEX Solt. (Restr.)		
No	**N**	**L**	**B**	**m**	**TD**	**UB**	**LB**	**CPU**	**UB**	**LB**	**CPU**
1	48	24	24	4	500	4635.30	1019.19	10810	**1143.26**	**1041.97**	10801
2	48	32	16	4	500	**1338.07**	**1080.44**	10803	1412.38	1080.13	10801
3	48	38	10	4	500	**1431.42**	**1068.02**	10803	1449.36	1067.29	10801
4	96	48	48	4	480	NF	796.77	10805	NF	**858.03**	10802
5	96	64	32	4	480	6771.07	**1581.22**	10823	**5678.59**	1579.16	10805
6	96	76	20	4	480	12935.87	1508.07	10810	**10530.64**	**1528.45**	10803
7	144	72	72	4	460	NF	1005.56	10818	NF	**1059.15**	10805
8	144	96	48	4	460	**8673.56**	1043.70	10830	11135.92	**1180.92**	10860
9	144	115	29	4	460	28592.11	2228.84	10826	**23867.27**	**2286.80**	10814
10	192	96	96	4	440	NF	1133.71	10832	NF	**1166.74**	10826
11	192	128	64	4	440	31386.54	1386.55	10851	**13907.28**	**1451.40**	10811
12	192	153	39	4	440	36133.89	1416.14	10850	**35529.90**	**2581.56**	10811
13	240	120	120	4	420	NF	1487.15	10863	NF	**1517.40**	10814
14	240	160	80	4	420	36745.55	1804.19	10880	**18236.92**	**1866.99**	10821
15	240	192	48	4	420	**39200.99**	1975.61	10881	39950.52	**1975.62**	10820
16	288	144	144	4	400	NF	1489.10	10900	NF	**1626.06**	10821
17	288	192	96	4	400	46880.99	**1983.33**	10921	**23238.12**	1967.19	10827
18	288	230	58	4	400	52507.37	2182.00	10899	**52083.42**	**2182.09**	10828
19	72	36	36	6	500	NF	722.73	10808	NF	**793.12**	10802
20	72	48	24	6	500	**9838.64**	1264.94	10813	10609.70	**1284.44**	10804
21	72	57	15	6	500	**7181.57**	1349.38	10810	9694.40	**1374.99**	10802
22	144	72	72	6	475	NF	945.92	10822	NF	**992.01**	10807
23	144	96	48	6	475	23954.15	936.25	10840	**23254.65**	**2126.50**	10813
24	144	115	29	6	475	**26107.44**	1083.33	10836	27093.42	**2226.30**	10809
25	216	108	108	6	450	NF	1132.18	10879	NF	**1173.98**	10817
26	216	144	72	6	450	36450.16	1398.35	10895	**12531.50**	**1489.87**	10822
27	216	172	44	6	450	41672.62	1626.60	10891	**40623.36**	**1626.69**	10824
28	288	144	144	6	425	NF	1644.70	10969	NF	**1644.95**	10832
29	288	192	96	6	425	**48315.11**	**2289.82**	11086	48884.17	2269.65	10841
30	288	230	58	6	425	53231.15	2414.49	11014	**52279.96**	**2414.50**	10841

approximately the same value between VNS-GRAMPS and VNS algorithms in order to make a fair comparison. For this purpose, the outer iteration number of the Reactive Greedy Randomized Adaptive Search Procedure (RGRASP) phase of VNS-GRAMPS was set to 100 and the outer iteration number of the VNS algorithm used in the local search step was restricted to 10. Another strategy would be to set the same overall CPU time for both implementations even though one will use more iterations than the other.

The performances of the algorithms are assessed using the average percent gap values against the LB and UB values obtained by CPLEX. These gaps (in %) are given in the following Equations (1) and (2.)

$$Gap_{LB}(\%) = \frac{Z_{alg} - LB}{LB} \times 100 \tag{1}$$

$$Gap_{UB}(\%) = \frac{Z_{alg} - UB}{UB} \times 100 \tag{2}$$

where Z_{alg} is the cost of the solution found by VNS-GRAMPS or VNS algorithms whereas LB and UB are the lower and upper bound values obtained by CPLEX, respectively.

Results for Scenario 1:

We report the test results for the MDHFVRPB problem instances of scenario 1 in Tables 17 and 18. In this table, N, L, B, m represent the total number of customers (i.e., linehaul and backhaul customers) and depots, respectively. We also give the lower bound (LB) and the upper bound (UB) found by CPLEX. Moreover, the percentage gaps of UB, VNS, VNS-GRAMPS test results with respect to the LB values, and the CPU times for VNS and VNS-GRAMPS in minutes are also given here. We obtain the optimal solutions for the problems #2, #3, #7, #8, #9 and #11 by VNS-GRAMPS. The best solutions were also found for 14 out of 24 problem instances having 50 and more customers by VNS-GRAMPS, and the other 10 out of 24 best solutions were obtained by CPLEX. The lowest average gap against LB is 13.01% for VNS-GRAMPS, and the average gaps against LB for UB and VNS were reported as 39.05% and 18.44%, respectively. The performance of VNS-GRAMPS and VNS was also compared with respect to the average gaps against UB values recorded by CPLEX. The lowest average gap against UB was -9.39% for VNS-GRAMPS and -5.08% for VNS. With regard to CPU times, VNS-GRAMPS obtained the solutions in relatively shorter CPU times for the problem instances with 20 customers, but it took longer for the problems with 50 and more customers. This is because VNS-GRAMPS applies VNS in the local search step at each iteration. Finally, VNS-GRAMPS required 4.34 minutes CPU time on average, whereas VNS, as expected, needed a relatively shorter CPU time of 1.54 minutes only.

The summary results of UB, VNS and VNS-GRAMPS are reported in Table 19. In the case of using the average gap values against LB, the average gap value of UB for the problem instances with 75 customers was relatively very large, 151.35%. The largest average gap values against LB were also obtained by VNS and VNS-GRAMPS for the instances with 75 customers. It is worth mentioning that for the smaller instances, namely, those with 20 and 50 customers, the best average gap values against LB were found by CPLEX, whereas for the larger instances, namely, those with 75 and 100

Table 17. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 1

No	N	L	B	m	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
					LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
1	20	10	10	2	8584.87	**8584.87**	8900.86	8679.40	**0.00**	3.68	1.10	3.68	**1.10**	0.90	0.43
2	20	13	7	2	8412.76	**8412.76**	8662.66	**8412.76**	**0.00**	2.97	**0.00**	2.97	**0.00**	0.89	0.41
3	20	16	4	2	8604.80	**8604.80**	8758.70	**8604.80**	**0.00**	1.79	**0.00**	1.79	**0.00**	0.90	0.46
4	20	10	10	2	9991.10	**9991.10**	10507.40	10148.72	**0.00**	5.17	1.58	5.17	**1.58**	0.91	0.33
5	20	13	7	2	11191.66	**11191.66**	12176.87	11542.46	**0.00**	8.80	3.13	8.80	**3.13**	0.88	0.35
6	20	16	4	2	11514.78	**11729.72**	13373.36	12595.70	**1.87**	16.14	9.39	14.01	**7.38**	0.89	0.45
7	20	10	10	2	8301.97	**8301.97**	8538.53	**8301.97**	**0.00**	2.85	**0.00**	2.85	**0.00**	0.89	0.37
8	20	13	7	2	8472.11	**8472.11**	8606.85	**8472.11**	**0.00**	1.59	**0.00**	1.59	**0.00**	0.88	0.40
9	20	16	4	2	8309.15	**8309.15**	8753.47	**8309.15**	**0.00**	5.35	**0.00**	5.35	**0.00**	0.89	0.42
10	20	10	10	2	9859.35	**9859.35**	9958.71	9958.71	**0.00**	1.01	1.01	**1.01**	**1.01**	0.90	0.31
11	20	13	7	2	11051.13	**11051.13**	**11051.13**	**11051.13**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.89	0.34
12	20	16	4	2	11609.68	**11848.11**	13558.37	13025.68	**2.05**	16.79	12.20	14.43	**9.94**	0.89	0.36
13	50	25	25	3	11967.13	15167.00	15030.40	**14405.54**	26.74	25.60	**20.38**	-0.90	**-5.02**	1.21	2.03
14	50	33	17	3	12444.99	14457.46	15405.06	**14347.51**	16.17	23.79	**15.29**	6.55	**-0.76**	1.22	2.37
15	50	40	10	3	12917.50	15226.64	14906.61	**14669.36**	17.88	15.40	**13.56**	-2.10	**-3.66**	1.12	2.44
16	50	25	25	3	13409.08	**14658.83**	16224.56	15234.15	**9.32**	21.00	13.61	10.68	**3.92**	1.27	2.59
17	50	33	17	3	14185.28	**15427.15**	16372.70	16127.78	**8.75**	15.42	13.69	6.13	**4.54**	1.28	1.98
18	50	40	10	3	14949.96	**16067.50**	18550.25	17632.06	**7.48**	24.08	17.94	15.45	**9.74**	1.26	2.32
								Average:	**5.01**	10.63	6.83	5.41	**1.83**	**1.00**	1.02

Table 18. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 1 continued

No	N	L	B	m	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
					LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
19	50	25	25	4	10901.70	**11619.10**	13642.17	11946.07	**6.58**	25.14	9.58	17.41	**2.81**	1.11	2.17
20	50	33	17	4	11316.03	**12194.92**	12967.40	12738.71	**7.77**	14.59	12.57	6.33	**4.46**	1.11	2.52
21	50	40	10	4	11546.17	13561.53	13325.15	**13029.77**	17.45	15.41	**12.85**	-1.74	**-3.92**	1.09	2.68
22	50	25	25	4	11585.40	**12440.13**	13752.97	13153.29	**7.38**	18.71	13.53	10.55	**5.73**	1.10	2.38
23	50	33	17	4	11989.22	**12972.93**	15558.41	14005.61	**8.20**	29.77	16.82	19.93	**7.96**	1.05	2.45
24	50	40	10	4	12373.18	**13882.03**	14400.88	14241.72	**12.19**	16.39	15.10	3.74	**2.59**	1.08	2.69
25	75	37	38	5	13005.62	43207.03	16599.23	**15585.50**	232.22	27.63	**19.84**	-61.58	**-63.93**	1.87	6.97
26	75	50	25	5	12578.69	22245.36	15693.50	**15077.26**	76.85	24.76	**19.86**	-29.45	**-32.22**	1.75	6.86
27	75	60	15	5	12711.42	23567.46	16196.18	**15288.42**	85.40	27.41	**20.27**	-31.28	**-35.13**	1.68	7.47
28	75	37	38	5	12170.95	28893.98	16719.13	**15862.35**	137.40	37.37	**30.33**	-42.14	**-45.10**	1.81	7.39
29	75	50	25	5	12117.03	35236.64	17642.97	**16808.32**	190.80	45.60	**38.72**	-49.93	**-52.30**	1.74	7.32
30	75	60	15	5	12933.87	36914.88	17059.68	**16353.55**	185.41	31.90	**26.44**	-53.79	**-55.70**	1.60	8.21
31	100	50	50	4	17687.43	22019.57	21895.78	**21494.95**	24.49	23.79	**21.53**	-0.56	**-2.38**	4.70	15.13
32	100	66	34	4	18772.6	**22168.34**	26500.69	22187.92	**18.09**	41.17	18.19	19.54	**0.09**	4.24	14.62
33	100	80	20	4	19935.94	**23781.69**	24684.82	23816.79	**19.29**	23.82	19.47	3.80	**0.15**	3.28	14.07
34	100	50	50	3	16152.47	21666.42	21146.05	**18915.02**	34.14	30.92	**17.10**	-2.40	**-12.70**	3.02	10.73
35	100	66	34	3	16585.22	45792.06	19421.56	**19399.43**	176.10	17.10	**16.97**	-57.59	**-57.64**	2.78	12.18
36	100	80	20	3	17252.16	30331.06	20879.62	**20060.64**	75.81	21.03	**16.28**	-31.16	**-33.86**	2.35	11.99
								Average:	73.09	26.25	**19.19**	-15.57	**-20.62**	**2.08**	7.66

customers, the best gaps against LB were obtained by VNS-GRAMPS. In the case of using the average gap values against UB, the best gap values were obtained by VNS-GRAMPS for both smaller and larger instances.

