Optimization of Road Freight Operations of a Third-party Logistics Carrier

by

Onur Can Saka

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Doctor of Philosophy

in

Industrial Engineering and Operations Management

KOÇ ÜNİVERSİTESİ

August 14, 2020

Optimization of Road Freight Operations of a

Third-party Logistics Carrier

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a doctoral dissertation by

Onur Can Saka

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Prof. Sibel SALMAN

_____

Prof. Metin TÜRKAY

_____

Prof. Necati ARAS

_____

Assist. Prof. Tuba YILMAZ GÖZBAŞI

_____

Assist. Prof. Barış YILDIZ

Date: _____

To my family…

ABSTRACT

Optimization of Road Freight Operations of a Third-party Logistics Carrier
Onur Can Saka
Doctor of Philosophy in Industrial Engineering and Operations Management
August 14, 2020

Planning of daily freight operations is one of the main challenges in transport logistics. Our study originates from a complex real-life routing problem faced by a third-party logistics (3PL) firm. Most of the 3PL firms employ a decentralized approach to ease the planning process and manage the complexity in their operations. This leads to suboptimal decisions due to the lack of integrated use of their resources as well as missed consolidation opportunities. The aim of this study is to develop a solution methodology to optimize the planning of day-to-day road transport operations of the firm in a centralized manner within an acceptable time frame.

We address three novel problems related to road freight operations in this thesis. We first examine the less-than-truckload (LTL) planning problem, which involves pickup and delivery orders with deadlines and cross-docking opportunities. We assume that an unlimited number of trucks of different capacities can be hired from the spot market on a short-term basis to fulfill the LTL transportation requests. The objective is to find a minimum cost assignment of orders to feasible route-vehicle pairs while satisfying practical constraints. We propose a decomposition-based matheuristic algorithm to tackle this problem. Our solution approach demonstrates a significant reduction in the freight costs over the long term.

The 3PL firm also operates a limited number of dedicated trucks which are either directly owned or hired through a long-term contract. The second problem we address deals with the assignment of dedicated vehicles to less-than-truckload (LTL) and full truckload (FTL) routes. The LTL routes obtained in the first part can be re-scheduled in this phase while the spot-hired vehicles are replaced with dedicated ones to maximize

savings via an *Mixed Integer Linear Programming* (MILP)-based solution approach.

We finally examine the last-mile delivery operations of the 3PL firm where different types of products are delivered to customers within a certain geographic region from a central depot. We model this part as a *Rich Vehicle Routing Problem* (RVRP) with heterogeneous vehicle fleet, time windows, selectiveness and several other practical requirements while allowing multiple trips per vehicle. We propose a parallelized hybrid evolutionary algorithm for solving the problem.

The contribution of our study is twofold. On one hand, it creates actual value for the business through reducing freight costs with the proposed solution techniques. On the other hand, it introduces novel problems and solution algorithms to the optimization literature which can be extended to a wide range of industry problems in the future.

ÖZETÇE

Üçüncü-parti bir Lojistik Taşıyıcısının Karayolu Nakliye Operasyonlarının Optimizasyonu

Onur Can Saka
Endüstri Mühendisliği ve İşletme Yönetimi, Doktora
14 Ağustos 2020

Nakliye operasyonlarının günlük planlaması, taşıma lojistiğindeki en büyük zorluklardan biridir. Bu çalışma, bir üçüncü-parti lojistik (3PL) firmasının karmaşık gerçek hayat rotalama probleminden ilham almıştır. 3PL şirketlerinin pek çoğu planlama süreçlerini kolaylaştırmak ve operasyonlarındaki zorlukları yönetebilmek için dağıtık bir yaklaşım benimsemektedir. Bu durum, kaynakların entegre biçimde kullanılamaması ve kaçan konsolidasyon fırsatları sebebiyle en iyiden uzak kararlara sebep olabilmektedir. Bu çalışmanın amacı, firmanın günlük karayolu taşıma operasyonları planlamasını merkezi bir biçimde ve kabul edilebilir bir zaman dilimi içerisinde eniyileyecek bir çözüm metodolojisi geliştirilmesidir.

Bu tezde karayolu nakliye operasyonları ile ilgili üç özgün problem ele alınmaktadır. İlk olarak toplama-bırakma siparişleri, teslimat terminleri ve çapraz sevkiyat olanakları içeren bir tam-kamyon-yükünden-az planlama problemi incelenmiştir. Tam-kamyon-yükünden-az taşıma taleplerini karşılamak için spot piyasadan farklı kapasitelerde sınırsız sayıda kamyonun kısa zamanlı olarak kiralanabileceği varsayılmıştır. Amaç, pratikte karşılaşılan kısıtlara uyumlu şekilde siparişleri uygun rota-araç çiftlerine en uygun maliyetle atayabilmektir. Problemin üstesinden gelebilmek için ayrıştırma bazlı bir mat-sezgisel algoritma önerilmiştir. Önerilen çözüm yaklaşımı, nakliye maliyetlerinde uzun vadede önemli oranda düşüş sağlamıştır.

3PL firması aynı zamanda doğrudan sahip olduğu ya da uzun dönemli bir sözleşme ile kiraladığı tahsisli kamyonları işletmektedir. Ele aldığımız ikinci problem, tahsisli araçların tam-kamyon-yükü ve tam-kamyon-yükünden-az araç rotalarına atanmasını konu almıştır. İlk kısımda elde edilen tam-kamyon-yükünden-az araç rotaları bu fazda

yeniden çizelgelenebilmekte ve bu rotalara atalı spot araçlar, tahsisli araçlar ile değiştirilerek *Karmaşık Tamsayılı Doğrusal Programlama* bazlı bir çözüm yaklaşımı ile yapılan tasarruf maksimize edilmektedir.

Son olarak, 3PL firmasının belirli bir coğrafi bölgede yer alan müşterilerine merkezi bir depodan çeşitli ürünleri teslim etmesini baz alan son mil teslimatı operasyonları incelenmiştir. Bu kısım, çoklu araç tipli, zaman pencereli, sipariş seçme opsiyonuna sahip ve pratikte karşılaşılan daha birçok gerekliliği içeren, araç başına birden fazla tur atma imkanı tanıyan bir *Zengin Araç Rotalama Problemi* olarak modellenmiştir. Problemi çözmek için paralelleştirilmiş bir hibrit evrimsel algoritma önerilmiştir.

Çalışmamızın katkısı iki yönlüdür. Bir yanda, önerilen çözüm teknikleri ile nakliye maliyetleri düşürülerek işletme için gerçek anlamda değer yaratılmıştır. Diğer yanda ise optimizasyon literatürüne, gelecekte geniş bir yelpazede sektör problemlerine uyarlanabilecek özgün problemler ve çözüm algoritmaları sunulmuştur.

ACKNOWLEDGMENTS

Undertaking this PhD has been a truly life-changing experience for me and it would not have been possible to do without the support and guidance that I received from many people.

I wish to express my sincere gratitude to my supervisor Prof. Sibel Salman for her guidance and constructive feedbacks throughout this study. She provided me with the tools that I needed to choose the right direction and successfully complete my dissertation. I would also like to thank to the rest of my thesis monitoring committee, Assist. Prof. Tuba Yılmaz Gözbaşı, Assist. Prof. Barış Yıldız and Prof. Emre Alper Yıldırım for their suggestions and valuable criticism.

I would like express my deepest appreciation to Varlık Kılıç, my supervisor at Borusan R&D, who provided me the opportunity to work on this project. I am also thankful to my teammates at Borusan R&D, Dilara Aykanat, Kanan Aghayev, Fırat Bozlak and Sinan Selamoğlu, who helped me putting the methodologies developed in this study into practice.

I would like to thank my dear wife Aylin Önder Saka for her endless support and affection. It would have been impossible for me to finalize this study without her kind help. I will always remember the patience and understanding she showed me during the toughest times.

Finally, I would like to express all my love to my parents Ayşegül Saka and Prof. Günay Saka for their support, guidance, and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

2E-VRP	Two-echelon VRP
3PL	Third-party Logistics
ACO	Ant Colony Optimization
ALNS	Adaptive Large Neighborhood Search
ALNS	Adaptive LNS
CGH	Column Generation-based Heuristic
CH	Centralized Heuristic
CVRP	Capacitated VRP
DARP	Dial-a-Ride Problem
DH	Decentralized Heuristic
DMPVRP	Dynamic MPVRP
DP	Dynamic Programming
DVAP	Dedicated Vehicle Assignment Problem
DVRP	Dynamic VRP
EVRP	Electric Vehicle Routing Problem
FSMVRP	Fleet Size and Mix VRP
FTL	Full Truckload
FTVRPP	Full-truckload Vehicle Routing Problem with Profits
GA	Genetic Algorithm
GA	Genetic Algorithms
GDP	Gross Domestic Product
GI	Greedy Insertion
GIN	Greedy Insertion with Noise
GIRS	Greedy Insertion with Random Selection
GPS	Global Positioning System
GVRP	Green Vehicle Routing Problem
HEA	Hybrid Evolutionary Algorithms
HVRP	Heterogeneous Fleet VRP
HVRPTW	Heterogeneous Fleet Vehicle Routing Problem with Time Windows
ILP	Integer Linear Programming
ILS	Iterated Local Search
IMP	Integer Master Problem
IP	Integer Programming
LAR	Latest Arrival Removal
LER	Largest Earliness Removal

LLR	Largest Lateness Removal
LMDP	Last-mile Delivery Planning Problem
LMP	Linear Master Problem
LNS	Large Neighborhood Search
LTL	Less-than Truckload
LTLP	LTL Planning Problem
LTLP-C	LTLP with Continuous Planning Horizon
LTLP-C-Sub	LTLP-C Sub-problem
LTLP-DM	LTLP with Discrete Planning Horizon and Multi-period Considerations
LTLP-DM-Subs	LTLP-DM Sub-problem
MA	Memetic Algorithms
MDR	Modified Distance Removal
MDVRP	Multi-Depot VRP
MEVRP	Multi-echelon VRP
MH	Metaheuristic
MILP	Mixed-Integer Programming
MPIH	Modified Potential Insertion Heuristic
MPVRP	Multi-period VRP
MTRVRP	Multi-trip Rich VRP
MTVRP	Multi-trip VRP
PDP	Pickup and Delivery Problem
PDPCD	PDP with Cross-Docking
PDPT	PDP with Transfers
PDP-T	PDP with Transshipment
PDPTW	PDP with Time Windows
PH	Pricing Heuristic
PIH	Potential Insertion Heuristic
PM	Pricing Model
PRP	Pollution-Routing Problem
PSO	Particle Swarm Optimization
PVRP	Periodic VRP
RI	Regret Insertion
RIMP	Restricted Integer Master Problem
RLMP	Restricted Linear Master Problem
RR	Random Removal
RVRP	Rich VRPs
SA	Simulated Annealing
SDVRP	Split-Delivery VRP
TDVRP	Time-Dependent VRP
TMS	Transportation Management Systems
TS	Tabu Search
TSP	Travelling Salesman Problem

VARR	Violation of Accessibility Restriction Removal
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
VRPB	VRP with Backhauls
VRPCD	VRP with Cross-docking
VRPFT	VRP with Full-Truckloads
VRPPD	VRP with Pickup and Deliveries
VRPSD	VRP with Stochastic Demands
VRPTW	VRP with Time Windows
WR	Worst Removal

Chapter 1

INTRODUCTION

Logistics industry constitutes approximately 10-15% of the total global GDP and is an integral part of the economy. Transportation is the largest component of total logistics expenditure with a share of more than 60% [1]. Any firm that operates in the real sector should plan its transportation operations efficiently to sustain competitive advantage. Many companies outsource their transport activities to third-party logistics (3PL) service providers. Using 3PL creates a cost advantage through economies of scale and reduction of capital investments for businesses [2].

This study originates from a complex real-world problem faced by one of the largest 3PL providers in Turkey, where 85% of domestic freight is handled by trucking and road transport [3]. The focus of our research is the domestic road transport operations of the company. However, we believe that the problems defined and solution methods proposed in this thesis would be applicable to many logistics service providers.

After conducting a thorough analysis of the less-than truckload (LTL), full truckload (FTL) and last-mile delivery operations of the 3PL carrier, we have identified three main operational level planning problems to be tackled. Although these problems are addressed separately throughout the thesis, we recognize the interdependent nature of the underlying processes and design our solution approaches so as to preserve the integrity of the overall problem.

In this introductory chapter, we first present our objectives and motivation for this study, which is followed by the demonstration of the contributions of our work. Finally, we provide an outline for the rest of the thesis.

1.1 Objectives and Motivation

Thanks to the recent advancements in information technology, most of the logistics firms have begun to manage their operations through software systems, which has become a crucial part of the logistics processes [4] all over the world. The management of freight operations is often carried out by Transportation Management Systems (TMS). While many TMS applications offer decision support capabilities for planning and optimizing transportation operations [5], a large number of companies still use spreadsheets and manual approaches for daily transportation planning. Each logistics company has its own dynamics and best practices developed over the years for dealing with operational planning problems. Therefore, integrating off-the-shelf software systems into the existing operations brings about many challenges and technical difficulties. On the other hand, planning freight operations manually is also a highly demanding task for logistics firms that operate within a complex transportation network due to the high number of critical decisions that need to be made on a daily basis. Most of the 3PL carriers employ a decentralized approach to ease the planning process to manage the complexity in their operations. This leads to suboptimal decisions due to lack of integrated use of their resources as well as missed consolidation opportunities.

Our main motivation for conducting this research is to identify critical problems in planning of road transport operations and develop efficient, generic and scalable solution methods to create value for a specific 3PL company while opening a new channel for future studies to incorporate and extend the problems and techniques presented in this study to solve other logistics carriers' planning challenges.

As we aim to generate both commercial and academic value through this research, we have set certain objectives to be achieved in line with our motivation.

i) Precise problem definition: Due to the high level of complexity in the existing operations, a comprehensive analysis of the system is required to accurately define the problems that will be addressed throughout this study by identifying their inputs, outputs, limitations and performance metrics correctly.

ii) Reduction of transportation costs: The solutions produced by the techniques developed in this study should be better than the existing solutions of the 3PL

carrier in terms of the overall transportation cost in order to create actual value for the business.

iii) Suitability for practical use: Any solution algorithm to be developed throughout the study should be sufficiently fast so that its outcomes can be implemented in real-life. Existing practical rules and constraints should be incorporated into the solution methods to ensure their applicability in the field. Major structural changes that may lead to impractical results should be avoided.

iv) Originality of problems and solution approaches: The problems we are going to deal with should be structured in a way that they have not been explicitly addressed in the literature before. Novel solution techniques that stem from state-of-the-art methodologies in the literature should be designed to establish a solid academic contribution.

v) Generality and scalability: Solution methods should be generic in a sense that they can be easily applied to less restrictive problems with similar type of objectives and constraints. The methods should be easily adaptable to larger problem instances that may be encountered in the future.

These principles are taken into serious consideration when defining problems or proposing solution approaches throughout the study. In the next section we highlight the contribution of this thesis from academic and business perspectives.