Results for Scenario 2:

The results for Scenario 2 are given in Tables 20-23. In these tables, the same format as Table 17 is used except the TD column representing the maximum travel distance constraint. The VNS algorithm was able to find the same or better results than VNS-GRAMPS for only 9 out of 78 problem instances. VNS and VNS-GRAMPS found the same result for 3 of these 9 problems (#13, #46, #49), while CPLEX found the same UB values for problems #46 and #49. When the performances of the VNS and VNS-GRAMPS algorithms were compared according to the gap from the LB values found with CPLEX, an average of 41.90% and 44.58% gap values were obtained for VNS-GRAMPS and VNS, respectively. On the other hand, an average of -57.88% and -57.02% gap values were obtained for VNS-GRAMPS and VNS, respectively, when the performances of the VNS and VNS-GRAMPS algorithms were compared according to the gap from the UB values found with CPLEX. The reason for the average gap values against UB being so close is that there was no UB values for 18 out of 78 problem instances. In terms of CPU times, VNS always found the solutions in less time due to the same reason noted in the earlier section. For instance, VNS-GRAMPS found a solution in 35.11 minutes on average while VNS in 10.26 minutes.

In Table 24, summary performance values of VNS and VNS-GRAMPS algorithms are reported. The results are presented under two categories, namely, the small-sized problems, with 100 or less customers, and the large-sized problems, with 160 or more customers. In the first category, VNS-GRAMPS found better solutions than VNS. VNS-GRAMPS obtained the largest average gap value against LB for instances with 75 customers and the smallest average gap value against LB for those with 55 customers. In brief, VNS-GRAMPS and VNS algorithms achieved an average gap of 25.61% and 28.31% respectively, with respect to LB. When the average gap values from the UB values obtained with CPLEX, VNS-GRAMPS obtained lower gap values than VNS for all cases. While, the lowest gap value against UB was obtained for instances with 75 customers, the largest gap value was obtained for instances with 80

Table 19. Summary of VNS and VNS-GRAMPS Test Results for Scenario 1

N	# Best Solutions			Average Gap wrt LB (%)			Average Gap wrt UB (%)		CPU (min)	
	UB	**VNS**	**VNS-GRAMPS**	**UB**	**VNS**	**VNS-GRAMPS**	**VNS**	**VNS-GRAMPS**	**VNS**	**VNS-GRAMPS**
20	12/12	1/12	6/12	0.33	5.51	2.37	5.14	2.01	0.89	0.39
50	8/12	0/12	4/12	12.16	20.44	14.58	7.67	2.37	1.16	2.39
75	0/6	0/6	6/6	151.35	32.45	25.91	-44.69	-47.40	1.74	7.37
100	2/6	0/6	4/6	57.99	26.30	18.26	-11.40	-17.72	3.40	13.12
			Average:	39.05	18.44	**13.01**	-5.08	**-9.39**	**1.54**	4.34

Table 20. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 2

No	N	L	B	m	TD	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
						LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
1	55	27	28	4	∞	840.14	3042.09	1063.98	**1010.05**	262.09	26.64	**20.22**	-65.02	**-66.80**	1.19	2.27
2	55	36	19	4	∞	972.81	4351.09	1131.35	**1107.42**	347.27	16.30	**13.84**	-74.00	**-74.55**	1.12	2.54
3	55	44	11	4	∞	1121.06	4699.69	1256.17	**1247.05**	319.22	12.05	**11.24**	-73.27	**-73.47**	1.11	2.11
4	85	42	43	3	∞	1242.52	NF	1465.54	**1430.83**	-	17.95	**15.16**	-	-	1.72	2.87
5	85	56	29	3	∞	1476.95	5680.93	1658.54	**1647.06**	284.64	12.29	**11.52**	-70.81	**-71.01**	1.63	3.00
6	85	68	17	3	∞	1707.70	5807.52	1906.47	**1882.55**	240.08	11.64	**10.24**	-67.17	**-67.58**	1.37	5.02
7	85	42	43	3	∞	904.93	NF	1158.82	**1156.99**	-	28.06	**27.85**	-	-	2.35	6.43
8	85	56	29	3	∞	1056.44	7071.45	1318.37	**1272.85**	569.37	24.79	**20.48**	-81.36	**-82.00**	2.04	5.95
9	85	68	17	3	∞	1195.13	4992.75	1444.65	**1421.84**	317.76	20.88	**18.97**	-71.07	**-71.52**	1.92	6.26
10	50	25	25	4	∞	951.42	3447.81	1212.26	**1165.15**	262.39	27.42	**22.46**	-64.84	**-66.21**	1.13	1.71
11	50	33	17	4	∞	1102.63	3725.38	1331.48	**1313.74**	237.86	20.75	**19.15**	-64.26	**-64.74**	1.09	1.78
12	50	40	10	4	∞	1217.03	2448.02	1465.90	**1437.95**	101.15	20.45	**18.15**	-40.12	**-41.26**	1.06	1.84
13	50	25	25	4	∞	692.61	5047.55	**848.48**	**848.48**	628.77	**22.50**	**22.50**	**-83.19**	**-83.19**	1.35	2.22
14	50	33	17	4	∞	749.55	4484.57	951.65	**925.13**	498.30	26.96	**23.42**	-78.78	**-79.37**	1.28	2.38
15	50	40	10	4	∞	796.48	5261.67	973.23	**951.76**	560.62	22.19	**19.50**	-81.50	**-81.91**	1.25	2.20
16	75	37	38	5	∞	773.34	NF	1341.52	**1338.30**	-	73.47	**73.05**	-	-	1.88	5.16
17	75	50	25	5	∞	1110.03	6722.27	1380.84	**1352.42**	505.59	24.40	**21.84**	-79.46	**-79.88**	1.64	3.09
18	75	60	15	5	∞	1198.16	7732.58	1511.64	**1503.50**	545.37	26.16	**25.48**	-80.45	**-80.56**	1.61	5.26
19	100	50	50	2	∞	1184.35	NF	1827.67	**1755.52**	-	54.32	**48.23**	-	-	2.46	7.90
20	100	66	34	2	∞	1630.19	4681.64	1967.41	**1883.18**	187.18	20.69	**15.52**	-57.98	**-59.78**	2.21	6.75
									Average:		25.50	**22.94**	-70.83	**-71.49**	**1.57**	3.84

Table 21. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 2 continued

No	N	L	B	m	TD	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
						LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
21	100	80	20	2	∞	1836.20	4795.68	2161.50	**2134.02**	161.17	17.72	**16.22**	-54.93	**-55.50**	1.88	4.32
22	100	50	50	2	∞	681.85	NF	1281.12	**1249.14**	-	87.89	**83.20**	-	-	6.00	6.83
23	100	66	34	2	∞	1101.70	5078.68	1420.06	**1371.77**	360.99	28.90	**24.51**	-72.04	**-72.99**	5.13	6.50
24	100	80	20	2	∞	1179.78	7264.42	1491.38	**1477.85**	515.74	26.41	**25.26**	-79.47	**-79.66**	3.73	6.47
25	100	50	50	3	∞	1087.12	NF	1760.11	**1731.02**	-	61.91	**59.23**	-	-	2.58	4.58
26	100	66	34	3	∞	1566.77	10585.77	1933.77	**1834.22**	575.64	23.42	**17.07**	-81.73	**-82.67**	2.24	4.74
27	100	80	20	3	∞	1761.92	6289.22	2116.29	**2092.40**	256.95	20.11	**18.76**	-66.35	**-66.73**	1.98	4.40
28	100	50	50	4	∞	1090.80	NF	1710.64	**1697.03**	-	56.82	**55.58**	-	-	2.51	4.48
29	100	66	34	4	∞	1563.01	6409.48	1911.48	**1903.59**	310.07	22.29	**21.79**	-70.18	**-70.30**	2.08	5.15
30	100	80	20	4	∞	1756.86	5951.27	2103.12	**2089.83**	238.74	19.71	**18.95**	-64.66	**-64.88**	1.83	4.93
31	249	124	125	2	310	4176.58	NF	5764.92	**5722.93**	-	38.03	**37.02**	-	-	13.70	72.36
32	249	166	83	2	310	4545.91	36266.06	5972.5	**5944.65**	697.77	31.38	**30.77**	-83.53	**-83.61**	12.76	72.36
33	249	199	50	2	310	5109.86	23872.02	6576.35	**6419.16**	367.18	28.70	**25.62**	-72.45	**-73.11**	10.39	47.89
34	249	124	125	3	310	3956.85	NF	5354.1	**5254.69**	-	35.31	**32.80**	-	-	13.69	64.33
35	249	166	83	3	310	4303.9	37802.66	5682.18	**5501.2**	778.33	32.02	**27.82**	-84.97	**-85.45**	11.78	55.98
36	249	199	50	3	310	4748.22	28850.25	6012.82	**5972.33**	507.60	26.63	**25.78**	-79.16	**-79.30**	10.33	42.92
37	249	124	125	4	310	3854.86	NF	5296.05	**5224.08**	-	37.39	**35.52**	-	-	13.81	59.74
38	249	166	83	4	310	4156.53	50797.46	5430.98	**5422.48**	1122.11	30.66	**30.46**	-89.31	**-89.33**	11.35	47.95
39	249	199	50	4	310	4542.36	56917.59	5915.18	**5777.57**	1153.04	30.22	**27.19**	-89.61	**-89.85**	9.23	61.46
									Average:		34.50	**32.29**	-76.03	**-76.41**	**7.21**	30.39