1.2 Contributions

In this thesis, we introduce three novel problems that are relevant to logistics service providers or any other organization that manages its road freight logistics operations in-house. The first problem is the planning of less-than truckload (LTL) operations, which applies to LTL carriers that fulfill transportation requests using trucks hired from the spot market. We model this as a pickup and delivery problem with cross-docking opportunities. We address two versions of the problem: The first one is a continuous-time version with synchronization requirements at the cross-dock facilities. The second version is a discrete-time, multi-period variant of the problem, which better fits the requirements of the 3PL carrier. While both versions are original in terms of incorporating a variety of practical rules and constraints, the second version introduces a new problem

class by addressing the pickup and delivery problem with cross-docking opportunities in a multi-period setting.

The second problem we aim to tackle is the assignment of dedicated vehicles to less-than truckload (LTL) and full truckload (FTL) routes. This problem not only extends the first problem by introducing the possibility of using dedicated vehicles, which are either owned by the company or hired on a long-term contract, but also it embodies the planning challenges of FTL carriers with a mixed fleet in terms of the ownership type of the vehicles.

The last problem we present is the daily planning of last-mile delivery operations, which is modeled as a rich vehicle routing problem (VRP). This problem fills a gap in the academic literature by introducing a new class of VRP with many realistic considerations that can easily be adapted to less restrictive variants of VRP.

Another important contribution of this thesis is that it proposes an overall solution strategy for planning the daily road freight operations of the 3PL company by solving instances of the abovementioned problems. Although each of these problems is manageable on its own and features its own scope, by solving these problems in a systematic way and allowing exchange of information between solution methods, the unity of the overall problem can be preserved.

We introduce a different way of modelling for the LTL planning problem compared to the existing approaches in the literature. Most researchers propose vehicle and commodity flow formulations or set partitioning models [6] for routing problems. We extend the set partitioning approach by incorporating the assignment of transportation orders to previously generated route-vehicle pairs. We also present a novel route construction algorithm, which produces the set of all admissible routes in accordance with the rules and limitations specified by the 3PL company.

Solution techniques developed for solving the problems addressed also constitute a contribution from an academic perspective. We propose a decomposition-based matheuristic approach for solving the LTL planning problems, which accommodates unique characteristics in terms of the overall algorithm structure and the use of mathematical programming solvers, heuristics and column generation techniques in a specialized manner for solving sub-problems. We devise a novel hybrid evolutionary algorithm, which combines the *Adaptive Large Neighborhood Search* (ALNS) method

with *Genetic Algorithm* (GA), to tackle the daily last-mile delivery problem. New algorithmic features are introduced to deal with a variety of practical elements specific to the problem. Both the matheuristic algorithm for solving the LTL planning problem and the hybrid evolutionary algorithm for solving the last-mile delivery problem are designed in a way to allow parallelization to obtain faster solutions through distributed computing when executed on a suitable hardware.

Computational experiments prove that the proposed solution methods create value for the business through a significant cost reduction compared to the current planning strategy, while the computation times are short enough to enable their usage in practice.

1.3 Outline of the Thesis

In this section we describe how the rest of the thesis is organized by briefly introducing the main topics included in the subsequent chapters.

In chapter 2, we present a literature review in which we highlight and categorize important research efforts devoted to similar types of problems to those addressed within this thesis.

Chapter 3 demonstrates the analysis of the current system and an overall solution design through a holistic view of the existing operations of the 3PL company and the methodologies that have been developed throughout this study.

In chapters 4, 5 and 6 we address the LTL planning problem, the dedicated vehicle assignment problem and the last-mile delivery planning problem, respectively. At each of these chapters we provide a formal definition and a mathematical programming formulation of the respective problem, describe the corresponding solution methodology in detail, and present the results of literature the computational experiments.

The final chapter summarizes the presented work, encapsulates our concluding remarks and concludes with our recommendations for future research opportunities.

Chapter 2

LITERATURE REVIEW

Operational level route planning and scheduling problems that arise in transportation logistics are often addressed under the heading of *Vehicle Routing Problems* (VRP) in the literature. VRP is one of the most popular combinatorial optimization problems and it has been extensively studied by researchers for more than sixty years. The first known work on VRP was proposed by Dantzig and Ramser [7] with the title "*The Truck Dispatching Problem*". Since then, a wide range of VRP variants have emerged and numerous models and solution techniques have been developed to tackle these problems.

In the first section of this chapter, we present an overview of VRPs by highlighting the important problem variants in the literature and discussing the modelling efforts and solution methods. In the second section, we focus specifically on problem variants that are similar to those addressed within this thesis.

2.1 Overview of the VRP Literature

VRP is the problem of finding minimum cost routes to visit a given set of customer points using vehicles that depart from and return to a certain depot location. It is a generalization of the well-known *Travelling Salesman Problem* (TSP), however it is much more challenging compared to TSP in terms of computational complexity [8].

Braekers et al. [9] present a taxonomic framework for the VRP literature, which is built upon the classification proposed by Eksioglu et al. [10]. The authors suggest several criteria to categorize the academic work on VRP such as the type of study, scenario, problem information and data characteristics. We will first review the main extensions of VRP which differ based on their scenario and problem characteristics. Then we will go over the most common approaches used for solving main variants of the VRP.

2.1.1 Main Extensions of the VRP

The *Capacitated VRP* (CVRP), which is also referred to as the *Classical VRP* [8], is the basic extension of the problem incorporating the vehicle capacity constraints, which ensure that the total demand of the customers visited in a single tour does not exceed the capacity of the vehicle.

One of the most commonly studied variants of VRP is *VRP with Time Windows* (VRPTW). In this problem, a time interval is specified for the visit to each customer. Time window constraints can be defined as hard or soft. Hard time windows cannot be violated at any cost, whereas soft time window violations are allowed but penalized in the objective function. Desrosiers et al. [11] and Solomon [12] are among the first authors to introduce this variant into VRP literature.

Another VRP variant addressed extensively in the literature is *Heterogeneous Fleet VRP* (HVRP), which is a generalization of CVRP by enabling the use of vehicles with varying capacities [13]. HVRP has different versions depending on whether the number of vehicles of each type is limited or not. An important extension to HVRP is the *Fleet Size and Mix VRP* (FSMVRP) [14]. This version of the problem incorporates a fixed vehicle cost in addition to the variable travel costs and aims to determine the number of vehicles to be used for each vehicle type as well as the routes of each vehicle.

VRP with Backhauls (VRPB), is a general case of VRP in which backhaul customers are present in addition to the regular linehaul customers [15]. While items are picked up from the depot and delivered to the linehaul customers, items are collected from the customer location and shipped to the depot in case of backhauls.

While most versions of VRP assume that there exists a single depot where all routes originate from, the *Multi-Depot VRP* (MDVRP) introduces multiple depots, each having the ability to serve customers [16]. The resulting problem is not only finding the sequence of customers that will be visited by each vehicle, but also deciding which depot will serve each customer.

VRP with Pickup and Deliveries (VRPPD), which is also referred to as the *Pickup and Delivery Problem* (PDP) is another important extension of the VRP. In the classical VRP, each customer is served from a central depot. However PDP involves transportation requests from a pickup location to a delivery location, and the pickup locations may be different for each request [17]. A special case of PDP is the *Dial-a-Ride Problem* (DARP)

where the commodities represent people. Cordeau and Laporte [18] present an overview of the models and algorithms proposed for the DARP.

The *Multi-period VRP* (MPVRP) aims to generate optimal vehicle routes to visit customers over a planning horizon of several periods. The *Periodic VRP* (PVRP) is a generalization of MPVRP such that each customer may require more than one visit over the planning period. PVRP involves decisions regarding the selection of days to serve a certain customer and the routing of vehicles for each day within the planning horizon [19].

The regular VRP assumes that the vehicles perform a single trip during the planning period. The *Multi-trip VRP* (MTVRP) [20], also called as *VRP with Multiple Use of Vehicles* [21] allows the usage of vehicles multiple times a day. This extension increases the utilization of available operating time for each vehicle especially in a problem setting where short tours are possible.

Since the beginning of the 21st century, researchers have incorporated environmental concerns into VRP variants in line with the increasing environmental awareness of the society. Several VRP extensions have been proposed with the aim of reducing fuel consumption and CO_2 emissions along with the routing decisions. Some important extensions are the *Green Vehicle Routing Problem* (GVRP) [22], [23], the *Pollution-Routing Problem* (PRP) [24], [25] and the *Electric Vehicle Routing Problem* (EVRP) [26], [27].

Numerous other practical problem characteristics have been added to VRPs in the last decades, resulting in a diverse range of problem variants. The most noteworthy ones are the *Time-Dependent VRP* (TDVRP) [28], *VRP with Stochastic Demands* (VRPSD) [29], the *Split-Delivery VRP* (SDVRP) [30] and the *Dynamic VRP* (DVRP) [31].

While all the presented VRP extensions have been suggested for increasing the practical coverage, most of the real-life problems require multiple characteristics applied at the same time. In the VRP literature, the problems dealing with multiple realistic conditions are classified as *Rich VRP*s (RVRP). RVRP is formally defined as a generalization or union of other independent VRPs [32]. Majority of the recent studies on VRP fall into the rich category. Likewise, the problems addressed in this thesis display rich characteristics.

2.1.2 Modelling and Solution Approaches for the VRP

VRPs are often represented by *Integer Linear Programming* (ILP) or *Mixed-Integer Programming* (MILP) formulations in the literature. Magnanti [33] suggests that three main types of ILP formulations are available in the VRP literature:

- Vehicle flow models
- Commodity flow models
- Set partitioning models

Vehicle flow formulations are the most commonly used type of models for VRPs. In general, two-index or three-index vehicle flow variables are used in this type of models. In two-index formulations, binary flow variable x_{ij} represents whether a vehicle travels from node i to node j and in three-index formulations, a new dimension k is used for differentiating vehicles or vehicle types. The following simple two-index model of Toth and Vigo [34] is a basic illustration of vehicle flow models:

$$Minimize \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \tag{2.1}$$

$$subject\ to$$

$$\sum_{i \in N} x_{ij} = 1 \qquad \forall j \in N \setminus \{0\} \tag{2.2}$$

$$\sum_{j \in N} x_{ij} = 1 \qquad \forall i \in N \setminus \{0\} \tag{2.3}$$

$$\sum_{i \in N} x_{i0} = K \tag{2.4}$$

$$\sum_{j \in N} x_{0j} = K \tag{2.5}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S) \qquad \forall S \subseteq N \setminus \{0\}, S \neq \emptyset \tag{2.6}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in N \tag{2.7}$$

x_{ij} takes the value of one if arc (i, j) is used in the solution, and zero, otherwise. c_{ij} is the cost associated with traveling from node i to node j. N is the set of all vertices and

K denotes the number of vehicles. Constraints (2.2) and (2.3) ensure that each node has exactly one entering and leaving arc. Constraints (2.4) and (2.5) impose the degree requirements for the depot vertex. Given that $r(S)$ is the minimum number of vehicles to serve a given customer set S, the capacity-cut constraints (2.6) satisfy both the connectivity of the solution and the vehicle capacity requirements.

Kulkarni and Bhave [35] and Laporte et al. [36] present alternative two-index models for the VRP. Basic three-index models incorporating heterogeneous vehicle fleet are proposed by Golden et al. [37] and Fisher and Jaikumar [38].

Commodity flow models extend the vehicle flow models by the addition of decision variables associated with the amount of load carried on each arc. Gavish and Graves [39] present an early example of commodity flow formulations for the VRP. A more recent example of commodity flow models for the VRP can be found in [40].

Balinski and Quandt [41] introduced the idea of using set partitioning models to solve VRPs. Set partitioning models require that the set of feasible routes (J) is generated beforehand. In such models, each feasible route is denoted by a binary decision variable, y_j where the cost of route j is c_j. Binary parameter a_{ij} takes the value of one if node $i \in N \setminus \{0\}$ belongs to route j, and zero, otherwise. The resulting ILP formulation is as follows:

$$Minimize \quad \sum_{j \in J} c_j y_j \tag{2.8}$$

$$subject\ to$$

$$\sum_{j \in J} a_{ij} y_j = 1 \qquad \forall i \in N \setminus \{0\} \tag{2.9}$$

$$y_j \in \{0, 1\} \qquad \forall j \in J \tag{2.10}$$

The objective of the model (2.8) is to minimize to total cost of routes used while constraint (2.9) ensures that each node is visited by exactly one route. Laporte [42] states that it is very difficult to generate the complete set of feasible routes since the potential number of feasible routes grow quickly as the problem size gets larger. Even if the set of all feasible routes is available, it would be computationally very difficult to solve them due to the high number of binary variables. In order to overcome these issues, many

authors have employed a column generation approach to start with a small set of feasible routes, and add variables associated with promising routes iteratively.

Both exact and heuristic solution techniques have been developed over the years to tackle different versions of VRP. Due to the computational complexity of the problem, exact solution approaches have not been able to solve practical instances with over 100 customer nodes optimally, but these techniques have been useful in solving small instances, computing lower bounds and finding good upper bounds through hybrid approaches in combination with heuristics. Toth and Vigo [43] and Laporte [8] present several exact approaches proposed for solving the VRP.

Early examples of exact approaches are direct tree search methods, in which routes are generated sequentially via branch and bound trees [44]. Laporte et al. [45] propose an assignment based lower bound and a related branch and bound algorithm for asymmetric capacitated VRPs whereas Christofides et al. [46] present tree search algorithms incorporating lower bounds computed from shortest spanning k-degree centre trees and q-routes. Hadjiconstantinou et al. [47] improve the solutions using these lower bounds.

Dynamic Programming (DP) was also an exact solution approach used in the early days of VRP research. Eilon et al. [48] were the first authors to use DP for solving VRPs. Their method suffers from the curse of dimensionality due to the growth of the state space. Christofides et al. [49] provide a state-space relaxation procedure for reducing the number of states. DP-based approaches are usually preferred in case of dynamic and stochastic demands and related examples can be found in [50], [51] and [52].

Many researchers have used column generation-based techniques to solve different variants of VRP. A column generation algorithm aims to solve the linear relaxation of the set partitioning formulation of the problem, and it can be embedded within a branch-and-bound scheme to obtain an exact solution through the branch-and-price technique. Example applications of branch-and-price technique to solve VRPs can be found in [53], [54], [55] and [56].

Another exact approach used for addressing VRPs is branch-and-cut, which tries to strengthen the linear relaxations of the problems associated with the nodes of the branch-and-bound tree through adding valid linear inequalities. Naddef and Rinaldi [57] and Lysgaard et al. [58] propose branch-and-cut algorithms for solving CVRP while Bard et al. [59] solve VRPTW using this technique.

Branch-and-price and branch-and-cut techniques can also be combined to form the method known as branch-and-price-and-cut. Notable applications of branch-and-price-and-cut to certain extensions of the VRP can be found in [60], [61], [62], [63] and [64].

Heuristics are more commonly used for addressing VRPs thanks to their ability to find high quality solutions to large problem instances much faster compared to the exact methods. Laporte et al. [65] categorize the heuristic algorithms for the VRP as follows:

- Constructive heuristics
- Classical improvement heuristics
- Metaheuristics
- Hybridizations
- Unified algorithms

Constructive heuristics are the first heuristic efforts to find good feasible solutions to the practical-size VRPs. The earliest instance of constructive heuristics is the savings algorithm developed by Clarke and Wright [13]. The algorithm is initialized with the assumption that each customer is served by a separate vehicle. The routes are iteratively merged to obtain longer routes throughout the algorithm, selecting the combination of routes with the highest potential cost saving at each step. The savings heuristic has been improved and adapted to other variants of VRP later on as shown in [37], [12], [66] and [67].