Table 22. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 2 continued

No	N	L	B	m	TD	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
						LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
40	249	124	125	5	310	2057.12	NF	5311.47	**5175.42**	-	158.20	**151.59**	-	-	13.81	61.75
41	249	166	83	5	310	2452.86	50671.50	5519.69	**5355.20**	1965.81	125.03	**118.32**	-89.11	**-89.43**	13.81	71.89
42	249	199	50	5	310	4524.18	58203.43	**5776.31**	5790.75	1186.50	**27.68**	28.00	**-90.08**	-90.05	9.54	61.02
43	80	40	40	2	∞	966.80	5586.96	1853.88	**1719.08**	477.88	91.75	**77.81**	-66.82	**-69.23**	2.92	8.48
44	80	53	27	2	∞	1826.95	4863.88	2283.56	**2219.67**	166.23	24.99	**21.50**	-53.05	**-54.36**	2.53	9.26
45	80	64	16	2	∞	1722.40	11101.78	2161.20	**2096.82**	544.55	25.48	**21.74**	-80.53	**-81.11**	2.32	6.21
46	80	40	40	2	200	1565.52	1695.10	**1695.10**	**1695.10**	8.28	8.28	**8.28**	0.00	**0.00**	1.87	9.42
47	80	53	27	2	200	2060.53	2227.32	2335.07	**2304.24**	8.09	13.32	**11.83**	4.84	**3.45**	1.70	9.88
48	80	64	16	2	200	1896.39	2147.20	2299.75	**2251.03**	13.23	21.27	**18.70**	7.10	**4.84**	1.70	6.86
49	80	40	40	2	180	1694.02	1732.48	**1732.48**	**1732.48**	2.27	2.27	**2.27**	0.00	**0.00**	1.89	6.93
50	80	53	27	2	180	2171.40	2467.20	2581.84	**2502.89**	13.62	18.90	**15.27**	4.65	**1.45**	1.56	5.62
51	80	64	16	2	180	1953.17	2220.13	2394.00	**2381.37**	13.67	22.57	**21.92**	7.83	**7.26**	1.62	6.10
52	160	80	80	4	∞	1794.07	NF	3452.42	**3408.77**	-	92.44	**90.00**	-	-	8.75	16.77
53	160	106	54	4	∞	1858.60	18409.91	4367.81	**4302.27**	890.53	135.01	**131.48**	-76.27	**-76.63**	8.40	18.10
54	160	128	32	4	∞	3330.79	37027.98	4292.18	**4202.76**	1011.69	28.86	**26.18**	-88.41	**-88.65**	7.10	18.51
55	160	80	80	4	200	2939.16	3734.24	3614.02	**3340.14**	27.05	22.96	**13.64**	-3.22	**-10.55**	4.13	19.20
56	160	106	54	4	200	3931.49	5340.57	4642.76	**4623.60**	35.84	18.09	**17.60**	-13.07	**-13.42**	3.14	17.30
57	160	128	32	4	200	3617.92	10144.85	4483.47	**4333.87**	180.41	23.92	**19.79**	-55.81	**-57.28**	3.14	17.48
58	160	80	80	4	180	3080.43	4972.77	3707.52	**3572.75**	61.43	20.36	**15.98**	-25.44	**-28.15**	3.65	18.38
59	160	106	54	4	180	4148.91	4823.18	5081.23	**4878.64**	16.25	22.47	**17.59**	5.35	**1.15**	2.68	14.98
60	160	128	32	4	180	3695.85	5396.20	4783.88	**4628.89**	46.01	29.44	**25.25**	-11.35	**-14.22**	2.84	14.57
									Average:		44.44	**40.70**	-32.81	**-34.47**	**4.72**	19.94

Table 23. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 2 continued

No	N	L	B	m	TD	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
						LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
61	240	120	120	6	∞	1898.29	NF	**5130.18**	5136.00	-	**170.25**	170.56	-	-	19.80	40.82
62	240	160	80	6	∞	2387.68	39988.20	**6394.73**	6444.87	1574.77	**167.82**	169.92	**-84.01**	-83.88	19.24	42.91
63	240	192	48	6	∞	2553.35	43581.99	6334.81	**6320.47**	1606.86	148.10	**147.54**	-85.46	**-85.50**	13.65	37.09
64	240	120	120	6	200	4255.21	NF	5136.37	**5054.45**	-	20.71	**18.78**	-	-	7.83	38.15
65	240	160	80	6	200	5731.72	8505.03	7145.17	**6918.83**	48.39	24.66	**20.71**	-15.99	**-18.65**	5.77	36.61
66	240	192	48	6	200	5223.25	14641.67	6684.20	**6671.42**	180.32	27.97	**27.73**	-54.35	**-54.44**	5.60	32.07
67	240	120	120	6	180	4506.11	NF	5629.08	**5421.14**	-	24.92	**20.31**	-	-	6.56	32.80
68	240	160	80	6	180	6076.72	7405.87	**7221.77**	7240.41	21.87	**18.84**	19.15	**-2.49**	-2.23	5.61	33.48
69	240	192	48	6	180	5472.49	13126.48	**7076.98**	7202.43	139.86	**29.32**	31.61	**-46.09**	-45.13	6.16	32.27
70	360	180	180	9	∞	2585.88	NF	7927.65	**7628.62**	-	206.57	**195.01**	-	-	51.02	72.11
71	360	240	120	9	∞	3154.55	NF	9837.55	**9723.95**	-	211.85	**208.25**	-	-	36.68	72.16
72	360	288	72	9	∞	3593.63	NF	9622.14	**9464.52**	-	167.76	**163.37**	-	-	45.66	72.03
73	360	180	180	9	200	6303.77	29102.73	**7621.36**	7717.74	361.67	**20.90**	22.43	**-73.81**	-73.48	20.54	68.83
74	360	240	120	9	200	8298.08	41292.83	10474.75	**10210.19**	397.62	26.23	**23.04**	-74.63	**-75.27**	13.96	68.87
75	360	288	72	9	200	7783.37	41722.85	10111.23	**9953.53**	436.05	29.91	**27.88**	-75.77	**-76.14**	14.55	67.98
76	360	180	180	9	180	6591.66	26302.62	8336.12	**8027.14**	299.03	26.46	**21.78**	-68.31	**-69.48**	11.26	72.09
77	360	240	120	9	180	8939.87	12536.3	10787.26	**10707.66**	40.23	20.66	**19.77**	-13.95	**-14.59**	8.19	60.84
78	360	288	72	9	180	8002.54	58001.29	10832.86	**10632.7**	624.79	35.37	**32.87**	-81.32	**-81.67**	8.87	69.44
									Average:		76.57	**74.48**	-56.35	**-56.71**	**16.72**	52.81

customers. To sum up, VNS-GRAMPS and VNS algorithms achieved an average gap of -57.01% and -56.21% respectively, with respect to UB. VNS is much faster than VNS-GRAMPS where an average of 2.04 minutes was required by the former and 5.07 minutes by the latter. In the second category, VNS-GRAMPS obtained the largest average gap value against LB for problems with 360 customers but the smallest value for instances with 160 customers. VNS-GRAMPS and VNS algorithms obtained an average of 58.18% and 60.85% gap values respectively. When the performances of the algorithms were compared with respect to the average gap values against UB, VNS-GRAMPS always obtained smaller gap values than VNS. VNS-GRAMPS obtained the smallest gap value for instances with 249 customers and the largest gap for instances with 160 customers. VNS-GRAMPS and VNS algorithms obtained an average of -58.87% and -57.95% gap values, respectively, with respect to UB. With respect to CPU times, VNS is faster than VNS-GRAMPS with the former using an average of 12.54 minutes while the latter requiring 46.81 minutes.

Table 24. Summary of VNS and VNS-GRAMPS Test Results for Scenario 2

Small-Sized

N	# Best Solts.		Avr. Gap vs LB (%)		Avr. Gap vs UB (%)		CPU (min)	
	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
50	1/6	6/6	23.38	20.86	-68.78	-69.45	1.19	2.02
55	0/3	3/3	18.33	15.10	-70.76	-71.60	1.14	2.31
75	0/3	3/3	41.34	40.12	-79.95	-80.22	1.71	4.50
80	2/9	9/9	25.43	22.15	-19.55	-20.86	2.01	7.64
85	0/6	6/6	19.27	17.37	-72.60	-73.03	1.84	4.92
100	0/12	12/12	36.68	33.69	-68.42	-69.06	2.89	5.59
		Average:	28.31	**25.61**	-56.21	**-57.01**	**2.04**	5.07

Large-Sized

N	# Best Solts.		Avr. Gap vs LB (%)		Avr. Gap vs UB (%)		CPU (min)	
	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
160	0/9	9/9	43.73	39.72	-33.53	-35.97	4.87	17.25
240	4/9	5/9	70.29	69.59	-48.06	-48.31	10.02	36.24
249	1/12	11/12	50.10	47.57	-84.78	-85.02	12.02	59.97
360	1/9	8/9	82.86	79.38	-64.63	-65.11	23.41	69.37
		Average:	60.85	**58.18**	-57.95	**-58.87**	**12.54**	46.81

Presence of maximum distance/time constraint-

In Table 25, a summary performance for the two algorithms was reported according to the maximum travel distance, TD, constraint. The application of TD constraint to the problems decreased the average gap values obtained against LB. Also, as the TD

constraint value increased, the average gap against LB increased, i.e. the solution quality decreased. In other words, we can conclude that the addition of TD constraints eases the problem and improves the performance of the meta-heuristics. One of the reason for the increase in performance of the meta-heuristics is due to the fact that the TD constraint results in the generation of solutions consisting of many routes with fewer customers. This feature provides extra flexibility to the local search to improve the solution. On the contrary, the average gap values against UB decreased as the TD constraint value increased, i.e. the UB values found by CPLEX increased. In terms of CPU time, the TD constraint increased the problem solving times. Furthermore, it was noted that as TD constraint value increased, the CPU times also increased.

Table 25. VNS and VNS-GRAMPS Summary Test Results with respect to Travel Distance Constraint for Scenario 2

TD	# Best Solts.		Avr. Gap vs LB (%)		Avr. Gap vs UB (%)		CPU (min)	
	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
180	3/12	10/12	22.63	20.31	-21.01	-22.33	5.07	30.63
200	2/12	11/12	21.52	19.20	-32.25	-33.72	6.99	32.72
310	1/12	11/12	50.10	47.57	-84.78	-85.02	12.02	59.97
∞	3/42	40/42	55.86	52.92	-71.91	-72.53	6.65	12.94
		Average:	44.58	**41.90**	-57.02	**-57.88**	**7.29**	25.94

Results for Scenario 3:

The performances of the VNS-GRAMPS and VNS meta-heuristics for the scenario 3 data set were reported in Tables 26 and 27. Results were reported in the same format as Table 20. The VNS algorithm only found better results than VNS-GRAMPS for two problem instances, #3 and #28. VNS-GRAMPS and VNS algorithms have found solutions with average gap values of 91.08% and 96.07%, respectively, from the LB values found with CPLEX. When the solutions are compared with respect to the average gaps from the UB values of CPLEX, VNS-GRAMPS and VNS obtained solutions with average gap values of -72.38% and -71.82%, respectively. While the VNS algorithm found a solution in an average of 11.46 minutes, VNS-GRAMPS required an average of 29.88 minutes.