Petal algorithms, also known as set partitioning heuristics, are constructive methods that aim to generate feasible routes and combining them by solving a set partitioning problem. An early example of this method can be observed in the sweep-based algorithm [68]. Foster and Ryan [69] and Renaud et al. [70] also employ this approach in their work.

Other well-known constructive techniques for VRP are the cluster-first, route-second method [38] and its reversed version, route-first, cluster-second method [71], which is also called as the giant tour heuristic. Both approaches are two-phase methods that aim to find feasible solutions to VRP in a constructive manner.

In modern solution methodologies, constructive methods are usually employed for generating quick initial solutions for further improvement. Classical improvement

heuristics generate simple neighborhood structures by performing modifications on the existing solutions [72]. Intra-route and inter-route movements are the most commonly used operators that are applied to improve a given VRP solution. Classical improvement techniques are also employed within more advanced techniques for solving different variants of the VRP.

Metaheuristics are the most popular solution approaches designed to tackle the VRP and its extensions. A metaheuristic is an improvement algorithm which coordinates sub-heuristics with different capabilities to utilize higher quality search opportunities [72]. Laporte et al. [65] suggest that metaheuristics can be classified into two main categories: Local search algorithms and population-based algorithms. Local search methods employ certain neighborhood search operators to jump from one solution to another with the aim of improving the solution iteratively until a given stopping criterion is met. *Simulated Annealing* (SA), *Tabu Search* (TS), *Iterated Local Search* (ILS) and *Variable Neighborhood Search* (VNS) are among the local search methods often used to address VRPs.

Simulated annealing (SA) approach has a temperature parameter, which designates the probability of accepting or rejecting a new non-improving solution. At the beginning of the algorithm, the temperature of the system is high and so is the probability of accepting worsening solutions. As the algorithm proceeds the system cools down at a given rate, the system temperature decreases and accepting non-improving solutions become less likely; therefore, the solution converges to a local minimum. Readers may refer to Osman [73] for a well-known application of SA to the VRP.

Tabu search (TS) has been one of the most popular metaheuristic methods among VRP researchers. The main idea of Tis to keep solutions sharing similar characteristics with the active solution in a tabu list for a certain number of iterations and keep the algorithm from moving to solutions in the tabu list in order to avoid cycling. TS practices that have made a major impact in VRP literature can be found in [74], [75], [76], [77] and [78].

Iterated local search (ILS) employs an embedded local search procedure (e.g. TS) which returns a local minimum solution given a starting solution. At the end of each outer iteration, the solution found by the local search method is perturbed (or shaken) and the obtained new solution becomes the starting solution for the next iteration. This procedure

continues until a specified stopping criterion is met. An application of ILS to CVRP can be found in [79].

Variable Neighborhood Search (VNS) algorithm contains several neighborhood structures which are applied sequentially to a given initial solution. The final solution found at the end of this procedure becomes the initial solution for the next iteration. The algorithm runs until any of the predetermined stopping conditions are satisfied. Kytöjoki et al. [80] successfully applied this method to large-scale instances of VRP and report good feasible solutions obtained in reasonable computation times.

While local search methods usually work with small neighborhoods, some authors have developed methods which exploit large neighborhood structures. This type of approach is referred to as *Large Neighborhood Search* (LNS). LNS was first introduced by Shaw [81]. Pisinger and Ropke applied this technique to VRPs [82]. LNS constructs a large neighborhood through destroy and repair operators. Destroy operator removes certain customers from their positions and repair operator re-inserts them into the solution. Different types of destroy and repair operators can be used to diversify the search. The most important extension to LNS is the incorporation of adaptiveness, establishing the well-known metaheuristic technique called the *Adaptive LNS* (ALNS). Ropke and Pisinger [83] proposed the ALNS method and implemented it to *PDP with Time Windows* (PDPTW). In the ALNS, each destroy or repair operator has a certain selection probability at each iteration. The probability associated with each operator is periodically updated at certain steps of the algorithm based on their past success.

Population-based algorithms are nature inspired methods which make use of diversification mechanisms to escape from local optimum solutions. These techniques often include embedded local search procedures for exploiting the promising solutions. *Genetic Algorithms* (GA) are one of the population-based methods that have been extensively used in VRP literature to produce high quality solutions. In an arbitrary GA application, solutions are modeled as chromosomes and the set of chromosomes form a population. Chromosomes evolve over generations through genetic operators such as crossovers and mutated by randomized techniques or local search procedures. At each generation, improving offspring generated from parent solutions replace the worst elements of the population which increases the overall fitness of the population as the algorithm proceeds. Baker and Ayechew [84], Berger and Barkaoui [85], Prins [86],

Mester and Bräysy [87] and Nagata [88] present an important stream of work on application of GA to VRPs. *Ant Colony Optimization* (ACO) (see Reimann et al. [89]) and *Particle Swarm Optimization* (PSO) (see Ai and Kachitvichyanukul [90]) are other important population-based heuristic methods applied to VRPs.

Hybrid methods that combine different algorithmic approaches have gained popularity in the recent years. Common types of hybridizations are the ones integrating population-based methods with local search or metaheuristics, metaheuristics with other types of metaheuristics and heuristics with mathematical programming solvers. Hybridization of GA with other heuristics are named as *Memetic Algorithms* (MA) [91] or *Hybrid Evolutionary Algorithms* (HEA) [92]. A heuristic algorithm that utilizes mathematical programming models for solving certain sub-problems is called a matheuristic. Archetti and Speranza [93] present a classification of matheuristic approaches for routing problems.

Unified algorithms are flexible methods that can solve multiple variants of the VRP effectively [65]. These algorithms are especially useful for people or organizations who wish to develop generic VRP algorithms that can function properly for different types of operations.

Another recent advancement in heuristic methods for VRPs is hyper-heuristics. Hyper-heuristics operate in two ways: They either select a proper combination of sub-heuristics to be applied to a given problem instance, or they generate a new heuristic using the existing sub-heuristics through learning mechanisms [94].

In the next sub-section, we focus on the problems in the literature that share similar characteristics with the problems that are addressed in this study.

2.2 Review of the Relevant Literature

The problems we introduce in this thesis can be classified as rich problems since they combine characteristics of several distinct problem variants addressed in the literature. This sub-section provides a more detailed review of the literature related to the problems that we discuss in the rest of this thesis. We review prominent studies that are relevant to the overall system that we analyze, as well as the individual problems that we solve.

2.2.1 Network Design Problems

Transportation planning for LTL networks is performed in strategic, tactical and operational levels. Strategic level plans are mainly related to structural changes such as opening or closing facilities. Tactical level planning problems are extensively represented by network design models. The term service network design is used by Crainic [95] for tactical level decisions such as service selection, traffic distribution, terminal policies and empty balancing strategies.

Some notable efforts for modeling and solving network design problems for freight transportation are given as follows: Hewitt et al. [96] propose a hybrid technique which combines mathematical programming algorithms with heuristic methods to solve the capacitated fixed-charge network flow problem. Crainic et al. [97] present a simplex-based tabu search method to solve large instances of capacitated network design problem with many commodities. Ghamlouche et al. [98] suggest a neighborhood search mechanism using a shortest-pathlike network optimization procedure to identify promising moves for this problem. Crainic et al. [99] use Lagrangean perturbation scheme within a scope scaling heuristic to provide better solutions on larger and more difficult instances. Hewitt et al. [100] employ column generation approach to generate problem restrictions and produce high-quality solutions quickly with an exact algorithm. Barcos et al. [101] propose a metaheuristic algorithm based on ant colony optimization techniques for LTL routing design. Jarrah et al. [102] use decomposition technique to fragment the main network model into separate integer programming models for each destination terminal and solve these models efficiently in coordination with the master problem.

Load planning, a fundamental concept in service network design in an LTL environment, determines how the loads are consolidated along paths. Powell and Sheffi [103] describe the load planning problem and propose a local improvement heuristic. Erera et al. [104] suggest a path-based formulation with time and space considerations for the LTL load planning problem and solve it via an integer programming-based local search algorithm. Lindsey et al. [105] improve the algorithm of Erera et al. [104] by changing the neighborhood structure and obtain better results. More recently, Yamada and Febri [106] propose a solution approach to freight transport network design problem

using the particle swarm optimization (PSO) technique. Most of the studies in this area focus on tactical level decisions whereas we concentrate on LTL operations.

2.2.2 VRP with Cross-docking

Cross-docking is an important aspect of the LTL planning problem since the efficiency of the transportation plan is highly dependent on the consolidation of shipments at cross-dock facilities. VRP and cross-docking are two subjects that have been intensely, but separately studied in the literature. One of the first studies that combine these two concepts is proposed by Lee et al. [107], who define the problem for a single cross-dock center and assume that pickup and delivery routes are distinct. The authors provide a mathematical programming formulation for this problem and solve it via a tabu search (TS) heuristic. Liao et al. [108] suggest a different TS algorithm for the same problem and report better results than Lee et al. [107] in terms of both solution quality and computation time. Tarantilis [109] presents an adaptive multi-start TS algorithm and further improves the results. Santos et al. [110] enhances the problem such that some vehicles are allowed to fulfill pickup and delivery requests within the same tour without having to stop at the cross-dock center. They solve the problem with a branch-and-price algorithm. Morais et al. [111] propose an iterative local search (ILS) algorithm for the case with time windows imposed on pickup and delivery requests and bring about better results compared to the best-known solutions in the literature. In a recent study, Grangier et al. [112] introduce a matheuristic algorithm based on large neighborhood search and find new best-known solutions to several instances in the literature.

2.2.3 Multi-Echelon Distribution Logistics

The LTL network we consider has a multi-echelon structure where long-haul and short-haul operations act interdependently. According to the description of Schmid et al. [113], in multi-echelon distribution systems, loads are collected from suppliers and brought into hub locations. Long-haul transportation takes place between hubs. Once the goods arrive at the destination hubs, they are distributed by smaller vehicles to their final delivery locations. Gonzales-Feliu [114] provides a comprehensive literature overview on multi-echelon systems. The author defines an echelon as the elementary organization unit of a

supply chain and claims that an N-echelon network can be decomposed into N separate echelons that are linked via intermediary facilities. A set-partitioning based mathematical programming model is presented for the defined problem. A special case of multi-echelon distribution problems is the *Multi-echelon VRP* (MEVRP). This problem is closely related with the *VRP with Cross-docking* (VRPCD) in the sense that there exists an extra layer for consolidation between pickup and delivery requests. The majority of academic attention in the literature of multi-echelon distribution systems is paid to *Two-echelon VRP* (2E-VRP), which is a special case of MEVRP. For a detailed analysis of work on this area one may refer to the survey by Cuda et al. [115]. Exact, heuristic and hybrid methods have been developed for solving 2E-VRP. Baldacci et al. [116] and Jepsen et al. [117] propose exact solution algorithms to solve this problem, whereas Perboli et al. [118] suggest a hybrid mathematical model-based heuristic. Notable pure heuristic algorithms for this problem are presented by Crainic et al. [119] and Hemmelmayr et al. [120]. The LTL planning problem we address has a three-echelon network, and the shipments may go through two cross-docking operations while direct shipment and shipment through a single cross-dock are also possible.

2.2.4 Synchronization in Routing Problems

Synchronization is a critical issue that needs to be addressed in multi-echelon and cross-docking systems. Synchronization of both loads and vehicles should be maintained in order to ensure the feasibility of solutions. Many authors have focused on the synchronization aspects of VRP recently. Drexl [121] presents a comprehensive survey of VRPs with synchronization requirements and introduces different types of synchronization issues experienced in routing and scheduling problems. Neves-Moreira et al. [122] focus on the synchronization of resources allowing exchange of trailers between tractors and achieve good quality solutions in a reasonable amount of computation time via a mathematical programming based heuristic. Gschwind [123] incorporates synchronization into the *Pickup and Delivery Problem* (PDP) and solves the problem by four different branch-and-cut-and-price algorithms. Grimault et al. [124] apply the same concept to PDP with full truckloads and propose an ALNS algorithm.

2.2.5 Pickup and Delivery Problem

PDP [125] is another relevant problem class due to the pickup and delivery orders present in our system. *PDP with Time Windows* (PDPTW) is a variant of PDP in which time windows are imposed on pickups and/or deliveries, as in our problem. Dumas et al. [126] were the first ones to define PDPTW. They present a column generation based exact algorithm for this problem allowing multiple depots and multiple vehicle types. Up to now, many exact and heuristic solution methods have been developed by researchers to solve the PDPTW. Novel exact algorithms are proposed by Ropke et al. [127] and Ropke and Cordeau [128]. Notable heuristic methods presented so far can be found in Ropke and Pisinger [129], Nanry and Barnes [130] , Li and Lim [131] , and Bent and Hentenryck [132].

The LTL planning problem that we solve combines PDP with cross-docking in a three-echelon environment. Nikolopoulou et al. [133] demonstrate a comparative analysis of PDP and VRPCD and deduce that each approach has advantages and disadvantages depending on the structure of the problem network. The authors also propose a hybrid model which outperforms both approaches. The problems that integrate PDP and cross-docking appear under different names in the literature such as *PDP with Transfers* (PDPT), *PDP with Transshipment* (PDP-T) and *PDP with Cross-Docking* (PDPCD). Mitrović-Minić and Laporte [134] introduce the idea of incorporating transshipment points into PDPs and show the value created through this approach. Cortés et al. [135], Santos et al. [110] and Rais et al. [136] propose mathematical programming formulations for PDPT. Cortés et al. [135] solve the problem via Benders decomposition method whereas Santos et al. [110] employ a branch-and-price algorithm. Petersen and Ropke [137] and Masson et al. [138] use metaheuristic approaches to solve the PDPT.

2.2.6 The Multi-Period VRP

Our solution methodology is designed in such a way to produce LTL transportation plans over a multi-day planning environment. In this context, our work shares common characteristics with the *Multi-Period VRP* (MPVRP). Most of the authors who have worked on MPVRP aim to generate solutions for multiple days within their multi-period planning horizon whereas our approach is to solve daily planning problems that consider

postponing some pickup and delivery requests to following days within the due dates since future orders are not known in the real LTL planning system that we focus on. Wen et al. [139] introduce the Dynamic MPVRP (DMPVRP) which incorporates newly arriving customer orders on each day, similar to the problem we are dealing with. They present the Mixed Integer Linear Programming (MILP) formulation of the problem and solve it through a three-phase heuristic over a rolling planning horizon. As in our approach, Wen et al. [139] allow unfulfilled orders with a penalty that increases over time. The authors also try to balance the daily workload in addition to minimizing the travel costs and customer waiting time, which adds a multi-objective character to their problem. Albareda-Sambola et al. [140] extend DMPVRP by incorporating probabilistic information for customer requests. They propose an adaptive policy to determine when to serve a certain customer. Archetti et al. [141] represent alternative formulations for MPVRP with due dates. They add valid inequalities and solve these formulations via the branch-and-cut technique. They show the potential savings that can be achieved by relaxing the due dates. Mancini [142] and Dayarian et al. [143] both propose an Adaptive Large Neighborhood Search (ALNS) for solving MPVRP while Mancini [142] tackles a variant of the problem with multiple depots and heterogeneous vehicle fleet. Yu et al. [144] study the multi-period cross-dock distribution problem, which involves the transportation of goods from manufacturers to customers through cross-dock terminal. They present a PSO algorithm to solve single-period and multi-period versions of their problem and compare their results against MILP solutions for small instances.

2.2.7 VRP with Full-Truckloads

The dedicated vehicle assignment problem for LTL and FTL routes that we solve shares common characteristics with the *VRP with Full-Truckloads* (VRPFT). In VRPFT, each transportation order is a full-truck service request between designated origin and destination locations. Most variants of VRPFT also incorporate pickup and delivery time windows. Percentage of empty trailer movements is either directly or indirectly included in the objective function.

In most of the papers on VRPFT, it is assumed that there exists multiple depots in the problem network and each vehicle has a base depot. In this sense, the problem is similar

to MDVRP. Usually, it is required that the vehicles return to their base depots upon completing their routes.

Doerner et al. [145] propose a hybrid ACO algorithm to solve the full truckload transportation problem with a homogeneous fixed fleet and time windows. Arunapuram et al. [146] suggest an exact solution method with a column generation framework which can easily handle business constraints such as pickup time windows. Gronalt et al. [147] provide a mathematical programming formulation that explicitly considers pickup and deliveries with time windows and solve a relaxed version of their model to obtain lower bounds. They present a savings based heuristic which produces solutions in a short time period and compare their results with the lower bounds. Liu et al. [148] solve a variant of this problem incorporating collaborating transport opportunities between carriers via a two-phase heuristic algorithm.

In the case of our problem, any transportation request that is not assigned to a dedicated vehicle is assumed to be fulfilled by a truck hired from the spot market. Since it is cheaper to use dedicated vehicles, each re-assignment is considered to generate a saving. In this respect, our work displays similarities with the *Full-truckload Vehicle Routing Problem with Profits* (FTVRPP). Li and Lu [149] address this problem and propose a hybrid genetic algorithm to solve small instances of a logistics company.

2.2.8 The Multi-Trip VRP

The last-mile delivery problem we address in this thesis is a multi-trip rich VRP which possesses several practical considerations. The *Multi-trip VRP* (MTVRP) is an extension of VRP which incorporates the use of vehicles multiple times within the planning horizon. The first known effort to define and solve MTVRP was in 1990, by Fleischmann [150] for the VRP with heterogeneous fleet and time windows. Although other variants of VRP have been studied extensively ever since, the number of researchers that addressed multiple trips in VRP is quite limited. Cattaruzza et al. [151] present a comprehensive survey on MTVRPs. They provide an overview of mathematical formulations and solution approaches developed for MTVRP and its extensions.

Both exact and heuristic solution approaches have been proposed to solve variants of MTVRP. Azi et al. [152] and Hernandez et al. [153] attempt to solve the problem by branch-and-price algorithms. Azi et al. [152] report solutions to modified version of

Solomon's test instances with 25 and 40 customers. Hernandez et al. [153] use the same set of instances to test their algorithm. They solve more instances than Azi et al. [152] in significantly lower solution times.

Heuristic algorithms have been employed to solve larger instances of MTVRP. Petch and Salhi [154] propose a multi-phase constructive heuristic whereas Despaux and Basterrech [155] introduce a local search algorithm within a simulated annealing framework and solve instances with up to 100 customers. Seixas and Mendes [156] solve MTVRP via tabu search and justify the quality of their heuristic solutions by comparing their results with lower bounds obtained by column generation. Evolutionary algorithms are among commonly used methods for solving MTVRP. Anggodo et al. [157] propose a genetic algorithm for an application of MTVRP to optimization of tourist routes. Ayadi and Benadada [158] and Cattaruzza et al. [159] implement memetic algorithms, in which they incorporate local search procedures into their population-based methods.

An extension to MTVRP is presented by Alonso et al. [160] where they combine periodic VRP with multiple use of vehicles. Nguyen et al. [161] and Crainic et al. [162] solve a special class of MTVRP in which customers are divided into zones with distinct supply points and each trip takes place within single zone.

In this chapter we presented an overview of the VRP literature and a review of the relevant VRP variants. The next chapter demonstrates the detailed analysis of the current system and a design proposal for its future state.

Chapter 3

SYSTEM ANALYSIS AND SOLUTION DESIGN

This chapter presents an analysis of the current road transport system of the 3PL company, a more detailed description of the problems and proposes a general solution approach for the overall system by integrating the techniques designed for the three main identified problems.

At the beginning of the chapter, we provide the essential terminology and briefly describe the main system components that are extensively referred to throughout the thesis. Afterwards, we characterize the network structure, which is followed by a demonstration of the type of routes that are typically employed in the system and a description of the existing planning processes. We finally provide a general solution strategy that can be implemented to transform the planning approach of the overall system.

3.1 Main Components of the System

In this section we define the basic system elements. By doing so, we also aim to familiarize the readers with the terminology that will be used in the rest of the thesis. We cover several network elements as well as certain inputs and outputs of the problems.

3.1.1 Location

Locations are classified as pickup and delivery locations, cross-dock terminals, depots and vehicle home locations. Any location may fall into more than one category depending on the problem instance. Transfer operations take place at the cross-dock terminals. Depots represent starting locations of dedicated vehicles, however most of the depots also function as pickup or cross-dock terminals. Vehicle home locations are designated locations for dedicated vehicles to return upon completing their tasks. Latitude and

longitude of each location is known. Road distance and duration between each pair of locations are available. Locations are also associated with non-negative loading and unloading times.

3.1.2 Zone

The planning region is divided into a number of zones. Typically, each zone accommodates one cross-dock terminal. Each location belongs to a certain zone and location-zone assignments are known.

3.1.3 Route

A route is simply represented as a sequence of visited locations in a single trip in the LTL planning problem. In the context of the last-mile delivery planning problem, a route is a collection of trips performed by the same vehicle on the same day. Total distance and duration of a route can be calculated using the distance and time matrices, as well as loading and unloading times. Routes used in the system exhibit certain characteristics due to practical considerations. We categorize all the routes based on their structures when we introduce the concept of route types.

3.1.4 Order

An order is simply a transportation request for certain items between designated origin and destination locations. Therefore, there is a pickup terminal and a delivery terminal associated with each order. In the system we address, total size of orders is represented in terms of the total volume of its items in cubic meters. However, the models and approaches we present can be easily adapted to multiple size units such as weight or quantity. While LTL and FTL orders have due times that can be converted to due dates, last-mile orders have delivery time windows. FTL orders are full-truck transportation requests and they cannot be combined with other orders in a single vehicle.

3.1.5 Vehicle

Vehicles are the entities which perform the transportation activities. Vehicles used in the system are usually trucks. Vehicles are differentiated based on their ownership types and

physical characteristics. Vehicles owned by the 3PL company or hired on a long-term contract are called as “dedicated vehicles”. The company can also rent trucks from the spot market on a daily basis. Such trucks are named as “spot vehicles”. Physical characteristics determine the capacity of the vehicle in terms of volume. Both the ownership type and physical type of the vehicles has an impact on the vehicle-based costs. Fixed costs, fuel costs and extra costs such as cross-docking costs changes from one vehicle type to another. Dedicated vehicles are in general cheaper to use, but they are limited in number. We assume that an unlimited number of spot vehicles can be used to fulfill customer requests at higher costs. Dedicated vehicles have designated home locations. We assume that the initial locations of the dedicated vehicles can be detected through GPS devices, and these vehicles must return to their home locations within a given time period.

3.1.6 Route-vehicle Pair

One of the main inputs of the LTL planning problem is the set of feasible route-vehicle pairs, which we generate beforehand through a constructive algorithm based on the characteristics of the route types. A route-vehicle pair entity exhibits properties of the route and the vehicle it is composed of. Therefore, it is possible to assign a total cost, capacity, distance and duration to any route-vehicle pair.

3.1.7 Combination

By the term “combination”, we refer to a certain route-vehicle pair and the set of orders assigned to it. By solving the LTL planning problem, we produce a set of activated combinations where each combination features a route, a vehicle and a set of orders.

3.1.8 Trip

In the context of the last-mile delivery problem, we use the term “trip” instead of “combination”. A trip is a single tour in which a certain sequence of customers are visited by a certain vehicle. Since each customer represents an order in the last-mile delivery problem, a trip in the last-mile delivery context is equivalent to a combination in the LTL context.

The next section illustrates the network structures of LTL, FTL and last-mile delivery operations in more detail.

3.2 Network Structure

The 3PL company employs a hybrid hub-and-spoke network to fulfill its customers' LTL and FTL transportation requests. The whole territory is divided into twenty zones, each of which accommodates a single hub. These hubs function as regional cross-docking facilities for consolidation and splitting purposes. Each end-of-line terminal, which can be a pickup, delivery and/or a vehicle home location, is served by the hub within its region.

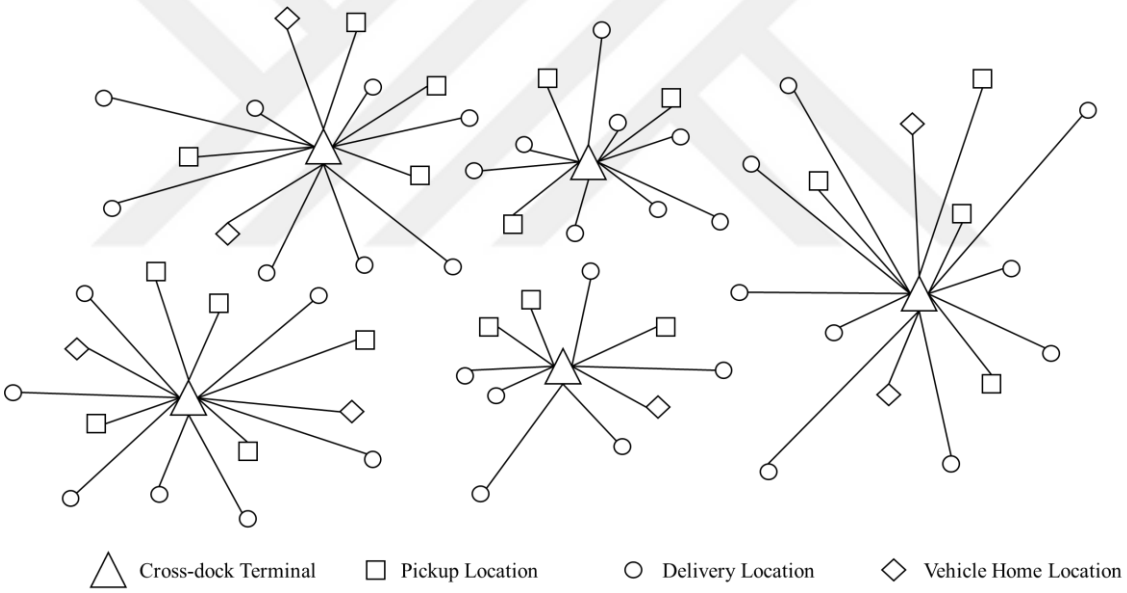


Figure 3.1. Sample Network Structure for LTL and FTL Operations.

While in a pure hub-and-spoke system all the flow is routed through hubs, the hybrid strategy allows direct shipments between origin and destination pairs as well. Any LTL order to be transported across the network can be picked up and delivered along a direct route or undergo one (or two) transfers before arriving at its final destination. This implies that an item can be carried by one, two or three vehicles from pick up until delivery. Transfers can take place at the cross-dock terminals located in the origin zone and/or the destination zone of any shipment.

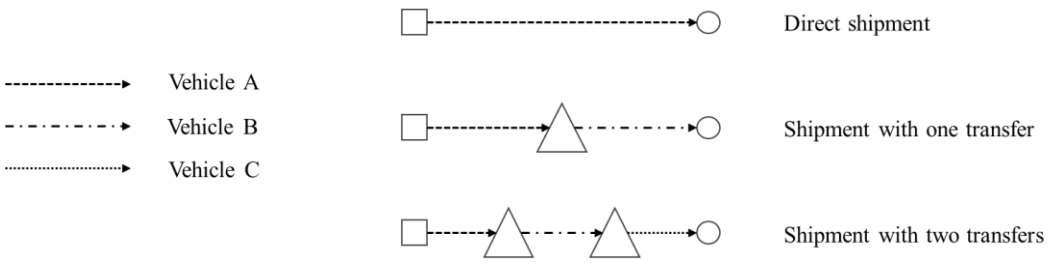


Figure 3.2. LTL Shipment Types.

The network for last-mile delivery operations consists of a central depot, which acts as a pickup location, a large number of delivery locations, which are also called as customers, and several vehicle home locations (see Figure 3.3).

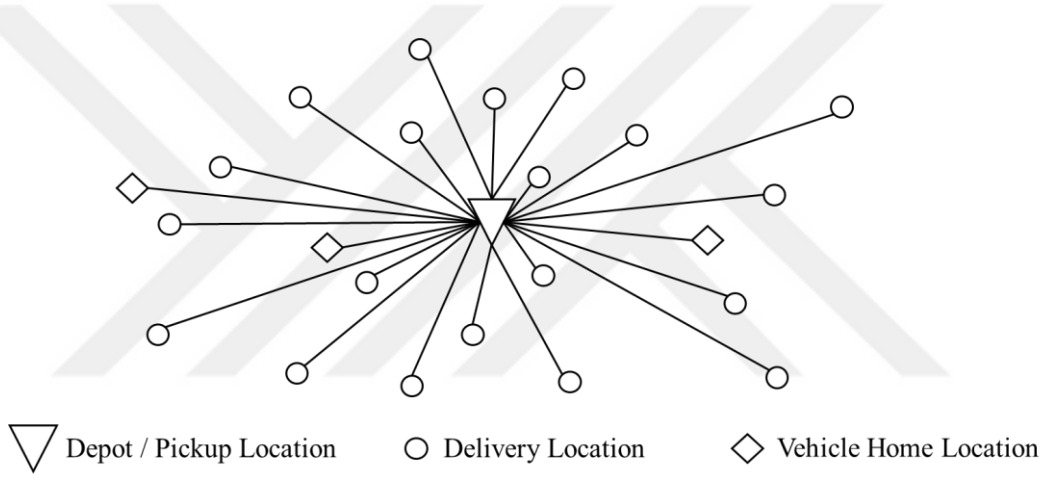


Figure 3.3. Sample Network Structure for Last-mile Delivery Operations.

In the last-mile delivery process, customers are directly served from the depot they are assigned to, without any transfers. Dedicated vehicles perform trips to deliver orders to customers, and they return to their home locations after completing their duties. The home locations may or may not be the same as the depot.

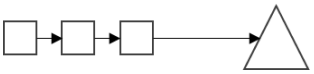
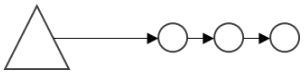
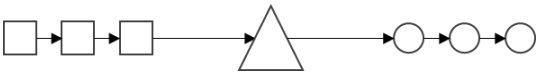
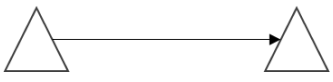
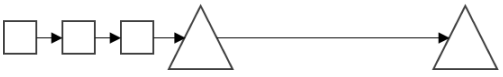
There are more than thirty different last-mile delivery networks operated by the 3PL company, and each of them is independent from each other in terms of the locations within the network and their vehicle fleets.