In Table 28, the performance of VNS-GRAMPS and VNS algorithms was summarized under two categories similarly to scenario 2. Here, the small-sized

Table 26. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 3

No	N	L	B	m	TD	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
						LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
1	48	24	24	4	500	1041.97	1143.26	1163.66	**1143.14**	9.72	11.68	**9.71**	1.78	**-0.01**	1.35	1.30
2	48	32	16	4	500	1080.44	1338.07	1266.17	**1232.63**	23.84	17.19	**14.09**	-5.37	**-7.88**	1.33	1.33
3	48	38	10	4	500	1068.02	1431.42	**1194.12**	1227.41	34.03	**11.81**	14.92	**-16.58**	-14.25	1.27	1.60
4	96	48	48	4	480	858.03	NF	1890.23	**1877.73**	-	120.30	**118.84**	-	-	3.26	7.10
5	96	64	32	4	480	1581.22	5678.59	1929.81	**1824.28**	259.13	22.05	**15.37**	-66.02	**-67.87**	3.91	7.22
6	96	76	20	4	480	1528.45	10530.64	1895.20	**1850.47**	588.98	23.99	**21.07**	-82.00	**-82.43**	3.61	7.17
7	144	72	72	4	460	1059.15	NF	2697.76	**2634.71**	-	154.71	**148.76**	-	-	7.99	18.76
8	144	96	48	4	460	1180.92	8673.56	2849.78	**2643.98**	634.47	141.32	**123.89**	-67.14	**-69.52**	8.09	17.43
9	144	115	29	4	460	2286.80	23867.27	2907.00	**2885.79**	943.70	27.12	**26.19**	-87.82	**-87.91**	6.27	16.99
10	192	96	96	4	440	1166.74	NF	3122.18	**3027.00**	-	167.60	**159.44**	-	-	13.57	31.91
11	192	128	64	4	440	1451.40	13907.28	3334.06	**3281.09**	858.20	129.71	**126.06**	-76.03	**-76.41**	11.43	32.57
12	192	153	39	4	440	2581.56	35529.90	3460.82	**3351.26**	1276.30	34.06	**29.82**	-90.26	**-90.57**	11.29	32.07
13	240	120	120	4	420	1517.40	NF	3818.00	**3709.25**	-	151.61	**144.45**	-	-	19.08	72.14
14	240	160	80	4	420	1866.99	18236.92	3945.46	**3900.39**	876.81	111.33	**108.91**	-78.37	**-78.61**	16.78	44.09
15	240	192	48	4	420	1975.62	39200.99	4161.90	**4146.69**	1884.24	110.66	**109.89**	-89.38	**-89.42**	18.40	40.55
									Average:		82.34	**78.09**	-59.74	**-60.44**	**8.51**	22.15

Table 27. VNS and VNS-GRAMPS Test Results for MDHFVRPB Scenario 3 continued

No	N	L	B	m	TD	Total Cost				Gap wrt LB (%)			Gap wrt UB (%)		CPU (min)	
						LB	UB	VNS	VNS-GRAMPS	UB	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
16	288	144	144	4	400	1626.06	NF	4276.32	**4163.07**	-	162.99	**156.02**	-	-	29.69	70.72
17	288	192	96	4	400	1983.33	23238.12	4614.10	**4507.43**	1071.67	132.64	**127.27**	-80.14	**-80.60**	24.93	64.08
18	288	230	58	4	400	2182.09	52083.42	4808.65	**4611.96**	2286.86	120.37	**111.36**	-90.77	**-91.15**	21.97	60.57
19	72	36	36	6	500	793.12	NF	1630.02	**1574.27**	-	105.52	**98.49**	-	-	2.06	3.59
20	72	48	24	6	500	1284.44	9838.64	1587.98	**1510.63**	665.99	23.63	**17.61**	-83.86	**-84.65**	1.96	3.48
21	72	57	15	6	500	1374.99	7181.57	1700.50	**1666.80**	422.30	23.67	**21.22**	-76.32	**-76.79**	1.94	3.97
22	144	72	72	6	475	992.01	NF	2591.85	**2499.28**	-	161.27	**151.94**	-	-	7.26	18.79
23	144	96	48	6	475	2126.5	23254.65	2746.64	**2720.16**	993.56	29.16	**27.92**	-88.19	**-88.30**	6.62	17.05
24	144	115	29	6	475	2226.3	26107.44	2823.41	**2739.19**	1072.68	26.82	**23.04**	-89.19	**-89.51**	5.57	17.72
25	216	108	108	6	450	1173.98	NF	3493.77	**3354.34**	-	197.60	**185.72**	-	-	18.46	42.73
26	216	144	72	6	450	1489.87	12531.5	3550.49	**3472.94**	741.11	138.31	**133.10**	-71.67	**-72.29**	14.74	42.41
27	216	172	44	6	450	1626.69	40623.36	3812.2	**3750.37**	2397.30	134.35	**130.55**	-90.62	**-90.77**	11.35	37.78
28	288	144	144	6	425	1644.95	NF	**4425.41**	4441.84	-	**169.03**	170.03	-	-	25.30	66.98
29	288	192	96	6	425	2289.82	48315.11	4944.00	**4708.69**	2010.00	115.91	**105.64**	-89.77	**-90.25**	23.62	59.31
30	288	230	58	6	425	2414.5	52279.96	4968.41	**4854.11**	2065.25	105.77	**101.04**	-90.50	**-90.72**	20.60	54.91
									Average:		109.80	**104.06**	-85.10	**-85.50**	**14.40**	37.61

problems have 96 and fewer customers whereas the large ones contain 144 and more customers. In the first group, VNS-GRAMPS achieved lower average gap values against LB for each problem size, and as the number of customers increased, the average gap values increased. Here, VNS-GRAMPS achieved an average gap of 36.81% against LB while VNS obtained 39.98%. VNS-GRAMPS also obtained lower average gap values against UB than VNS for each problem size. The average gap values against UB were decreased as the number of customers increased, since the UB values found with CPLEX increased. VNS-GRAMPS achieved an average gap of -47.70% against UB while VNS obtained -46.91%. However, VNS solved these small-sized problems in an average of 2.30 minutes while VNS-GRAMPS needed an average of 4.08 minutes. In the second group, VNS-GRAMPS obtained lower average gap values against LB for each problem size, and the average gap values generally increased as the number of customers increased. VNS-GRAMPS and VNS obtained the largest average gap against LB for problem size with 216 customers and the smallest average gap against LB for problem size with 144 customers. For large-sized problems, VNS-GRAMPS and VNS achieved an average gap of 114.34% and 120.11%, respectively, with respect to LB. When the results were compared with respect to the average gap values from the UB values found with CPLEX, VNS-GRAMPS always found smaller gap values, but the obtained gap results were similar and there was a slight difference for each class. The reason for this similarity was that the UB values found by CPLEX for larger problem instances were too far from the solutions found by VNS-GRAMPS and VNS. VNS found solutions in shorter average CPU time for each problem size. For instance, VNS solved large problems in an average of 15.38 minutes, while VNS-GRAMPS required an average of 40.93 minutes.

3.2.2 Computational Performance of VNS-GRAMPS on Other Problems

In this section, the computational performance of the hybrid unified VNS-GRAMPS meta-heuristic embedded in the DSS was examined on the single-depot and multi-depot VRP instances described in the previous section except MDHFVRPB.

The results of the VNS-GRAMPS obtained on the problem instances included in Set 1 for HFFVRP were given in Table 29. In this table and other tables in this section,

Table 28. VNS and VNS-GRAMPS Summary Test Results for Scenario 3

Small-Sized

N	# Best Solts.		Avr. Gap vs LB (%)		Avr. Gap vs UB (%)		CPU (min)	
	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
48	1/3	2/3	13.56	12.91	-6.72	-7.38	1.32	1.41
72	0/3	3/3	50.94	45.77	-80.09	-80.72	1.99	3.68
96	0/3	3/3	55.45	51.76	-74.01	-75.15	3.59	7.16
		Average:	39.98	**36.81**	-46.91	**-47.70**	**2.30**	4.08

Large-Sized

N	# Best Solts.		Avr. Gap vs LB (%)		Avr. Gap vs UB (%)		CPU (min)	
	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS	VNS	VNS-GRAMPS
144	0/6	6/6	90.07	83.62	-83.08	-83.81	6.97	17.79
192	0/3	3/3	110.46	105.11	-83.14	-83.49	12.10	32.18
216	0/3	3/3	156.75	149.79	-81.14	-81.53	14.85	40.97
240	0/3	3/3	124.53	121.08	-83.87	-84.02	18.09	52.26
288	1/6	5/6	134.45	128.56	-87.79	-88.18	24.35	62.76
		Average:	120.11	**114.34**	-84.27	**-84.72**	**15.38**	40.93

N represents the total number of customers, BKS represents the best solution found in the literature for the related problem, and "Reference" represents the study reported the best solution. VNS-GRAMPS meta-heuristic obtained results with an average 4.18% gap from the best known solutions in a short CPU time of 0.81 minutes. The results obtained for the problems identified in Set 2 with VNS-GRAMPS were reported in Table 30. VNS-GRAMPS obtained the results with an average gap of 2.84% from the best known solutions in 0.64 minutes.

The best known solutions reported in the literature for FSMVRPB were obtained in the studies of Penna et al. (2019), which suggested a hybrid unified heuristic solution method, Salhi et al. (2013), which proposed a set partitioning-based heuristic, and Belloso et al. (2019), which suggested a biased-randomized heuristic. The results of the VNS-GRAMPS meta-heuristic on 36 problem instances proposed for the FSMVRPB were reported in Table 31. VNS-GRAMPS achieved an average gap value of 5.87% with an average CPU time of 0.76 minutes. Moreover, a new best solution for problem #13 with an average gap of 0.08% from the best known solution reported in the literature.

There are many studies in the literature suggesting exact and heuristic solution methods for MVRP. The best known solutions for the problem instances examined

Table 29. Computational results of HFFVRP for the instances in Set 1

No	N	Vehicle Fleet	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
13	50	4A, 2B, 4C, 4D, 2E, 1F	1518.05	Taillard (1999)	1596.12	5.14	0.70
14	50	4A, 2B, 1C	607.53	Li (2010)	622.86	2.52	0.39
15	50	4A, 3B, 2C	1015.29	Li (2010)	1034.20	1.86	0.35
16	50	2A, 4B, 3C	1145.52	Tarantilis (2003)	1167.00	1.88	0.32
17	75	4A, 4B, 2C, 1D	1061.96	Li (2010)	1093.99	3.02	0.73
18	75	4A, 4B, 2C, 2D, 1E, 1F	1823.58	Li (2010)	1941.67	6.48	0.86
19	100	4A, 3B, 3C	1117.51	Taillard (1999)	1191.97	6.66	1.88
20	100	6A, 4B, 3C	1534.17	Li (2010)	1624.96	5.92	1.22
					Average:	4.18	0.81

Table 30. Computational results of HFFVRP for the instances in Set 2

No	N	Vehicle Fleet	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
13	50	1B, 1C, 7D, 2E, 1F	1491.86	Tütüncü (2010)	1510.16	1.23	0.30
14	50	3A, 4B	603.21	Tütüncü (2010)	611.94	1.45	0.30
15	50	7A, 3B, 1C	999.82	Tütüncü (2010)	1009.55	0.97	0.30
16	50	8A, 1B, 3C	1131.00	Gendreau (1999)	1159.68	2.54	0.31
17	75	2B, 4C, 1D	1038.60	Gendreau (1999)	1095.77	5.50	0.67
18	75	2B, 4C, 2E, 1F	1801.40	Gendreau (1999)	1919.95	6.58	0.74
19	100	5A, 5B	1100.96	Taillard (1999)	1133.07	2.92	1.30
20	100	6A, 1B, 5C	1541.18	Gendreau (1999)	1564.41	1.51	1.16
					Average:	2.84	0.64