In general, networks of LTL/FTL operations and the last-mile delivery operations are distinct. However, it is commonly observed that a cross-dock terminal for the LTL operations is physically the same facility as a depot for the last-mile delivery operations. It is also possible that the depot of a certain last-mile delivery network becomes a delivery location for the LTL/FTL network.

3.3 Route Types

Since we focus on the road transport operations, all freight is moved by trucks in the problem we aim to tackle. Each truck travels along a certain route. A route can be represented by a sequence of locations to be visited by the truck. The types of routes that are typically used in the LTL operations of the 3PL company have various distinctive characteristics.

It is possible for a truck to stop at pickup locations, cross-dock terminals and delivery locations in a single LTL route. Nonetheless, pickup locations are always visited before cross-dock terminals and cross-dock terminals are always visited before delivery locations. There is an origin zone and a destination zone associated with each route. A route is classified as an *intra-zone route* if it has the same origin and destination zones, and an *inter-zone route*, otherwise. An important feature of the inter-zone routes is that the stops on the route are either in the origin zone or in the destination zone. This implies that a route cannot have stops in more than two zones. We can also categorize the LTL routes based on the type of locations they accommodate. Eight main types of LTL routes (see Figure 3.4) can be characterized in this manner.

Route Type	**Illustration**
1. Collection Route	
2. Distribution Route	
3. Pickup and Delivery Route	
4. Pickup and Delivery Route with Single Transfer	
5. Trunk Route	
6. Trunk Route with Pickup	

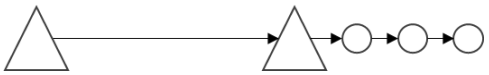
7. Trunk Route with Delivery	
8. Trunk Route with Pickup and Delivery	

Figure 3.4. Illustration of LTL Route Types.

Collection routes are used for picking up items and shipping them to a cross-dock terminal whereas distribution routes start from a cross-dock terminal and stop at delivery locations for dropping the items at their final destinations. Collection and distribution routes can be inter-zonal or intra-zonal depending on whether the visited cross-dock terminal is in the same region as the pickup locations (for collection routes) or delivery locations (for distribution routes). Pickup and delivery routes are used for direct shipments from origins to destinations. In an intra-zone pickup and delivery route all locations are in the same zone, whereas an inter-zone pickup and delivery route implies that the pickup locations are in the origin zone and the delivery locations are in the destination zone. Pickup and delivery routes with single transfer is the enhancement of the inter-zone pickup and delivery routes with an extra stop at a cross-dock terminal. The visited cross-dock terminal can either be in the origin zone or in the destination zone of the route. Trunk routes facilitate the movement of freight between cross-dock terminals. This type of routes can be augmented by accommodating additional stops for pickups, deliveries or both. Each type of enhancement establishes a separate route type. These route types are namely, trunk routes with pickup, trunk routes with delivery, and trunk routes with pickup and delivery. In the solution approaches we propose for addressing the LTL problem, we make use of the presented characterization of route types to generate feasible routes.

FTL routes are a special case of pickup and delivery routes with a single pickup and single delivery location. Last-mile delivery routes have a slightly different structure. Since vehicles can perform multiple trips during the day, a typical last-mile delivery routes consists of one or more tours that start and end at the depot location while the final trip terminates at the vehicle home location (see Figure 3.5).

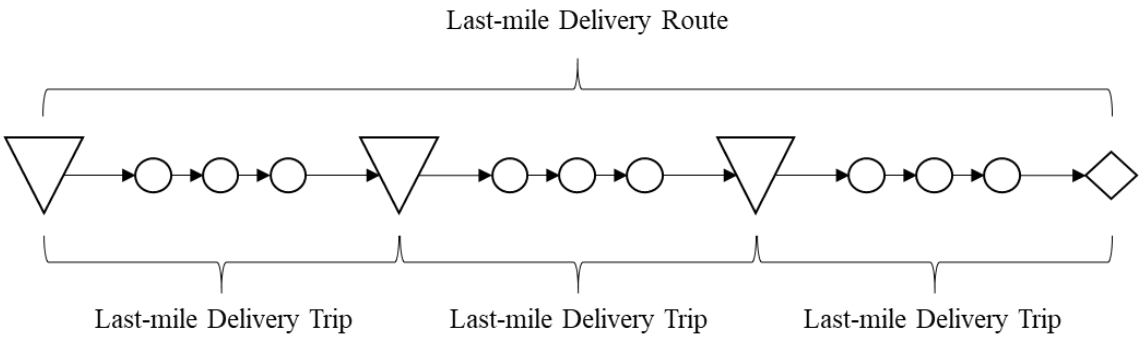


Figure 3.5. Illustration of a Sample Last-mile Delivery Route.

While LTL routes accommodate a small number of stops in general, last-mile delivery routes may contain up to thirty customer visits per day. Therefore the structure of routes in the LTL operations and last-mile delivery operations are quite different from each other.

The next section describes the current planning processes employed by the 3PL company.

3.4 Existing Planning Approach

The context of this study is directly related with the duties of four different operation units within the 3PL organization. These are namely, LTL management, FTL management, fleet management and last-mile delivery management divisions. These divisions are responsible for both managing their networks over the long term and sustaining the performance of day-to-day operations.

Planning of operations is performed in a highly decentralized manner by on-site dispatchers and planners. These employees are located at three types of facilities: depots, cross-dock terminals and fleet management offices. Depots also represent pickup locations in the LTL context and most of them are warehouses of the clients of the 3PL firm. Depending on the roles and facilities they are located at, we define the following planner profiles:

- LTL / FTL planners located at the depots
- LTL planners located at the cross-dock terminals

- Fleet planners located at fleet management offices
- Last-mile delivery planners located at the depots

The main responsibility of LTL / FTL planners located at the depots is planning the LTL and FTL routes that originate from their depots. LTL planners located at the cross-dock terminals exchange information with the LTL planners at the depots within their zones and plan inbound and outbound LTL routes to achieve a better consolidation through transfers. The planning of routes is done manually, and usually in a sequential manner. The planners located at both depots and cross-dock terminals communicate with the fleet management office for arrangement of trucks to perform the routes they have planned. Fleet planners usually work on a first-come first-served basis and they assign dedicated vehicles or spot vehicles to fulfill the requests using the most up-to-date information they have. Last-mile delivery operations are planned separately from the other operations by planners located at the depots that serve last-mile delivery customers.

The 3PL company continuously receives orders from its clients every day. Transportation plans are made either early in the morning or in the evening before the day trucks are dispatched. An LTL transportation plan is simply represented by the assignment of the orders to routes which are matched with a certain vehicle type. It is possible that an order is not assigned to any route in a daily transportation plan, in which case it is referred to as an *unfulfilled* order for that day.

Each planner is responsible for planning a certain subset of daily orders depending on the type and location of the facilities they are settled in. The LTL planners located at the depots plan the orders which originate from their facilities. The routes they plan have a single pickup location, which is the facility they are settled in. Nevertheless, there may be more than one delivery location and/or cross-dock terminals on the routes planned by them. These route types are illustrated in Figure 3.6.

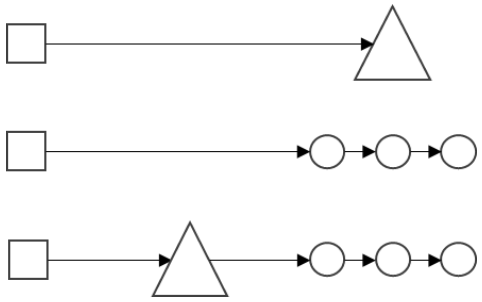


Figure 3.6. Route Types Planned by LTL Planners at the Depots.

One of the important factors that is taken into consideration by the LTL planners is the capacity utilization of the vehicles they use in the transportation plan. A route-vehicle pair is *admissible* only if the capacity usage resulting from the assigned orders is above a certain limit. This limit changes for each route type. The typical minimum capacity utilization is 80% for routes planned by the planners located at the depots. Any order that cannot be assigned to a route-vehicle pair due to this constraint is reported to the planner at the cross-dock terminal within the same zone as the depot. The planners located at the cross-dock terminals receive similar order information from all depots and other pickup locations within their zone. Based on this information, they plan the inbound shipments, which are transported through intra-zone collection routes. The planners at the cross-dock terminals are also responsible for planning outbound LTL shipments, which include the items that are already unloaded at their stations and waiting for their next transportation to their final destinations. Distribution routes, trunk routes and trunk routes with deliveries constitute the type of routes that are used in the daily plans of planners at the cross-dock terminals for processing outbound shipments (see Figure 3.7).

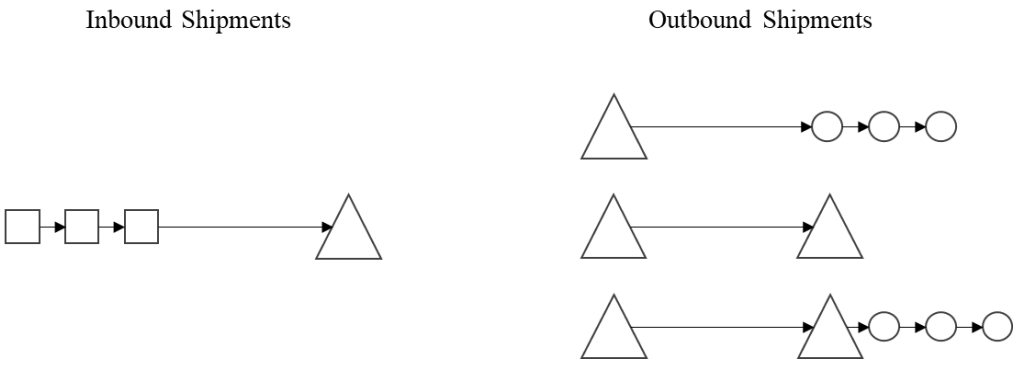


Figure 3.7. Route Types Planned by LTL Planners at the Cross-dock Terminals.

Once the planning is done, the information regarding the FTL orders, planned LTL routes and required vehicle types is sent to the corresponding fleet management offices. Fleet planners have the most recent information regarding the status of dedicated vehicles. They organize the assignment of dedicated vehicles to a certain portion of planned routes and arrange trucks from the spot market for the rest of the routes. The fleet planners try to achieve the least cost assignment by considering several factors such as the relative costs of performing a certain route with dedicated or spot vehicles, empty truck movements and returning dedicated vehicles to their home locations on time.

The last-mile delivery operations are carried out by their own dedicated vehicle fleet, which is composed of smaller trucks compared to those used in LTL and FTL operations. While the delivery locations in the LTL and FTL operations are often big facilities such as distribution centers or large supermarkets, last-mile delivery operations involve home deliveries and deliveries to small shops in urban areas. Therefore, last-mile delivery planners need to take into account several extra factors in the route planning process. Some of these factors are delivery time windows, accessibility restrictions in some areas for certain vehicle types, and home locations for vehicle returns.

Although transportation plans are made daily, due to the possibility of leaving an order unfulfilled on a certain day, the quality of the plans in terms of total freight costs should be evaluated in the long run. Optimizing the transportation plans for each day does not guarantee that the total cost is minimized in a multi-day setting. The main reason for this is the lack of information regarding future orders. In the daily LTL planning process, a planner has three options when deciding how to handle an order:

i) Do not assign it to any route,

ii) Assign it to a route by which it is shipped to its final destination directly,

iii) Assign it to a route by which it is shipped to a cross-dock terminal.

All three options have certain advantages and disadvantages and applying a good strategy has the potential to improve the overall freight cost in the long term. Let us assume that that an order is not fulfilled on a certain day due to lack of efficient consolidation opportunities. On one hand, it is possible that a better plan is achieved the next day with newly arriving orders. But it is also possible that no efficient combination

will be available and the order may need to be shipped by a truck with a low capacity utilization in order not to cause too many late deliveries.

There is also a trade-off between shipping an order directly to its final destination and shipping it via cross-dock terminals with transfer operations. The first option may require extension of a direct delivery route by an extra delivery stop, which increases the freight cost associated with that route but it also helps avoiding an extra transportation step and may be advantageous when no efficient consolidation opportunities arise in the subsequent plans. Therefore, direct shipping versus cross-docking decision is also a crucial point for the LTL planning process.

In this section we summarized the existing way of planning transportation operations. In the next section we present an overview of the new approach we propose.

3.5 Overall Solution Design

In this section we describe a general planning approach for the road transportation operations of the 3PL company by identifying the problems to be addressed and proposing a solution strategy to integrate the techniques that will be designed to solve each of these problems independently.

The first problem we identify is named as the *LTL Planning Problem* (LTLP). As opposed to the current LTL planning approach where routes are planned in a decentralized way, LTLP is defined as a centralized problem where all LTL orders are planned together. The main inputs of LTLP are LTL orders, while the specifications of available route types and vehicle types are also taken into consideration. LTLP produces combinations and a set of unfulfilled orders. Each combination is a route-vehicle pair with a set of orders assigned to it. The routes in the LTLP solutions comply with the rules that are currently taken into consideration by LTL planners. As a part of our general solution strategy, LTLP incorporates only spot vehicles, which implies dedicated vehicles are not used in LTLP solutions. It is assumed that an unlimited number of spot vehicles of each type is available in the system.

The second problem we address is called the *Dedicated Vehicle Assignment Problem* (DVAP). DVAP aims to assign dedicated vehicles to FTL orders and planned LTL routes. In this sense, DVAP takes the output of LTLP as an input. LTL outputs may include route

dependencies since an order may be assigned to more than one route when it undergoes a transfer. This creates synchronization issues that need to be dealt with. DVAP may reschedule routes in order to maintain the synchronization. DVAP may assign multiple routes to a dedicated vehicle taking into account empty vehicle movements and return of vehicles to their home locations.

The last problem we aim to solve is the *Last-mile Delivery Planning Problem* (LMDP). LMDP resembles the daily route planning problem encountered by last-mile delivery planners by incorporating all practical rules and constraints such as multiple trips, time windows and accessibility restrictions.

As we will present in the following chapters of this thesis, automated solution approaches have been developed for each of these problems. Let us refer to the collection of models and algorithms that are used for solving each problem independently as *solver*s. Having the LTLP, DVAP and LMDP solvers, the automated planning system illustrated in Figure 3.8 can be implemented.