Table 31. Computational results of FSMVRPB

No	N	L	B	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
HWS1	20	10	10	720.57	Penna (2019)	742.58	3.05	0.18
HWS2	20	13	7	818.12	Salhi (2013)	832.10	1.71	0.20
HWS3	20	16	4	848.32	Belloso (2019)	**848.31**	0.00	0.18
HWS4	20	10	10	4342.48	Belloso (2019)	4825.36	11.12	0.20
HWS5	20	13	7	5357.98	Belloso (2019)	5375.46	0.33	0.17
HWS6	20	16	4	5421.65	Penna (2019)	6290.82	16.03	0.17
HWS7	20	10	10	729.50	Belloso (2019)	**729.51**	0.00	0.17
HWS8	20	13	7	838.11	Belloso (2019)	**838.13**	0.00	0.17
HWS9	20	16	4	890.76	Belloso (2019)	**890.76**	0.00	0.17
HWS10	20	10	10	4349.13	Belloso (2019)	**4349.14**	0.00	0.17
HWS11	20	13	7	5363.58	Belloso (2019)	5840.55	8.89	0.17
HWS12	20	16	4	5497.98	Penna (2019)	6302.82	14.64	0.16
HWS13	50	25	25	1625.70	Salhi (2013)	**1624.48**	-0.08	0.47
HWS14	50	33	17	1771.53	Penna (2019)	1824.69	3.00	0.40
HWS15	50	40	10	1999.05	Penna (2019)	2020.94	1.10	0.38
HWS16	50	25	25	5551.19	Penna (2019)	6542.59	17.86	0.56
HWS17	50	33	17	6547.93	Penna (2019)	7063.11	7.87	0.48
HWS18	50	40	10	7120.52	Penna (2019)	7582.39	6.49	0.42
HWS19	50	25	25	1616.21	Penna (2019)	1713.63	6.03	0.44
HWS20	50	33	17	2015.67	Penna (2019)	2070.02	2.70	0.43
HWS21	50	40	10	2295.57	Penna (2019)	2369.81	3.23	0.41
HWS22	50	25	25	1717.60	Penna (2019)	1762.29	2.60	0.43
HWS23	50	33	17	2096.10	Penna (2019)	2162.23	3.15	0.40
HWS24	50	40	10	2401.04	Penna (2019)	2460.57	2.48	0.36
HWS25	75	37	38	1285.86	Penna (2019)	1327.65	3.25	1.30
HWS26	75	50	25	1399.36	Penna (2019)	1438.14	2.77	1.12
HWS27	75	60	15	1513.10	Penna (2019)	1562.01	3.23	1.01
HWS28	75	37	38	1572.38	Penna (2019)	1665.89	5.95	1.36
HWS29	75	50	25	1760.95	Penna (2019)	1921.85	9.14	1.07
HWS30	75	60	15	1950.99	Penna (2019)	2185.73	12.03	0.92
HWS31	100	50	50	4943.29	Salhi (2013)	5511.15	11.49	2.98
HWS32	100	66	34	5993.30	Penna (2019)	6894.74	15.04	2.46
HWS33	100	80	20	7097.81	Penna (2019)	8137.90	14.65	2.13
HWS34	100	50	50	2465.41	Salhi (2013)	2707.71	9.83	2.06
HWS35	100	66	34	2927.20	Penna (2019)	3100.83	5.93	1.89
HWS36	100	80	20	3450.73	Penna (2019)	3645.19	5.64	1.59
						Average:	5.87	0.76

in this study were obtained with the bi-level Voronoi diagram-based meta-heuristic proposed by Tu et al. (2014) and iterated local search and a hybrid genetic algorithm based on implicit depot assignments and rotations proposed by Vidal et al. (2014). The results of the VNS-GRAMPS meta-heuristic on the MVRP problem instances in the Set 1 were given in Table 32, and the results for the ones in the Set 2 were given in Table 33. VNS-GRAMPS achieved an average gap value of 2.52% with an average CPU time of 14.71 minutes for the problems in Set 1. VNS-GRAMPS also found the best known solution for problem #1 and the new best solutions for problems #12 and #13 with an average gap of -0.01%. VNS-GRAMPS obtained an average gap value of 1.42% with an average CPU time of 10.60 minutes for the problems in Set 2. VNS-GRAMPS also found the best known solution for problem #1, and a new best solution

an average gap value of -0.33% for problem #2.

Table 32. Computational results of MVRP for the instances in Set 1

No	N	m	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
1	50	4	576.87	Vidal (2014)	**576.85**	0.00	0.44
2	50	4	473.53	Vidal (2014)	480.05	1.38	0.65
3	75	2	640.65	Vidal (2014)	648.51	1.23	1.14
4	100	2	999.21	Vidal (2014)	1026.57	2.74	1.41
5	100	2	750.03	Vidal (2014)	763.32	1.77	2.2
6	100	3	876.50	Vidal (2014)	883.05	0.75	1.17
7	100	4	881.97	Vidal (2014)	898.31	1.85	1.64
8	249	2	4372.78	Vidal (2014)	4550.42	4.06	23.01
9	249	3	3858.66	Vidal (2014)	3938.98	2.08	16.54
10	249	4	3631.11	Vidal (2014)	3765.54	3.70	27.68
11	249	5	3546.06	Vidal (2014)	3708.66	4.59	21.72
12	80	2	1318.95	Vidal (2014)	**1318.88**	-0.01	1.14
13	80	2	1318.95	Vidal (2014)	**1318.88**	-0.01	1.89
14	80	2	1360.12	Vidal (2014)	1365.6	0.40	2.77
15	160	4	2505.42	Vidal (2014)	2585.47	3.20	6.41
16	160	4	2572.23	Vidal (2014)	2584.37	0.47	12.54
17	160	4	2708.99	Tu (2010)	2759.43	1.86	17.19
18	240	6	3702.85	Vidal (2014)	3860.54	4.26	16.14
19	240	6	3827.06	Vidal (2014)	3925.2	2.56	29.56
20	240	6	4058.07	Vidal (2014)	4280.98	5.49	34.58
21	360	9	5474.84	Vidal (2014)	5813.89	6.19	48.35
22	360	9	5702.16	Vidal (2014)	5922.35	3.86	34.96
23	360	9	6078.75	Vidal (2014)	6411.52	5.47	35.24
					Average:	2.52	14.71

Table 33. Computational results of MVRP for the instances in Set 2

Problem	N	m	BKS	Reference	VNS-GRAMPS	Gap	CPU (min)
P1	48	4	861.32	Vidal (2014)	**861.31**	0.00	0.52
P2	96	4	1296.25	Vidal (2014)	**1291.92**	-0.33	2.22
P3	144	4	1803.80	Vidal (2014)	1819.37	0.86	5.63
P4	192	4	2042.45	Vidal (2014)	2120.21	3.81	12.07
P5	240	4	2324.12	Vidal (2014)	2360.06	1.55	16.96
P6	288	4	2663.56	Vidal (2014)	2734.7	2.67	25.32
P7	72	6	1075.12	Vidal (2014)	1085.61	0.98	1.16
P8	144	6	1658.23	Vidal (2014)	1694.52	2.19	4.45
P9	216	6	2131.70	Vidal (2014)	2172.14	1.90	12.57
P10	288	6	2805.53	Vidal (2014)	2822.58	0.61	25.14
					Average:	1.42	10.60

The best results reported in the literature for MVRPB were obtained in the study of Ropke and Pisinger (2004), which proposed a unified heuristic solution method. The results of the VNS-GRAMPS meta-heuristic on 33 problem instances proposed for MVRPB were reported in Table 34. VNS-GRAMPS obtained an average of 12.55% gap from the best known solutions with an average CPU time of 8.54 minutes.

There are many studies in the literature suggesting exact and heuristic solution methods for MDHFVRP. The best known solutions for the problem instances examined

Table 34. Computational results of MVRPB

No	N	L	B	m	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
1	50	25	25	4	499	Ropke (2004)	566	13.41	0.45
2	50	38	12	4	528	Ropke (2004)	572	8.34	0.42
3	50	45	5	4	569	Ropke (2004)	590	3.66	0.39
4	50	25	25	4	440	Ropke (2004)	540	22.76	0.49
5	50	38	12	4	450	Ropke (2004)	538	19.52	0.51
6	50	45	5	4	464	Ropke (2004)	497	7.20	0.49
7	75	38	37	5	581	Ropke (2004)	673	15.88	0.84
8	75	57	18	5	605	Ropke (2004)	689	13.87	1.05
9	75	68	7	5	624	Ropke (2004)	649	4.01	0.84
10	100	50	50	2	789	Ropke (2004)	900	14.13	1.54
11	100	75	25	2	875	Ropke (2004)	961	9.82	1.43
12	100	90	10	2	962	Ropke (2004)	1010	4.95	1.28
13	100	50	50	2	678	Ropke (2004)	815	20.27	2.65
14	100	75	25	2	700	Ropke (2004)	808	15.48	2.04
15	100	90	10	2	733	Ropke (2004)	795	8.42	1.93
16	100	50	50	3	745	Ropke (2004)	854	14.57	1.71
17	100	75	25	3	794	Ropke (2004)	887	11.75	1.53
18	100	90	10	3	851	Ropke (2004)	873	2.53	1.31
19	100	50	50	4	733	Ropke (2004)	828	12.94	1.66
20	100	75	25	4	802	Ropke (2004)	873	8.90	1.80
21	100	90	10	4	854	Ropke (2004)	881	3.13	1.74
22	249	125	124	2	3327	Ropke (2004)	4055	21.88	25.82
23	249	187	62	2	3762	Ropke (2004)	4287	13.96	23.22
24	249	225	24	2	4134	Ropke (2004)	4394	6.28	18.01
25	249	125	124	3	3005	Ropke (2004)	3721	23.82	26.02
26	249	187	62	3	3355	Ropke (2004)	3836	14.35	18.04
27	249	225	24	3	3677	Ropke (2004)	3932	6.94	18.15
28	249	125	124	4	2927	Ropke (2004)	3617	23.59	24.11
29	249	187	62	4	3242	Ropke (2004)	3683	13.61	23.39
30	249	225	24	4	3485	Ropke (2004)	3779	8.44	19.83
31	249	125	124	5	2855	Ropke (2004)	3565	24.88	21.10
32	249	187	62	5	3155	Ropke (2004)	3576	13.35	21.93
33	249	225	24	5	3390	Ropke (2004)	3650	7.68	16.23
							Average:	12.55	8.54

in this study were obtained with the VNS meta-heuristic application proposed by Salhi et al. (2014), iterated local search and a hybrid genetic algorithm based on implicit depot assignments and rotations recommended by Vidal et al. (2014), mathematical models based on 5 different formulations proposed by Lahyani et al. (2018) and a hybrid meta-heuristic which combines an iterated local search with variable neighbourhood descent method proposed by Penna et al. (2019). The results of the VNS-GRAMPS meta-heuristic for the MDHFVRP problem instances in the Set 1 data set were given in Table 35, and the results for the Set 2 dataset were given in Table 36. For Set 1, VNS-GRAMPS obtained an average of 4.54% gap value from the best known solutions suggested in the literature with an average CPU time of 7.89 minutes. For Set 2, VNS-GRAMPS achieved an average of 4.57% gap value with an average CPU time of 14.13 minutes.