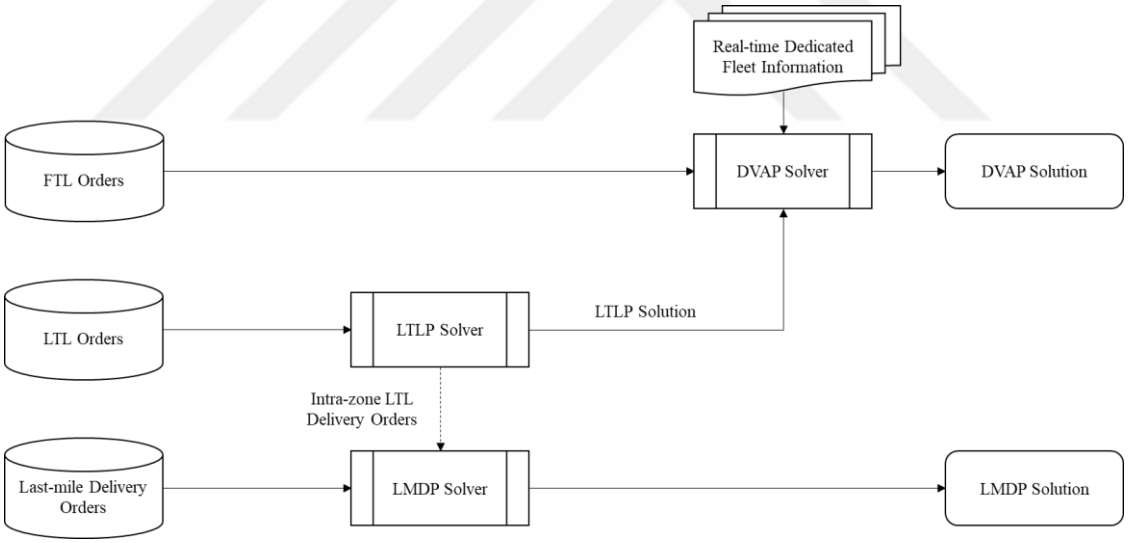


Figure 3.8. Automated Road Transportation Planning System Proposal.

The presented system proposes an integrated solution strategy for the planning of road transportation operations of the 3PL carrier. In the suggested process flow, firstly the LTLP is solved. LTLP and LMDP solvers can be integrated in two different ways. In the specific case where the cross-dock terminal and last-mile delivery depot are in the same facilities, it may be possible to combine the intra-zone LTL orders that need to be shipped from the cross-dock terminal to their final destinations with the other last-mile delivery

orders within the same zone. The combined order set can be planned through the LMDP solver. Another way to integrate these methods would be embedding the LMDP solver into the LTLP solver to solve the intra-zone collection and distribution sub-problems of LTLP via the LMDP solver. This can be a useful strategy if the intra-zone collection and distribution routes used in the LTL operations display similar characteristics as the last-mile delivery routes. In our specific case these route types are highly differentiated and this integration is not adopted. Currently, the LTL and last-mile delivery operations of the 3PL company are managed through different information systems. In order to overcome this issue, the two systems should be integrated in the future so that they can exchange order information automatically.

The solution of LTLP constitutes the input of DVAP along with the FTL orders and real-time location and status of dedicated vehicles. Therefore, DVAP solver can be executed right after the LTLP solution is obtained. LMDP solver can be run in parallel to the DVAP solver, since the last-mile delivery operations are performed by their own dedicated fleets and the two problems are not directly linked with each other.

The combination of DVAP and LMDP solutions become the output of the automated road transportation planning system.

The system should be structured in a modular way such that the individual solvers can also be executed independently when necessary. The ideal usage of the system would be running it on a daily basis, preferably during the night before plans are realized. However, it should also be possible to execute the system at any time, using the most recent order and vehicle information.

In this section we first identified the three main problems that we will address in the rest of the thesis. Then we proposed a system and an automated solution strategy for the overall road transportation planning problem of the 3PL firm. In the next three chapters, we will focus on the three problems we identified, respectively. For each problem, we will present a formal definition, our solution approach and computational results.

Chapter 4

THE LESS-THAN TRUCKLOAD PLANNING PROBLEM

This chapter addresses two different versions of the LTLP. The two versions are called the *LTLP with Continuous Planning Horizon* (LTLP-C) and the *LTLP with Discrete Planning Horizon and Multi-period Considerations* (LTLP-DM), respectively.

We first examine the LTLP-C, which aims to find the best combination of order-route-vehicle assignments over a single, continuous and open-ended planning horizon. The LTLP-C plans the complete set of routes to ship orders from their origins to their destinations. Therefore, orders that will be shipped through transfers will be assigned to multiple routes in a single transportation plan, which causes synchronization requirements at the cross-dock terminals. It is possible that an order is not fulfilled in a certain plan, in which case an artificial unfulfillment cost is incurred. We formulate the LTLP-C as an MILP problem, which can be solved for small instances. Larger instances are solved via a two-phase matheuristic approach.

After analyzing the LTLP-C solutions with the relevant parties at the 3PL company, we decided to slightly change the scope of the problem in order to make it a better fit to the real-world LTL planning problem. The new problem we address is the LTLP-DM, which is a simplified version of the LTLP-C in several aspects. We consider a discrete planning horizon in a multi-period setting where each period corresponds to a single day. We solve a daily planning problem each day by taking into account the implications of our decisions in a multi-day setting. The quality of the LTLP-DM solutions are therefore evaluated through long-term freight costs. Instead of planning the whole shipment process of an order from origin to destination, we make assignment decisions on a daily basis. This implies that an order can be assigned to at most one route in a certain transportation plan. If the order is not shipped to its final destination directly but it is transported to a cross-dock terminal to be transferred to another route, the subsequent shipment of the order will be included in the future LTLP-DM instances. This approach creates two main

advantages: Firstly, it increases the possibility of achieving a better consolidation for the orders-to-be-transferred since they are re-planned with newly received orders in the subsequent days. Secondly, the synchronization requirement can be lifted which reduces the complexity of the problem. In this way, realistic instances of the problem can be solved in a timely manner. We propose two matheuristic approaches for solving the LTLP-DM. The first one, called the Decentralized Heuristic (DH), mimics the existing planning approach of the 3PL company, which serves as a benchmark for comparison. The second method, which is named as the Centralized Heuristic (CH) adds a certain degree of centralization to the solution approach and produces better results in the long run.

Solution approaches for both LTLP-C and LTLP-DM are decomposition-based methods and they both involve solving a series of sub-problems through mathematical programming solvers. All the models designed for addressing the sub-problems require that the set of admissible routes are generated beforehand and available to use. Since LTL routes have certain characteristics and constraints, it is computationally easy to produce these routes for each of the sub-problems within the solution algorithms.

In the first section of this chapter we present our route generation algorithm and a procedure for constructing the set of route-vehicle pairs, which are both utilized by LTLP-C and LTLP-DM solvers. The second section focuses on the LTLP-C and the final section provides details of the LTLP-DM. In the second and the third sections, we give formal definitions of the respective problems along with their mathematical programming formulations. We then present solution approaches designed to tackle the problems and demonstrate the results of computational experiments.

4.1 Route Generation Algorithm

The LTL routes that are currently employed by the 3PL company have various distinctive characteristics. We have identified eight main LTL route types based on the common route structures. However, a more detailed classification is needed to explicitly define route types for the route construction process. The following attributes need to be specified for each route type:

$\beta^{minpick}$: Minimum number of pickup locations in the origin zone

$\beta^{maxpick}$: Maximum number of pickup locations in the origin zone

β^{mindel} : Minimum number of delivery locations in the destination zone

β^{maxdel} : Maximum number of delivery locations in the destination zone

$\beta^{xdorigin}$: True, if the route visits the cross-dock terminal in the origin zone; false, otherwise

β^{xddest} : True, if the route visits the cross-dock terminal in the destination zone; false, otherwise

$\beta^{maxspdev}$: The maximum percentage deviation allowed from the shortest path distance between the first and last locations of the given route

$\beta^{minutil}$: Minimum capacity utilization allowed for any vehicle to take the given route

The set of orders (I), the set of intra-zonal route types (H^{intra}) and the set of inter-zonal route types (H^{inter}) to be used in the route generation process are determined based on the corresponding problem instance to be solved. Based on the provided orders in I, set of all possible origin-destination pairs (W) is generated. Each order $i \in I$ has an origin location, l_i^O and a destination location, l_i^D. All possible distinct pairs of origin and destination locations (l^O, l^D) form the set W.

Each of the route types provided, $h \in (H^{intra} \cup H^{inter})$ is associated with a vector $\beta_h = (\beta_h^{minpick}, \beta_h^{maxpick}, \beta_h^{mindel}, \beta_h^{maxdel}, \beta_h^{xdorigin}, \beta_h^{xddest}, \beta_h^{maxspdev}, \beta_h^{minutil})$. Any route r is a sequence of locations visited. $\bar{L}_r^P, \bar{L}_r^D$ and $\bar{L}_r^C$ are ordered lists of visited pickup locations, delivery locations and cross-dock terminals along route r, respectively. Eventually, we define $r = (\bar{L}_r^P, \bar{L}_r^C, \bar{L}_r^D)$. Throughout the route generation algorithm, we maintain a list of routes-to-be-explored (R^E). Generated routes which are admissible are accumulated in the set R_I, given that the associated order set is I. Total distance of route r is denoted by D_r^T, whereas D_r^S stands for the shortest path distance between the first and the last locations of route r. Any admissible route r should satisfy $D_r^T \leq D_r^S * (1 + \beta^{maxspdev})$. Furthermore, S_r^{pot} represents the potential of route r in terms of size, which is the summation of the sizes of all orders that can be carried along route r. Any route r of type h can be admissible if $S_r^{pot} \geq \beta_h^{minutil} * Q^{min}$, where Q^{min} is the

capacity of the smallest vehicle for the given problem. The pseudocode of the route generation algorithm is provided in Algorithm 4.1.

Algorithm 4.1. Route Generation

1: *Get algorithm inputs: I, H^{intra} and H^{inter}. Let $W := \emptyset$ and $R_I := \emptyset$*
2: *Create the set of origin destination pairs (W) based on the given set of orders (I)*
3: *For each element $(l^O, l^D) \in W$:*
4: *If l^O and l^D are in the same zone:*
$H^* := H^{intra}$
Else:
$H^* := H^{inter}$
5: *For each route type $h \in H^*$:*
6: *Let $R^E := \emptyset$*
Let $\bar{L}^P_{r^*} := \emptyset$, $\bar{L}^C_{r^*} := \emptyset$, $\bar{L}^D_{r^*} := \emptyset$ and define empty route $r^* = (\bar{L}^P_{r^*}, \bar{L}^C_{r^*}, \bar{L}^D_{r^*})$
7: *If $\beta_h^{minpick} > 0$ and $l^O \in L^P$:*
Append l^O to $\bar{L}^P_{r^*}$
8: *If $\beta_h^{xdorigin} = True$:*
Let z_1 be the origin zone of r^*. Append $c_1 \in L^C_{z_1}$ to $\bar{L}^C_{r^*}$
9: *If $\beta_h^{xddest} = True$ and l^O and l^D are in different zones;*
Let z_2 be the destination zone of r^*. Append $c_2 \in L^C_{z_2}$ to $\bar{L}^C_{r^*}$
10: *If $\beta_h^{mindel} > 0$ and $l^D \in L^D$:*
Append l^D to $\bar{L}^D_{r^*}$
11: *Let $D^S_{r^*} := D^T_{r^*}$*
12: *If $S^{pot}_{r^*} \geq \beta_h^{minutil} * Q^{min}$, $|\bar{L}^P_{r^*}| \geq \beta_h^{minpick}$ and $|\bar{L}^D_{r^*}| \geq \beta_h^{mindel}$:*
Append r^* to R_I
13: *Append r^* to R^E*
14: *While $|R^E| > 0$:*
15: *Pick and remove the first route (r^{**}) from R^E*
16: *If $|\bar{L}^P_{r^{**}}| < \beta_h^{maxpick}$:*
17: *Let z_1 be the origin zone of r^{**}. For each pickup location $l_1 \in L^P_{z_1}$ such that $l_1 \notin \bar{L}^P_{r^{**}}$*
18: *For position i from 2 to $|\bar{L}^P_{r^{**}}| + 1$:*
19: *Insert l_1 into i^{th} position in $\bar{L}^P_{r^{**}}$ and obtain $\bar{L}^P_{r'}$, Let $\bar{L}^C_{r'} := \bar{L}^C_{r^{**}}$ and $\bar{L}^D_{r'} := \bar{L}^D_{r^{**}}$*
20: *Let $r' = (\bar{L}^P_{r'}, \bar{L}^C_{r'}, \bar{L}^D_{r'})$ and append r' to R^E*

21: *If* $S_{r'}^{pot} \geq \beta_h^{minutil} * Q^{min}$, $\left|\bar{L}_{r'}^{P}\right| \geq \beta_h^{minpick}$, $\left|\bar{L}_{r'}^{D}\right| \geq \beta_h^{mindel}$ *and* $D_{r'}^{T} \leq D_{r^*}^{S} * (1 + \beta^{maxspdev})$:

22: *Append* r^* *to* R_I

23: *If* $\left|\bar{L}_{r^{**}}^{D}\right| < \beta_h^{maxdel}$:

24: *Let* z_2 *be the destination zone of* r^{**}. *For each delivery location* $l_2 \in L_{z_2}^{D}$ *such that* $l_2 \notin \bar{L}_{r^{**}}^{D}$

25: *For position i from* 1 *to* $\left|\bar{L}_{r^{**}}^{D}\right|$:

26: *Insert* l_2 *into* i^{th} *position in* $\bar{L}_{r^{**}}^{D}$ *and obtain* $\bar{L}_{r'}^{D}$. *Let* $\bar{L}_{r'}^{P} := \bar{L}_{r^{**}}^{P}$ *and* $\bar{L}_{r'}^{C} := \bar{L}_{r^{**}}^{C}$

27: *Let* $r' = \left(\bar{L}_{r'}^{P}, \bar{L}_{r'}^{C}, \bar{L}_{r'}^{D}\right)$ *and append* r' *to* R^E

28: *If* $S_{r'}^{pot} \geq \beta_h^{minutil} * Q^{min}$, $\left|\bar{L}_{r'}^{P}\right| \geq \beta_h^{minpick}$, $\left|\bar{L}_{r'}^{D}\right| \geq \beta_h^{mindel}$ *and* $D_{r'}^{T} \leq D_{r^*}^{S} * (1 + \beta^{maxspdev})$:

29: *Append* r^* *to* R_I

30: *Remove duplicate routes in* R_I *having the same set of pickup locations, cross-dock locations and delivery locations but not necessarily in the same order. Ensure that only the routes with the minimum distance remain.*

31: *Return* R_I

Inputs of the route generation process is the set of orders (I), the set of intra-zone route types (H^{intra}) and the set of inter-zone route types (H^{inter}). The algorithm starts by identifying all origin-destination pairs (l^O, l^D) in the network for the corresponding problem based on the pickup and delivery locations of the transportation orders in set I (Step 2). For each origin-destination pair (l^O, l^D) and admissible route type h, we first construct a base route which includes the selected origin location l^O, destination location l^D and associated cross-dock terminals, depending on the values of $\beta_h^{xdorigin}$ and β_h^{xddest}, through steps 6 – 11 of the algorithm. The base route is added to the list of routes-to-be-explored (R^E) and if it is admissible with respect to the parameters specified by the β_h vector, it is also added to the set of generated routes (R_I). Next, we apply a breadth-first search by adding new pickup and delivery locations to each route in R^E one by one at steps 14 through 29 of Algorithm 4.1. Each newly created route is tested for admissibility and added to the R_I if the required conditions are met. Note that the shortest path distance associated with any route r' ($D_{r'}^{S}$) equals the total distance of its base route r^* ($D_{r^*}^{T}$). At step 30, generated routes that have exactly the same set of pickup, cross-dock and delivery locations are reduced by removing the duplicates such that only the routes

with the minimum total distance stay at R_I. The algorithm returns the set of generated routes R_I.