Table 35. Computational results of MDHFVRP for the instances in Set 1

No	N	m	TD	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
1	55	4	∞	1398.30	Lahyani (2018)	1433.41	2.51	1.70
2	85	3	∞	2131.46	Lahyani (2018)	2190.45	2.77	0.60
3	85	3	∞	1483.51	Lahyani (2018)	1522.29	2.61	0.70
4	50	4	∞	1477.73	Penna (2019)	1517.84	2.71	0.20
5	50	4	∞	957.73	Penna (2019)	996.83	4.08	0.70
6	75	5	∞	1569.67	Penna (2019)	1627.30	3.67	1.50
7	100	2	∞	2292.64	Penna (2019)	2387.86	4.15	1.40
8	100	2	∞	1453.64	Penna (2019)	1501.30	3.28	3.00
9	100	3	∞	2208.66	Penna (2019)	2302.52	4.25	2.10
10	100	4	∞	2198.91	Penna (2019)	2317.49	5.39	2.10
11	249	2	310	6441.36	Vidal (2014)	6809.93	5.72	28.27
12	249	3	310	5998.70	Vidal (2014)	6332.56	5.57	20.30
13	249	4	310	5807.53	Vidal (2014)	6217.34	7.06	19.91
14	249	5	310	5770.42	Vidal (2014)	6158.83	6.73	21.88
15	80	2	∞	2072.18	Penna (2019)	2119.58	2.29	1.20
16	80	2	200	2096.39	Penna (2019)	2128.89	1.55	1.10
17	80	2	180	2139.30	Salhi (2014)	2235.94	4.52	1.48
18	160	4	∞	3973.47	Penna (2019)	4100.23	3.19	7.80
19	160	4	200	4119.76	Penna (2019)	4336.66	5.26	2.70
20	160	4	180	4309.09	Penna (2019)	4565.60	5.95	3.30
21	240	6	∞	5887.43	Penna (2019)	6248.56	6.13	4.80
22	240	6	200	6130.36	Penna (2019)	6477.87	5.67	5.10
23	240	6	180	6458.07	Penna (2019)	6723.81	4.11	2.00
24	360	9	∞	8709.26	Penna (2019)	9197.26	5.60	10.90
25	360	9	200	9151.64	Vidal (2014)	9803.88	7.13	55.40
26	360	9	180	9678.75	Penna (2019)	10273.16	6.14	4.90
						Average:	4.54	7.89

Table 36. Computational results of MDHFVRP for the instances in Set 2

No	N	m	TD	BKS	Reference	VNS-GRAMPS	Gap (%)	CPU (min)
P1	48	4	500	1181.47	Vidal (2014)	1227.57	3.90	0.50
P2	96	4	480	1901.39	Vidal (2014)	1992.17	4.77	2.51
P3	144	4	460	2712.71	Vidal (2014)	2776.82	2.36	7.67
P4	192	4	440	3370.85	Vidal (2014)	3521.08	4.46	13.12
P5	240	4	420	4066.52	Vidal (2014)	4336.57	6.64	18.49
P6	288	4	400	4669.16	Vidal (2014)	4907.16	5.10	33.68
P7	72	6	500	1550.87	Vidal (2014)	1595.9	2.90	1.16
P8	144	6	475	2705.46	Vidal (2014)	2843.15	5.09	7.85
P9	216	6	450	3637.39	Vidal (2014)	3815.87	4.91	22.15
P10	288	6	425	4973.74	Vidal (2014)	5249.53	5.54	34.21
						Average:	4.57	14.13

CHAPTER 4: CONCLUSION

In this thesis, we introduced a new logistical problem that is commonly faced in practice. In real life, the companies generally have a heterogeneous vehicle fleet to serve customers and multiple depots to supply the demands of the customers. We called this practical problem as the Multi-Depot Heterogeneous VRP with Backhauls (MDHFVRPB). We first defined two mathematical models where the first one is a basic one whereas the second one contains some new added neighbourhood reductions. We also generated new problem instances as these do not exist in the literature. To solve larger instance, we then developed an interesting unified hybridisation of VNS and GRAMPS which we refer to as the VNS-GRAMPS meta-heuristic. For comparison purposes we also presented a basic VNS meta-heuristic. We conducted 3 hours of CPU time limit to solve the basic and restricted mathematical models and found the optimal solutions of 10 out of 12 problems with 20 customers of scenario 1 data set with both models. The restricted model obtained the optimal solutions in shorter CPU times and generally found tighter lower bound for the other 26 instances where optimality cannot be guaranteed. The restricted model also found tighter lower bound in general for the generated data sets for scenarios 2 and 3.

When using VNS-GRAMPS, we obtained the optimal solutions for the 6 instances of scenario 1, namely, #2, #3, #7, #8, #9 and #11. The best solutions were also obtained for 14 out of 24 problem instances having 50 and more customers by VNS-GRAMPS, and the other 10 out of 24 best solutions were obtained by CPLEX. The lowest average gap obtained is 13.01% for VNS-GRAMPS, and the average gaps for UB and VNS were reported as 39.05% and 18.44%, respectively. With regards to CPU times, VNS-GRAMPS requires shorter CPU times for the problems with 20 customers, but needed much longer amount of time for the problem instances with 50 and more customers.

We also developed a Decision Support System (DSS) that can solve VRPs with single-depot, multi-depot, heterogeneous vehicle fleet and backhaul features with the embedded VNS-GRAMPS meta-heuristic. The developed DSS is a visual interactive solution tool called as ADVISER2. The user can save the problems to be solved and the

solutions obtained into the database of DSS, and print the report of the saved solution. After selecting the problem to be solved, the user can solve the problem with VNS-GRAMPS and can make changes on the solution, such as adding/removing customers to routes, adding/removing routes, changing the vehicle type of the route, using the interactive tools of DSS. In addition, after the user opens a solution saved in the database and makes changes on it, he can run the algorithm and get a solution quickly. The developed DSS will be an effective solution alternative to the fast and dynamic routing goal of today's world, with its interactive structure and ability to provide fast results. The developed DSS was tested on the problem instances suggested in the literature for 5 different problems, and it was able to find solutions in a short CPU time with a maximum average gap value of 5.87% from the best known solutions, except for the MVRPB problem. Moreover, new best solutions of one problem for FSMVRP and 3 problems for MVRP were found with VNS-GRAMPS.

The following research avenues could be worth exploring. For instance, in real life, it is often not practical to visit backhaul customers after completing all deliveries. Hence, one way forward is to analyse other extensions of MDHFVRPB including different pickup and delivery orders such as mixed and simultaneous. In this study, we only increased the size of the RCL dynamically but this could be made more flexible by allowing both the increase as well as the decrease. Another approach that integrates VNS with other meta-heuristics such as large neighbourhood search instead of GRASP could also be worthwhile studying. One way forward would be to hybridise one of the meta-heuristics with exact methods to yield an effective matheuristic resulting in tight bounds and exciting mathematical properties. On the DSS side, DSS could be updated to include real maps, thus real life problems could be solved. The development of this update would also enable the commercialization of the DSS.

REFERENCES

Adelzadeh, M., Mahdavi Asl, V. and Koosha, M. (2014). *A mathematical model and a solving procedure for multi-depot vehicle routing problem with fuzzy time window and heterogeneous vehicle*, The International Journal of Advanced Manufacturing Technology, Vol. 75 (5), pp. 793–802.

Ahmadi, S. and Osman, I. H. (2005). *Greedy random adaptive memory programming search for the capacitated clustering problem*, European Journal of Operational Research, Vol. 162 (1), pp. 30–44.

Aras, N., Aksen, D. and Tekin, M. T. (2011). *Selective multi-depot vehicle routing problem with pricing*, Transportation Research Part C: Emerging Technologies, Vol. 19 (5), pp. 866 – 884.

Avci, M. and Topaloglu, S. (2016). *A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery*, Expert Systems with Applications, Vol. 53, pp. 160 – 171.

Baker, B. M. (1992). *Further inprovements to vehicle routeing heuristics*, Journal of the Operational Research Society, Vol. 43 (10), pp. 1009–1012.

Ball, M. O., Golden, B. L., Assad, A. A. and Bodin, L. D. (1983). *Planning for truck fleet size in the presence of a common-carrier option*, Decision Sciences, Vol. 14 (1), pp. 103–120.

Belloso, J., Juan, A. A. and Faulin, J. (2019). *An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls*, International Transactions in Operational Research, Vol. 26 (1), pp. 289–301.

Benslimane, M. T. and Benadada, Y. (2013). *Ant colony algorithm for the multi-depot vehicle routing problem in large quantities by a heterogeneous fleet of vehicles*, INFOR: Information Systems and Operational Research, Vol. 51 (1), pp. 31–40.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I. and Laporte, G. (2007). *Static pickup and delivery problems: a classification scheme and survey*, TOP, Vol. 15 (1), pp. 1–31.

Bettinelli, A., Ceselli, A. and Righini, G. (2011). *A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows*,

Transportation Research Part C: Emerging Technologies, Vol. 19 (5), pp. 723–740.

Bianchessi, N. and Righini, G. (2007). *Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery*, Computers & Operations Research, Vol. 34 (2), pp. 578 – 594.

Brandao, J. (2006). *A new tabu search algorithm for the vehicle routing problem with backhauls*, European Journal of Operational Research, Vol. 173 (2), pp. 540 – 555.

Bräysy, O. and Hasle, G. (2014). Chapter 12: Software Tools and Emerging Technologies for Vehicle Routing and Intermodal Transportation, *Vehicle Routing*, Philadelphia PA: Society for Industrial and Applied Mathematics, pp. 351–380.

Cantu-Funes, R., Salazar-Aguilar, M. A. and Boyer, V. (2018). *Multi-depot periodic vehicle routing problem with due dates and time windows*, Journal of the Operational Research Society, Vol. 69 (2), pp. 296–306.

Carreto, C. and Baker, B. (2002). *A Grasp Interactive Approach to the Vehicle Routing Problem with Backhauls*, 1st Edition, Boston MA: Springer US, pp. 185–199.

Chao, I.-M., Golden, B. L. and Wasil, E. (1993). *A New Heuristic for the Multi-Depot Vehicle Routing Problem that Improves upon Best-Known Solutions*, American Journal of Mathematical and Management Sciences, Vol. 13 (3-4), pp. 371–406.

Chaovalitwongse, W., Kim, D. and Pardalos, P. M. (2003). *Grasp with a new local search scheme for vehicle routing problems with time windows*, Journal of Combinatorial Optimization, Vol. 7 (2), pp. 179–207.

Chen, J. F. and Wu, T. H. (2006). *Vehicle routing problem with simultaneous deliveries and pickups*, The Journal of the Operational Research Society, Vol. 57 (5), pp. 579–587.

Christofides, N. and Eilon, S. (1969). *An Algorithm for the Vehicle-dispatching Problem*, Journal of the Operational Research Society, Vol. 20 (3), pp. 309–318.