Our initial assumption is that an unlimited number of trucks of each type is available in the spot market. For each problem instance to be solved, we need to construct a set of route-vehicle pairs (P) by combining the routes in R_I with sufficient number of trucks of each type such that no order has to be unassigned, or assigned to an undesirable route-vehicle pair. Let V be the set of vehicle types for the given problem where Q_v^V denotes the capacity of the truck associated with the vehicle type $v \in V$. For each route $r \in R_I$ of type $h \in \left(H^{intra} \cup H^{inter}\right)$ and vehicle type $v \in V$, the number of route-vehicle pairs needed is calculated by the following formula:

$$min\left\{\left\lceil \frac{S_r^{pot}}{Q_v^V \beta_h^{minutil}} \right\rceil, |I_r^R|\right\} \qquad (4.1)$$

Let I_r^R be the subset of orders which can possibly be assigned to route r. S_r^{pot} is the sum of the sizes of orders in I_r^R, which represents the total load that can be potentially carried along route r. Each route is also associated with a minimum utilization limit $\beta_h^{minutil}$, which is specified for its route type. The presented formula (4.1) returns the minimum of two integer values. The first value stands for the number of vehicles of type v that would be needed if all orders in I_r^R are carried along the same route (r) only with vehicles of type v, which are loaded at their minimum capacity utilization limits. The second value is simply the number of orders in I_r^R. Since an order can be assigned to at most one route-vehicle pair, $|I_r^R|$ sets an upper limit on the number of vehicles needed to cover all orders in I_r^R. The minimum of the two numbers represents the maximum number of vehicles of type v that can potentially be used in combination with route r. Therefore, the set of route-vehicle pairs (P) is constructed by duplicating each route and vehicle type pair by the number found by formula (4.1).

4.2 The Less-than Truckload Planning Problem with Continuous Planning Horizon

The LTLP-C involves pickup and delivery orders with deadlines and cross-docking opportunities causing synchronization requirements. The routes have a multi-echelon

structure the type of routes employed adhere to the rules that are taken into consideration by the existing LTL planners of the 3PL company. The vehicle fleet consists of heterogeneous spot-hired vehicles. Each truck is linked with a certain vehicle type. Different vehicle types have different vehicle capacities and shipping costs. The total freight cost associated with a trip depends on the route to be traveled and the type of vehicle to be used. Although spot prices may vary from day to day, we assume that the cost of a route-vehicle pair is a deterministic parameter which can be calculated based on the route characteristics such as distance, duration origin-destination cities and number of stops, and vehicle characteristics such as fixed vehicle costs and fuel consumption per kilometer. The firm has a broad network of truck drivers and can find any type of vehicle, at any quantity within a short notice period. Therefore, it is assumed that the number of trucks that can be used is unlimited for all vehicle types.

The objective of the problem is to find a minimum cost assignment of orders to feasible route-vehicle pairs while maintaining synchronization and ensuring delivery deadlines are not exceeded. The objective function also incorporates an order unfulfillment cost, which is incurred if an order is not fulfilled in the transportation plan. We formulate this problem as an MILP model. We propose a decomposition-based iterative matheuristic algorithm that consists of construction and improvement steps for solving large instances. In the computational study, we compare the results of the MILP solutions against matheuristic results.

4.2.1 Problem Definition

The main inputs of the base problem are the sets of orders (I), pre-determined route-vehicle pairs (P) and locations (L). L^P, L^D and L^C denote the set of pickup locations, delivery locations and cross-dock terminals such that ($L = L^P \cup L^D \cup L^C$). Each order ($i \in I$) has an origin location ($l_i^O \in L^P$) and a destination location ($l_i^D \in L^D$). Size of order i is denoted by S_i. An order can be assigned to one or more route-vehicle pairs. In the latter case, it is transferred from one vehicle to another at least once. Transfer operations take place at cross-docking locations. It is possible that an order is not assigned to any route-vehicle pair. In that case, the order is unfulfilled and an unfulfillment cost of C_i^U is

incurred. There is a deadline associated with each order delivery. The latest delivery time for order i is denoted by D_i.

We form the set of feasible routes by the route generation algorithm. The set of route-vehicle pairs (P) is constructed through the procedure defined in Section 4.1. Each route-vehicle pair ($p \in P$) possesses attributes of both the route and the vehicle attached to it. C_p^F denotes the total freight cost of route-vehicle pair p. Q_p stands for the capacity of the vehicle used in route-vehicle pair p. Each route embodies a series of locations to be visited. The position of location $l \in L$ within the route associated with route-vehicle pair p is denoted by P_{lp}^L. For each route-vehicle pair, the time between the starting of the route and the arrival at a specific location within the route is given as a constant parameter, which is represented by T_{lp}^A for location l within the route linked with route-vehicle pair p. T_{lp}^A includes the driving time, which depends on the distance traveled up to the location l and the average speed of the vehicle, in addition to loading and unloading times spent prior to arriving at the location l along the route associated with p. All these components are assumed to have constant values. The unloading time at location l is denoted by T_l^U.

We define the following binary parameters to be used in the MILP formulation of the base problem:

$$O_{ip}^P : \begin{cases} 1, if\ order\ i\ can\ be\ picked\ up\ by\ route-vehicle\ pair\ p \\ 0, otherwise \end{cases} \quad i \in I, p \in P$$

$$O_{ip}^D : \begin{cases} 1, if\ order\ i\ can\ be\ delivered\ by\ route-vehicle\ pair\ p \\ 0, otherwise \end{cases} \quad i \in I, p \in P$$

$$R_{lp}^L : \begin{cases} 1, if\ location\ l\ is\ within\ the\ route\ linked\ with\ p \\ 0, otherwise \end{cases} \quad l \in L, p \in P$$

$$R_{p_1 p_2 l} : \begin{cases} 1, if\ routes\ of\ r_1\ and\ r_2\ intersect\ at\ cross-dock\ location\ l \\ 0, otherwise \end{cases} \quad \begin{matrix} p_1, p_2 \in P, \\ p_1 \neq p_2, \\ l \in L^C \end{matrix}$$

We define seven types of decision variables to formulate the base problem. x_{ip}^P is a binary variable which indicates whether order i is picked up by route-vehicle pair p. Similarly, x_{ip}^D takes the value of one if order i is delivered by route-vehicle pair p and zero, otherwise. Variables x_{ip}^P and x_{ip}^D are included in the MILP formulation for order $i \in I$ and route-vehicle pair $p \in P$, if $O_{ip}^P = 1$ and $O_{ip}^D = 1$, respectively. The binary variable associated with the transfer of an order from one route-vehicle pair to another is

represented by $x^T_{ip_1p_2l}$. This variable becomes equal to one if and only if order $i \in I$ is transferred from route-vehicle pair $p_1 \in P$ to route-vehicle pair $p_2 \in p$ at cross-docking location $l \in L^C$ where $p_1 \neq p_2$ and $R_{p_1p_2l} = 1$. We set the value of the binary variable u_i to one in the case where order $i \in I$ is not assigned to any route-vehicle pair and to zero, otherwise. Variable y_p shows whether route-vehicle pair $p \in P$ is activated, that is, at least one order is assigned to p. The starting time of the route linked with route-vehicle pair p is determined by the continuous variable a_p. Finally, $b_{p_1p_2l}$ takes the value of one, if the vehicle attached to route-vehicle pair $p_1 \in P$ arrives at the cross-docking location $l \in L^C$ before the vehicle associated with $p_2 \in P$ where $p_1 \neq p_2$ and $R_{p_1p_2l} = 1$. We assume that T^P is a sufficiently large number that represents the maximum length of the planning horizon. The resulting MILP formulation of the base problem is given as follows:

$$Minimize \quad \sum_{p \in P} C^F_p y_p + \sum_{i \in I} C^U_i u_i \tag{4.2}$$

subject to

$$\sum_{\substack{r \in R \\ O^P_{ip}=1}} x^P_{ip} + u_i = 1 \qquad i \in I \tag{4.3}$$

$$\sum_{\substack{r \in R \\ O^D_{ip}=1}} x^D_{ip} + u_i = 1 \qquad i \in I \tag{4.4}$$

$$x^P_{ip_1} \leq O^D_{ip_1} x^D_{ip_1} + \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l \in L^C \\ R_{p_1p_2l}=1}} x^T_{ip_1p_2l} \qquad i \in I, p_1 \in P, \; O^P_{ip_1} = 1 \tag{4.5}$$

$$x^P_{ip_1} + \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l \in L^C \\ R_{p_2p_1l}=1}} x^T_{ip_2p_1l} \leq 1 \qquad i \in I, p_1 \in P, \; O^P_{ip_1} = 1 \tag{4.6}$$

$$x^D_{ip_1} + \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l \in L^C \\ R_{p_1 p_2 l}=1}} x^T_{ip_1 p_2 l} \leq 1 \qquad i \in I, p_1 \in P, \; O^D_{ip_1} = 1 \qquad (4.7)$$

$$\sum_{\substack{p_2 \in P \\ R_{p_1 p_2 l_1}=1}} x^T_{ip_1 p_2 l_1} \leq O^P_{ip_1} x^P_{ip_1} + \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l_2 \in L^C \\ P^L_{l_2 p_1} < P^L_{l_1 p_1} \\ R_{p_2 p_1 l_2}=1}} x^T_{ip_2 p_1 l_2} \qquad i \in I, p_1 \in P, l_1 \in L^C \qquad (4.8)$$

$$\sum_{\substack{p_2 \in P \\ R_{p_2 p_1 l_2}=1}} x^T_{ip_2 p_1 l_1} \leq O^D_{ip_1} x^D_{ip_1} + \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l_2 \in L^C \\ P^L_{l_1 p_1} < P^L_{l_2 p_1} \\ R_{p_1 p_2 l_2}=1}} x^T_{ip_1 p_2 l_2} \qquad i \in I, p_1 \in P, l_1 \in L^C \qquad (4.9)$$

$$x^P_{ip} \leq y_p \qquad i \in I, p \in P, O^P_{ip} = 1 \qquad (4.10)$$

$$x^D_{ip} \leq y_p \qquad i \in I, r \in R, O^D_{ip} = 1 \qquad (4.11)$$

$$x^T_{ip_1 p_2 l} \leq y_{p_1} \qquad i \in I, l \in L^C, p_1, p_2 \in P, \; p_1 \neq p_2, R_{p_1 p_2 l} = 1 \qquad (4.12)$$

$$x^T_{ip_1 p_2 l} \leq y_{p_2} \qquad i \in I, l \in L^C, p_1, p_2 \in P, \; p_1 \neq p_2, R_{p_1 p_2 l} = 1 \qquad (4.13)$$

$$\begin{aligned} \sum_{\substack{i \in I \\ O^P_{ip_1}=1 \\ P^L_{l_i^O p_1} < P^L_{l_1 p_1}}} S_i x^P_{ip_1} + \sum_{i \in I} \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l_2 \in L^C \\ P^L_{l_2 p_1} < P^L_{l_1 p_1} \\ R_{p_2 p_1 l_2}=1}} S_i x^T_{ip_2 p_1 l_2} \\ - \sum_{\substack{i \in I \\ O^D_{ip_1}=1 \\ P^L_{l_i^D p_1} \leq P^L_{l_1 p_1}}} S_i x^D_{ip_1} \\ - \sum_{i \in I} \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l_2 \in L^C \\ P^L_{l_2 p_1} < P^L_{l_1 p_1} \\ R_{p_1 p_2 l_2}=1}} S_i x^T_{ip_1 p_2 l_2} \leq Q_{p_1} \end{aligned} \qquad \begin{aligned} & \\ & \\ & l_1 \in L, p_1 \in P, \\ & R^L_{l_1 p_1} = 1 \\ & \end{aligned} \qquad (4.14)$$

$$a_p + T^A_{l_i^D p} \leq D_i + T^P\left(1 - x^D_{ip}\right) \qquad i \in I, p \in P, O^D_{ip} = 1 \qquad (4.15)$$

$$\left(a_{p_2} + T^A_{lp_2}\right) - \left(a_{p_1} + T^A_{lp_1} + T^U_l\right) \geq M\left(b_{p_1 p_2 l} - 1\right) \qquad i \in I, l \in L^C, p_1, p_2 \in P, \qquad (4.16)$$

$$p_1 \neq p_2, R_{p_1p_2l} = 1$$

$$x^T_{ip_1p_2l} \leq b_{p_1p_2l} \qquad i \in I, l \in L^C, p_1, p_2 \in P, p_1 \neq p_2, R_{p_1p_2l} = 1 \qquad (4.17)$$

$$\sum_{\substack{p_2 \in R \\ p_2 \neq p_1}} \sum_{\substack{l \in L^C \\ R_{p_1p_2l}=1}} x^T_{ip_1p_2l} \leq 1 \qquad i \in I, p_1 \in P \qquad (4.18)$$

$$x^P_{ip} \in \{0,1\} \qquad i \in I, p \in P, O^P_{ip} = 1 \qquad (4.19)$$

$$x^T_{ip_1p_2l} \in \{0,1\} \qquad i \in I, l \in L^C, p_1, p_2 \in P, p_1 \neq p_2, R_{p_1p_2l} = 1 \qquad (4.20)$$

$$x^D_{ip} \in \{0,1\} \qquad i \in I, r \in R, O^D_{ip} = 1 \qquad (4.21)$$

$$u_i \in \{0,1\} \qquad i \in I \qquad (4.22)$$

$$y_p \in \{0,1\} \qquad p \in P \qquad (4.23)$$

$$b_{p_1p_2l} \in \{0,1\} \qquad l \in L^C, p_1, p_2 \in P, p_1 \neq p_2, R_{p_1p_2l} = 1 \qquad (4.24)$$

$$a_p \geq 0 \qquad p \in P \qquad (4.25)$$

The objective of the model (4.2) is to minimize the cost of activated route-vehicle pairs and unfulfilled orders. We assume that all freight costs regarding a route-vehicle assignment are included in the cost parameter C^F_p. The cost of not fulfilling an order is a special cost parameter that should be set carefully. If the size of an order is large or its deadline is approaching, a higher unfulfillment cost should be set for it. Setting this cost too high would lead to usage of inefficient route-vehicle pairs, whereas setting it too low would result in missing out on the opportunities to attach the order to a proper route-vehicle pair.