Cordeau, J.-F., Gendreau, M. and Laporte, G. (1997). *A tabu search heuristic for periodic and multi-depot vehicle routing problems*, Networks, Vol. 30 (2), pp. 105–119.

Cosco, D. O., Golden, B. L. and Wasil, E. A. (1988). *Vehicle routing with backhauls: models, algorithms and case studies*, 1st Edition, Amsterdam: Elsevier, pp. 127–147.

Crispim, J. and Brandao, J. (2005). *Metaheuristics applied to mixed and simultaneous*

extensions of vehicle routing problems with backhauls, The Journal of the Operational Research Society, Vol. 56 (11), pp. 1296–1302.

Cuervo, D. P., Goos, P., Sörensen, K. and Arraiz, E. (2014). *An iterated local search algorithm for the vehicle routing problem with backhauls*, European Journal of Operational Research, Vol. 237 (2), pp. 454 – 464.

Dantzig, G. B. and Ramser, J. H. (1959). *The truck dispatching problem*, Management Science, Vol. 6 (1), pp. 80–91.

de Oliveira, F. B., Enayatifar, R., Sadaei, H. J., Guimarães, F. G. and Potvin, J.-Y. (2016). *A cooperative coevolutionary algorithm for the multi-depot vehicle routing problem*, Expert Systems with Applications, Vol. 43, pp. 117 – 130.

Deif, I. and Bodin, L. (1984). *Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauling*, 1st Edition, Babson Park: Mass USA, pp. 75–96.

DellAmico, M., Righini, G. and Salani, M. (2006). *A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection*, Transportation Science, Vol. 40 (2), pp. 235–247.

Dethloff, J. (2001). *Vehicle routing and reverse logistics: The vehicle routing problem with simultaneous delivery and pick-up*, OR-Spektrum, Vol. 23 (1), pp. 79–96.

Drexl, M. (2012). *Rich vehicle routing in theory and practice*, Logistics Research, Vol. 5, pp. 47–63.

Ercan, C. and Gencer, C. (2018). *A Decision Support System for Dynamic Heterogeneous Unmanned Aerial System Fleets*, Gazi University Journal of Science, Vol. 31 (3), pp. 863–877.

Erdoğan, G. (2017). *An open source Spreadsheet Solver for Vehicle Routing Problems*, Computers & Operations Research, Vol. 84, pp. 62–72.

Erdoğan, G., Stylianou, N. and Vasilakis, C. (2019). *An open source decision support system for facility location analysis*, Decision Support Systems, Vol. 125 (March), pp. 113116.

Felici, G., Ferone, D., Festa, P., Napoletano, A. and Pastore, T. (2017). A grasp for the minimum cost sat problem, *in* R. Battiti, D. E. Kvasov and Y. D. Sergeyev (eds), *Learning and Intelligent Optimization*, Cham: Springer International Publishing,

pp. 64–78.

Feo, T. A. and Resende, M. G. C. (1995). *Greedy randomized adaptive search procedures*, Journal of Global Optimization, Vol. 6 (2), pp. 109–133.

Ferreira, J. A., Costa, M., Tereso, A. and A., O. J. (2015). *A Multi-Criteria Decision Support System for a Routing Problem in Waste Collection*, Vol. 9019 of *Lecture Notes in Computer Science*, 1st Edition, Cham: Springer International Publishing.

Festa, P., Pastore, T., Ferone, D., Juan, A. A. and Bayliss, C. (2018). Integrating biased-randomized grasp with monte carlo simulation for solving the vehicle routing problem with stochastic demands, *Proceedings of the 2018 Winter Simulation Conference*, WSC '18, Piscataway NJ: IEEE Press, pp. 2989–3000.

Festa, P. and Resende, M. G. C. (2009a). *An annotated bibliography of grasp – part i: Algorithms*, International Transactions in Operational Research, Vol. 16 (1), pp. 1–24.

Festa, P. and Resende, M. G. C. (2009b). *An annotated bibliography of grasp–part ii: Applications*, International Transactions in Operational Research, Vol. 16 (2), pp. 131–172.

Gajpal, Y. and Abad, P. (2009). *An ant colony system (acs) for vehicle routing problem with simultaneous delivery and pickup*, Computers & Operations Research, Vol. 36 (12), pp. 3215 – 3223.

Ganesh, K. and Narendran, T. (2007). *Cloves: A cluster-and-search heuristic to solve the vehicle routing problem with delivery and pick-up*, European Journal of Operational Research, Vol. 178 (3), pp. 699 – 717.

Geetha, S., Vanathi, P. T. and Poonthalir, G. (2012). *Metaheuristic Approach or The Multi-Depot Vehicle Routing Problem*, Applied Artificial Intelligence, Vol. 26 (9), pp. 878–901.

Gendreau, M., Laporte, G., Musaraganyi, C. and Taillard, É. D. (1999). *A tabu search heuristic for the heterogeneous fleet vehicle routing problem*, Computers & Operations Research, Vol. 26 (12), pp. 1153–1173.

Gillett, B. E. and Johnson, J. G. (1976). *Multi-terminal vehicle-dispatch algorithm*, Omega, Vol. 4 (6), pp. 711–718.

Goetschalckx, M. and Jacobs-Blecha, C. (1989). *The vehicle routing problem with backhauls*, European Journal of Operational Research, Vol. 42 (1), pp. 39 – 51.

Golden, B., Alfaro, J., Baker, E. and Schaffer, J. (1985). The vehicle routing problem with backhauling: two approaches, *Proceedings of the 21st Annual Meeting of the Southeast Institute of Management Science*, pp. 90–92.

Golden, B., Assad, A., Levy, L. and Gheysens, F. (1984). *The fleet size and mix vehicle routing problem*, Computers & Operations Research, Vol. 11 (1), pp. 49–66.

Golden, B. L., Magnanti, T. L. and Nguyen, H. Q. (1977). *Implementing vehicle routing algorithms*, Networks, Vol. 7 (2), pp. 113–148.

Gulczynski, D., Golden, B. and Wasil, E. (2011). *The multi-depot split delivery vehicle routing problem: An integer programming-based heuristic, new test problems, and computational results*, Computers & Industrial Engineering, Vol. 61 (3), pp. 794 – 804.

Haddadene, S. R. A., Labadie, N. and Prodhon, C. (2016). *A grasp × ils for the vehicle routing problem with time windows, synchronization and precedence constraints*, Expert Systems with Applications, Vol. 66, pp. 274 – 294.

Halse, K. (1992). *Modeling and solving complex vehicle routing problems*, PhD thesis, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark, Lyngby.

Ho, W., Ho, G. T., Ji, P. and Lau, H. C. (2008). *A hybrid genetic algorithm for the multi-depot vehicle routing problem*, Engineering Applications of Artificial Intelligence, Vol. 21 (4), pp. 548–557.

Hsieh, F.-S. and Huang, H. W. (2015). Decision support for vehicle routing problem with arbitrary pickup/delivery points, *2015 IEEE 12th International Conference on Networking, Sensing and Control*, IEEE, pp. 597–602.

Irnich, S. (2000). *A multi-depot pickup and delivery problem with a single hub and heterogeneous vehicles*, European Journal of Operational Research, Vol. 122 (2), pp. 310 – 328.

Kara, I. (2011). Arc based integer programming formulations for the distance constrained vehicle routing problem, *3rd IEEE International Symposium on Logistics and Industrial Informatics*, pp. 33–38.

Kocatürk, F. and Özpeynirci, O. (2014). *Variable neighborhood search for the pharmacy duty scheduling problem*, Computers & Operations Research, Vol. 51, pp. 218–226.

Kocatürk, F., Tütüncü, G. Y. and Salhi, S. (2021). *The multi-depot heterogeneous VRP with backhauls: formulation and a hybrid VNS with GRAMPS meta-heuristic approach*, Annals of Operations Research, Vol. 307 (1-2), pp. 277–302.

Koç, C. and Laporte, G. (2018). *Vehicle routing with backhauls: Review and research perspectives*, Computers & Operations Research, Vol. 91, pp. 79 – 91.

Koç, C., Laporte, G. and Tükenmez, I. (2020). *A review of vehicle routing with simultaneous pickup and delivery*, Computers & Operations Research, Vol. 122, pp. 104987.

Kuo, Y. and Wang, C.-C. (2012). *A variable neighborhood search for the multi-depot vehicle routing problem with loading cost*, Vol. 39, p. 6949–6954.

Laguna, M. and Martí, R. (2001). *A grasp for coloring sparse graphs*, Computational Optimization and Applications, Vol. 19 (2), pp. 165–178.

Lahyani, R., Coelho, L. C. and Renaud, J. (2018). *Alternative formulations and improved bounds for the multi-depot fleet size and mix vehicle routing problem*, OR Spectrum, Vol. 40 (1), pp. 125–157.

Laporte, G., Nobert, Y. and Arpin, D. (1984). *Optimal solutions to capacitated multidepot vehicle routing problems*, 1st Edition, Montreal: Congressus Numerantium.

Laporte, G., Nobert, Y. and Taillefer, S. (1988). *Solving a Family of Multi-Depot Vehicle Routing and Location-Routing Problems*, Transportation Science, Vol. 22 (3), pp. 161–172.

Lenstra, J. K. and Kan, A. H. G. R. (1981). *Complexity of vehicle routing and scheduling problems*, Networks, Vol. 11 (2), pp. 221–227.

Leyerer, M., Sonneberg, M.-O., Heumann, M., Kammann, T. and Breitner, M. H. (2019). *Individually Optimized Commercial Road Transport: A Decision Support System for Customizable Routing Problems*, Sustainability, Vol. 11 (20), pp. 5544.

Li, J., Pardalos, P. M., Sun, H., Pei, J. and Zhang, Y. (2015). *Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups*, Expert Systems with Applications, Vol. 42 (7), pp. 3551 – 3561.

Liu, R., Jiang, Z., Fung, R. Y., Chen, F. and Liu, X. (2010). *Two-phase heuristic algorithms for full truckloads multi-depot capacitated vehicle routing problem in carrier collaboration*, Computers & Operations Research, Vol. 37 (5), pp. 950–959.

Mancini, S. (2016). *A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based matheuristic*, Transportation Research Part C: Emerging Technologies, Vol. 70, pp. 100 – 112.

Mendoza, J. E., Medaglia, A. L. and Velasco, N. (2009). *An evolutionary-based decision support system for vehicle routing: The case of a public utility*, Decision Support Systems, Vol. 46 (3), pp. 730–742.

Min, H. (1989). *The multiple vehicle routing problem with simultaneous delivery and pick-up points*, Transportation Research Part A: General, Vol. 23 (5), pp. 377 – 386.

Min, H. and Melachrinoudis, E. (2016). *A model-based decision support system for solving vehicle routing and driver scheduling problems under hours of service regulations*, International Journal of Logistics Research and Applications, Vol. 19 (4), pp. 256–277.

Mingozzi, A., Giorgi, S. and Baldacci, R. (1999). *An exact method for the vehicle routing problem with backhauls*, Transportation Science, Vol. 33 (3), pp. 315–329.

Mirabi, M., Fatemi Ghomi, S. and Jolai, F. (2010). *Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem*, Robotics and Computer-Integrated Manufacturing, Vol. 26 (6), pp. 564–569.

Mladenović, N. and Hansen, P. (1997). *Variable neighborhood search*, Computers & Operations Research, Vol. 24 (11), pp. 1097–1100.

Montane, F. A. T. and Galvao, R. D. (2006). *A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service*, Computers & Operations Research, Vol. 33 (3), pp. 595 – 619.

Montoya-Torres, J. R., López Franco, J., Nieto Isaza, S., Felizzola Jiménez, H. and Herazo-Padilla, N. (2015). *A literature review on the vehicle routing problem with multiple depots*, Computers & Industrial Engineering, Vol. 79, pp. 115–129.

Mosheiov, G. (1998). *Vehicle routing with pick-up and delivery: tour-partitioning heuristics*, Computers & Industrial Engineering, Vol. 34 (3), pp. 669 – 684.

Nagy, G. and Salhi, S. (2005). *Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries*, European Journal of Operational Research, Vol. 162 (1), pp. 126 – 141.

Osman, I. H. and Wassan, N. A. (2002). *A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls*, Journal of Scheduling, Vol. 5 (4), pp. 263–285.

Ouertani, N., Ben-Romdhane, H. and Krichen, S. (2022). *A decision support system for the dynamic hazardous materials vehicle routing problem*, Operational Research, Vol. 22, pp. 551–576.

Penna, P. H. V., Subramanian, A., Ochi, L. S., Vidal, T. and Prins, C. (2019). *A hybrid heuristic for a broad class of vehicle routing problems with heterogeneous fleet*, Annals of Operations Research, Vol. 273 (1-2), pp. 5–74.

Pinto, T., Alves, C. and Valério de Carvalho, J. (2020). *Variable neighborhood search algorithms for the vehicle routing problem with two-dimensional loading constraints and mixed linehauls and backhauls*, International Transactions in Operational Research, Vol. 27 (1), pp. 549–572.

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

Prais, M. and Ribeiro, C. C. (2000). *Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment*, INFORMS Journal on Computing, Vol. 12 (3), pp. 164–176.

Prins, C., Prodhon, C. and Calvo, R. W. (2006). *Solving the capacitated location-routing problem by a grasp complemented by a learning process and a path relinking*, 4OR, Vol. 4 (3), pp. 221–238.

Raft, O. M. (1982). *A modular algorithm for an extended vehicle scheduling problem*, European Journal of Operational Research, Vol. 11 (1), pp. 67–76.

Reimann, M. and Ulrich, H. (2006). *Comparing backhauling strategies in vehicle routing using ant colony optimization*, Central European Journal of Operations Research, Vol. 14 (2), pp. 105–123.

Renaud, J., Laporte, G. and Boctor, F. F. (1996). *A tabu search heuristic*

for the multi-depot vehicle routing problem, Computers & Operations Research, Vol. 23 (3), pp. 229–235.

Rincon-Garcia, N., Waterson, B. J. and Cherrett, T. J. (2018). *Requirements from vehicle routing software: perspectives from literature, developers and the freight industry*, Transport Reviews, Vol. 38 (1), pp. 117–138.

Ropke, S. and Pisinger, D. (2004). *A unified heuristic for a large class of Vehicle Routing Problems with Backhauls*. Technical report no. 2004/14, Department of Computer Science in University of Copenhagen. Copenhagen.

Royo, B., Fraile, A., Larrodé, E. and Muerza, V. (2016). *Route planning for a mixed delivery system in long distance transportation and comparison with pure delivery systems*, Journal of Computational and Applied Mathematics, Vol. 291, pp. 488 – 496.

Salhi, S. (2017). *Heuristic Search: The Emerging Science of Problem Solving*, 1st Edition, Cham: Palgrave Macmillan.

Salhi, S., Imran, A. and Wassan, N. A. (2014). *The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation*, Computers & Operations Research, Vol. 52, pp. 315–325.

Salhi, S. and Nagy, G. (1999). *A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling*, Journal of the Operational Research Society, Vol. 50 (10), pp. 1034–1042.

Salhi, S. and Sari, M. (1997). *A multi-level composite heuristic for the multi-depot vehicle fleet mix problem*, European Journal of Operational Research, Vol. 103 (1), pp. 95–112.

Salhi, S., Wassan, N. and Hajarat, M. (2013). *The Fleet Size and Mix Vehicle Routing Problem with Backhauls: Formulation and Set Partitioning-based Heuristics*, Transportation Research Part E: Logistics and Transportation Review, Vol. 56, pp. 22–35.

Sicilia, J. A., Quemada, C., Royo, B. and Escuín, D. (2016). *An optimization algorithm for solving the rich vehicle routing problem based on variable neighborhood search and tabu search metaheuristics*, Journal of Computational and Applied Mathematics, Vol. 291, pp. 468 – 477.

Sörensen, K., Sevaux, M. and Schittekat, P. (2008). *"Multiple Neighbourhood" Search*

in Commercial VRP Packages: Evolving Towards Self-Adaptive Methods, Heidelberg: Springer, pp. 239–253.

Sze, J. F., Salhi, S. and Wassan, N. (2016). *A hybridisation of adaptive variable neighbourhood search and large neighbourhood search: Application to the vehicle routing problem*, Expert Systems with Applications, Vol. 65, pp. 383 – 397.

Taillard, E. D. (1999). *A heuristic column generation method for the heterogeneous fleet VRP*, RAIRO - Operations Research, Vol. 33 (1), pp. 1–14.

Thangiah, S. R. and Salhi, S. (2001). *Genetic clustering: An adaptive heuristic for the multidepot vehicle routing problem*, Applied Artificial Intelligence, Vol. 15 (4), pp. 361–383.

Tillman, F. A. (1969). *The multiple terminal delivery problem with probabilistic demands*, Transportation Science, Vol. 3 (3), pp. 192–204.

Tlili, T. and Krichen, S. (2015). Decision Support System for the Multi-depot Vehicle Routing Problem, *in* H. A. Le Thi, T. Pham Dinh and N. T. Nguyen (eds), *Advances in Intelligent Systems and Computing*, Vol. 359 of *Advances in Intelligent Systems and Computing*, Cham: Springer International Publishing, pp. 56–63.

Toth, P. and Vigo, D. (1996). *A Heuristic Algorithm for the Vehicle Routing Problem with Backhauls*, 1st Edition, Berlin Heidelberg: Springer Berlin Heidelberg, pp. 585–608.

Toth, P. and Vigo, D. (1997). *An Exact Algorithm for the Vehicle Routing Problem with Backhauls*, Transportation Science, Vol. 31 (4), pp. 372–385.

Toth, P. and Vigo, D. (2002). *The Vehicle Routing Problem*, 1st Edition, Society for Industrial and Applied Mathematics.

Tu, W., Fang, Z., Li, Q., Shaw, S.-L. and Chen, B. (2014). *A bi-level Voronoi diagram-based metaheuristic for a large-scale multi-depot vehicle routing problem*, Transportation Research Part E: Logistics and Transportation Review, Vol. 61, pp. 84–97.

Tütüncü, G. Y. (2010). *An interactive GRAMPS algorithm for the heterogeneous fixed fleet vehicle routing problem with and without backhauls*, European Journal of Operational Research, Vol. 201 (2), pp. 593–600.

Tütüncü, G. Y., Carreto, C. and Baker, B. (2009). *A visual interactive*

approach to classical and mixed vehicle routing problems with backhauls, Omega, Vol. 37 (1), pp. 138–154.

Vidal, T., Crainic, T. G., Gendreau, M. and Prins, C. (2014). *Implicit depot assignments and rotations in vehicle routing heuristics*, European Journal of Operational Research, Vol. 237 (1), pp. 15–28.

Villegas, J. G., Prins, C., Prodhon, C., Medaglia, A. L. and Velasco, N. (2011). *A grasp with evolutionary path relinking for the truck and trailer routing problem*, Computers & Operations Research, Vol. 38 (9), pp. 1319 – 1334.

Wade, A. and Salhi, S. (2004). *An Ant System Algorithm for the Mixed Vehicle Routing Problem with Backhauls*, Boston MA: Springer US, pp. 699–719.

Wassan, N. (2007). *Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls*, The Journal of the Operational Research Society, Vol. 58 (12), pp. 1630–1641.

Wassan, N. and Nagy, G. (2014). *Vehicle routing problem with deliveries and pickups: Modelling issues and meta-heuristics solution approaches*, International Journal of Transportation, Vol. 2, pp. 95–110.

Wren, A. and Holliday, A. (1972). *Computer scheduling of vehicles from one or more depots to a number of delivery points*, Operational Research Quarterly (1970-1977), Vol. 23 (3), pp. 333–344.

Wu, W., Tian, Y. and Jin, T. (2016). *A label based ant colony algorithm for heterogeneous vehicle routing with mixed backhaul*, Applied Soft Computing, Vol. 47, pp. 224 – 234.

Xu, Y. and Jiang, W. (2014). *An improved variable neighborhood search algorithm for multi depot heterogeneous vehicle routing problem based on hybrid operators*, International Journal of Control and Automation, Vol. 7 (3), pp. 299 – 316.

Xu, Y., Wang, L. and Yang, Y. (2012). *A new variable neighborhood search algorithm for the multi depot heterogeneous vehicle routing problem with time windows*, Electronic Notes in Discrete Mathematics, Vol. 39, pp. 289 – 296.

Yellow, P. C. (1970). *A computational modification to the savings method of vehicle scheduling*, Journal of the Operational Research Society, Vol. 21 (2), pp. 281–283.

Yücenur, G. N. and Çetin Demirel, N. (2011). *A new geometric shape-based genetic*

clustering algorithm for the multi-depot vehicle routing problem, Expert Systems with Applications, Vol. 38 (9), pp. 11859 – 11865.

Zachariadis, E. E. and Kiranoudis, C. T. (2012). *An effective local search approach for the vehicle routing problem with backhauls*, Expert Systems with Applications, Vol. 39 (3), pp. 3174 – 3184.

Zhao, Q., Zhou, C. and Pedrielli, G. (2020). *A Decision Support System for Data-Driven Driver-Experience Augmented Vehicle Routing Problem*, Asia-Pacific Journal of Operational Research [Online], Vol. 37 (05). Available at: https://doi.org/10.1142/S0217595920500189 (Accessed: 15 September 2020).

VITA

Fatih Kocatürk received his undergraduate degree from Department of Mathematics at İzmir University of Economics in 2012. In the same year, he was accepted to Applied Mathematics and Statistics Integrated Ph.D. program at İzmir University of Economics and took part in a TÜBİTAK research project titled "Pharmacy duty scheduling problems" as a scholar. Later, the TÜBİTAK project on the doctorate subject was accepted and he worked as a scholarship student in this project. After the completion of the project, he worked as software development engineer in an e-commerce company, where he learned C# programming language and SQL Server. He developed the decision support system he proposed in his thesis using C# and SQL Server. For the last 3 years, he has been working as an R&D Engineer at Norm Cıvata, Turkey's largest fastener manufacturer. Here, he takes part in analytical model development, process optimization, computational software development and Industry 4.0 projects. The outputs of the studies are published as articles in various journals and as papers and presentations at international conferences.