Constraints (4.3) and (4.4) of the MILP formulation indicate that an order is either picked up and delivered by some route-vehicle pair or it is unfulfilled. Constraint (4.5) suggests that if an order is picked up by a certain route-vehicle pair, then it is either delivered by the same route-vehicle pair or it is transferred to another one at a cross-

docking location. Constraint (4.6) strengthens the formulation by ensuring that an order cannot be picked up by and transferred into the same route-vehicle pair. Similarly, constraint (4.7) guarantees that an order cannot be delivered by and transferred from the same route-vehicle pair. Constraint (4.8) implies that an order can be transferred from one route-vehicle pair to another if it is either picked up by its current route-vehicle pair or it is transferred into its current route-vehicle from another one earlier. A similar condition is imposed by constraint (4.9), which indicates if an order is transferred in to a route-vehicle pair, then it must be either delivered by its new route-vehicle pair or transferred to another route-vehicle pair later on. The binary variables related to pickups (x_{ip}^{P}), deliveries (x_{ip}^{D}) and transfers ($x_{ip_1p_2l}^{T}$) are allowed to take the value of one only if their associated route-vehicle pairs are activated. This requirement is imposed by constraints (4.10) through (4.13). Constraint (4.14) enforces that the vehicle capacity is not exceeded at any location along the route of a route-vehicle pair. We ensure that the difference between the size of total incoming orders and total outgoing orders up to each location within a route is less than or equal to the capacity of the vehicle attached to it. Constraint (4.15) guarantee that delivery deadlines are not exceeded. Constraints (4.16) and (4.17) maintain the synchronization of route-vehicle pairs by making sure that a transfer from a route-vehicle pair to another can take place only if the delivering vehicle arrives and finishes unloading at the corresponding cross-docking location before the receiving vehicle arrives. Constraint (4.18) satisfies that an order can be transferred into a certain route-vehicle pair at most once. Constraints (4.19) through (4.24) and (4.25) ensure binarity and non-negativity of variables, respectively.

In this formulation, if the number of route-vehicle pairs linked with the same route and a vehicle of the same type is high, then many feasible solutions will be symmetric. This redundancy will most likely adversely affect the performance of the solver. In order to overcome this handicap, we add symmetry breaking constraints inspired by the ones proposed by Coelho and Laporte [163]. Assume that we order the route-vehicle pairs from 1 to $|P|$ and the position of each route-vehicle pair represents its index. For any two route-vehicle pairs with the same route and a vehicle of the same type, the one having the larger index cannot be used unless the one with the smaller index is used. Let the index of route-vehicle pair p be denoted by $\delta(p)$ and let us define a new binary parameter $R_{p_1p_2}^{R}$ such that:

$$R^R_{p_1p_2} = \begin{cases} 1, if\ p_1\ and\ p_2\ have\ the\ same\ routes\ and\ same\ vehicle\ types \\ 0, otherwise \end{cases}$$

where $p_1, p_2 \in P$ and $p_1 \neq p_2$.

Then we have the symmetry breaking constraints $y_{p_1} \geq y_{p_2}$ (4.26) where $R^R_{p_1p_2} = 1$ and $\delta(p_1) < \delta(p_1)$.

Constraint (4.26) will not tighten the bound on the best possible integer solution, but they will make sure that route-vehicle pair p_1 is always activated before p_2. By incorporating these constraints, we prevent unnecessary branches that would arise by swapping identical route-vehicle pairs.

Next, we address the practical minimum utilization constraint. Suppose that M_p denotes the minimum vehicle utilization limit for route-vehicle pair $p \in P$. We can add constraints (4.27) to our formulation to ensure that if a route-vehicle pair is activated, then the total size of picked up orders and inbound transfers is greater than the provided minimum utilization limit.

$$\left(\sum_{\substack{i \in I \\ O^P_{ip}=1}} S_i x^P_{ip_1} + \sum_{i \in I} \sum_{\substack{p_2 \in P \\ p_2 \neq p_1}} \sum_{\substack{l_2 \in L^C \\ R_{p_2p_1l}=1}} S_i x^T_{ip_2p_1l} \right) \Big/ Q_{p_1} \geq M_{p_1} y_{r_1} \quad i \in I, p_1 \in P \qquad (4.27)$$

Proposed MILP formulation is computationally difficult to solve due to the high number of variables and constraints. Preliminary experiments showed that problem size grows quickly in instances with high consolidation opportunities. The main reason for the complexity is the 4-index t variable and the constraints that deal with transfers. Therefore, an alternative strategy is required for solving practical size instances. In the next section we propose a matheuristic approach for solving larger instances of LTLP-C in shorter computation times.

4.2.2 A Matheuristic for LTLP-C

A heuristic algorithm that utilizes mathematical programming models in its steps is called a *matheuristic*. This type of algorithms has been applied to different variants of VRPs. Archetti and Speranza [93] present a classification of matheuristic approaches for routing problems.

We propose a matheuristic algorithm for solving the LTLP-C. This algorithm consists of a construction phase and an improvement phase. In the construction phase, we employ a fix-and-optimize heuristic to build an initial solution. We then improve the initial solution by a simulated annealing (SA) algorithm which incorporates a tabu search (TS) mechanism. In both phases, we solve smaller sub-problems of the LTLP-C with exact solution methods. Next, we describe the algorithms developed for construction and improvement phases in detail.

Construction Matheuristic

The construction matheuristic uses our route generation method and a simplified version of the MILP model for the LTLP-C. In this version, decision variables and constraints related to cross-docking are excluded. Hereafter we call this problem as the LTLP-C-Sub. The LTLP-C-Sub is a special case of the LTLP-C in which only direct routes are used. A typical solution to the LTLP-C-Sub contains direct routes from pickup locations to delivery locations. The LTLP-C-Sub can be partitioned into multiple smaller sub-problems based on the pickup and delivery zones of the orders. For further computational efficiency, we solve separate sub-problems for orders of each zone-to-zone pair or intra-zone orders of each zone.

During the development of the matheuristic, we were inspired by several ideas that have been taken into consideration by the on-site planners of the 3PL company over the years. We start the algorithm with the vehicle types having the largest capacity. We try to find highly utilized direct routes by generating direct-type routes only. We decompose the remaining orders into multiple legs by setting their origin or destination locations to cross-docking locations. We also divide the orders based on their origin and destination zones and solve smaller sub-problems for each group. As we proceed through the algorithm, we switch to smaller vehicles at certain steps. We keep the multi-leg orders synchronized by adjusting their deadlines when necessary.

The LTLP-C-Sub formulation has a similar notation as the one proposed for LTLP-C in Section 4.2.1. We use a single type of binary variable to indicate the assignment of order to a route-vehicle pair, x_{ip}, which takes the value of one if order $i \in I$ is assigned to route-vehicle pair $p \in P$, and zero, otherwise. x_{ip} is defined only when $O_{ip} = 1$, that

is, order i can be assigned to route-vehicle pair p. The resulting formulation of LTLP-C-Sub is given follows:

$$Minimize \quad \sum_{p \in P} C_p^F y_p + \sum_{i \in I} C_i^U u_i \tag{4.2}$$

$subject\ to$

$$\sum_{\substack{r \in R \\ O_{ip}=1}} x_{ip} + u_i = 1 \qquad i \in I \tag{4.28}$$

$$x_{ip} \leq y_p \qquad i \in I, p \in P, O_{ip} = 1 \tag{4.29}$$

$$\sum_{\substack{i \in I \\ O_{ip}=1}} S_i x_{ip} \leq Q_p \qquad p \in P \tag{4.30}$$

$$a_p + T^A_{l_i^D p} \leq D_i + T^P\left(1 - x_{ip}\right) \qquad i \in I, p \in P, O_{ip} = 1 \tag{4.31}$$

$$\sum_{\substack{i \in I \\ O_{ip}=1}} S_i x_{ip} \geq M_p Q_p y_p \qquad p \in P \tag{4.32}$$

$$x_{ip} \in \{0, 1\} \qquad i \in I, p \in P, O_{ip} = 1 \tag{4.33}$$

$$u_i \in \{0, 1\} \qquad i \in I \tag{4.22}$$

$$y_p \in \{0, 1\} \qquad p \in P \tag{4.23}$$

$$a_p \geq 0 \qquad p \in P \tag{4.25}$$

The objective function of the LTLP-C-Sub is the same as that of the LTLP-C. Constraints (4.28), (4.29), (4.30), (4.31) and (4.32) are analogous to constraints (4.3), (4.10), (4.14), (4.15) and (4.27) in the LTLP-C model, respectively.

We solve instances of LTLP-C-Sub at certain steps of the construction matheuristic. The general flow our constructive matheuristic algorithm are demonstrated in Figure 4.1, where the route type numbers are adopted from Figure 3.4.

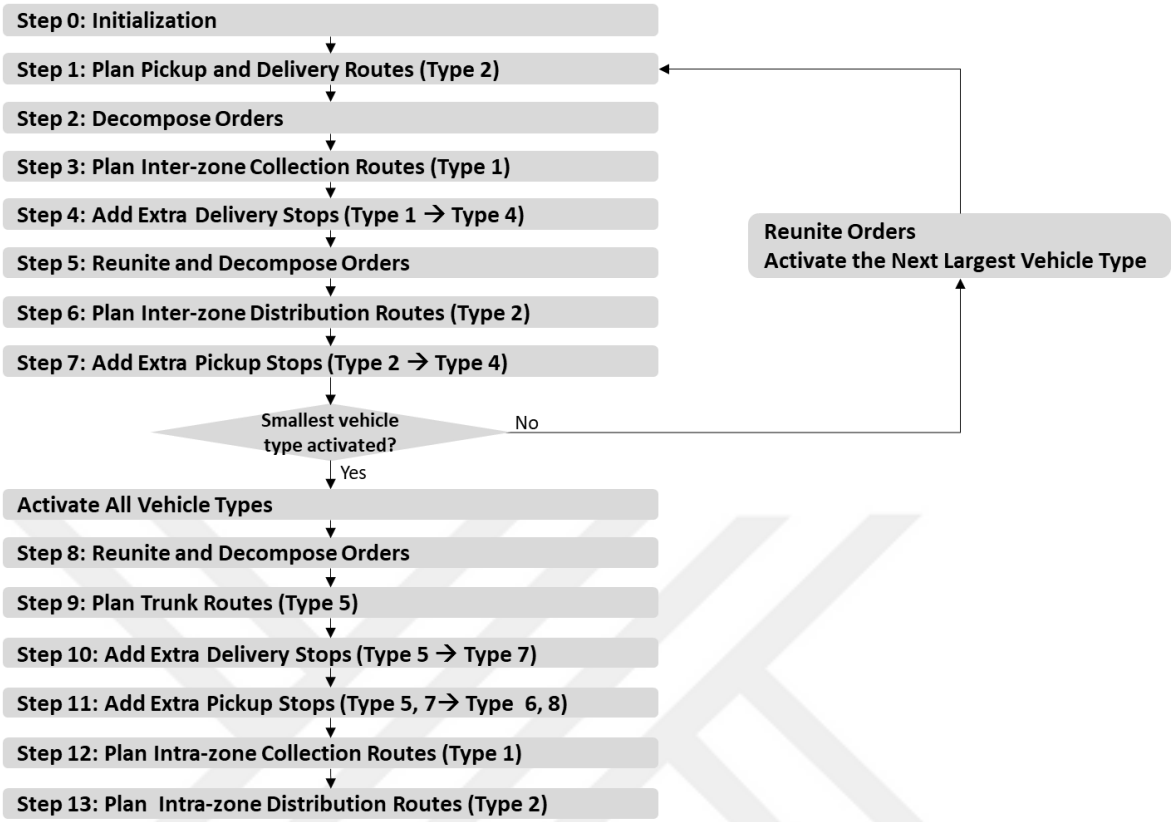


Figure 4.1. Construction Matheuristic for LTLP-C.

The algorithm starts with the vehicle type having the largest capacity. At each planning step (steps 1, 3, 6, 9, 12 and 13) we partition the active orders based on their origin and destination zones. For each segment, we generate the set or route-vehicle pairs for the relevant subset of orders and solve the LTLP-C-Sub. We keep track of four lists for orders and one list for activated route-vehicle pairs. A more detailed step-by-step explanation of the algorithm is provided in Appendix A.

This methodology aims to decompose the LTLP-C into computationally tractable sub-problems based on he current planning approach employed by the firm. The synchronization of orders is maintained at each step and the feasibility of the solution is preserved. An advantage of this procedure is that the LTLP-C-Subs can be solved in parallel for each order segment at each planning step.

The solution of the constructive matheuristic provides a feasible transportation plan for the LTLP-C. In the next phase, we further improve this solution by an improvement matheuristic.

Improvement Matheuristic

In the second phase of the matheuristic algorithm, we aim to improve the solution found in the construction phase by iteratively destroying and repairing parts of the solution by a tabu search (TS) algorithm implemented within a simulated annealing (SA) framework. A single destroy operator and three repair operators are developed for the improvement phase. Each repair operation involves the solution of the LTLP-C model for a subset of orders and route types. Several parameters are defined for the improvement algorithm. A list of algorithm parameters and their descriptions are provided in Table 4.1.

Table 4.1. Algorithm Parameters for Improvement Matheuristic.

Parameter	Description
$\boldsymbol{T_{start}}$	Starting temperature
$\boldsymbol{T}$	Current temperature
$\boldsymbol{CR}$	Cooling rate
$\boldsymbol{S_{init}}$	Initial solution obtained from construction matheuristic
$\boldsymbol{S_{act}}$	Active solution
$\boldsymbol{S_{best}}$	Best solution obtained so far
$\boldsymbol{S_{cand}}$	Candidate solution
$\boldsymbol{C_S}$	Total cost of solution S
$\boldsymbol{N}$	Maximum number of iterations
$\boldsymbol{P_T}$	Set of route-vehicle pairs in the tabu list
$\boldsymbol{P_S}$	Set of route-vehicle pairs of solution S
$\boldsymbol{K}$	Maximum number of iterations for a route-vehicle pair to stay in the tabu list

We define an acceptance probability function $AP\left(C_{S_{cand}}, C_{S_{best}}, T\right)$, which returns the probability of S_{cand} being accepted as the next active solution given that the best solution cost up to that point is $C_{S_{best}}$ and the current temperature is T. We also assume that the function $RAND(0,1)$ generates a value in the range $[0,1]$, uniformly at random.

The overview of the improvement algorithm is given as follows:

Algorithm 4.2. Improvement Matheuristic

1: *Initialization:* $T := T_{start}$, $S_{act} := S_{init}$, $S_{best} := S_{init}$, $R_T := \{\ \}$
2: *For i = 0 through N:*
3: *If* $P_{S_{act}} \subseteq P_T$, *stop. Report* S_{best}

4: *Else, pick some route-vehicle pair* $p \in P_{S_{act}} \setminus P_T$. *Determine the set of route-vehicle pairs to be removed from* $P_{S_{act}}$ *based on the selected route-vehicle pair* p

5: *Destroy* S_{act} *by removing the selected set of route-vehicle pairs from* $P_{S_{act}}$

6: *Select a repair operator*

7: *Apply the selected repair operator to obtain* S_{cand}

8: *If* $AP\left(C_{S_{cand}}, C_{S_{best}}, T\right) \geq RAND(0,1)$*:*

9: $S_{act} := S_{cand}$

10: *Append the set of newly formed route-vehicle pairs by the repair operation to* P_T

11: *If* $C_{S_{cand}} < C_{S_{best}}$, *then* $S_{best} := S_{cand}$

12: *Else:*

13: *Append the set of route-vehicle pairs removed from* S_{act} *by the destroy operation to* P_T

14: $T := T * CR$

15: *Update* P_T *by removing the route-vehicle pairs that have been in* P_T *for* K *iterations*

16: *Report* S_{best}

We use the acceptance probability function defined by Metropolis et al. [164]. This function is defined as:

$$AP\left(C_{S_{cand}}, C_{S_{best}}, T\right) = \begin{cases} e^{-\left(C_{S_{cand}} - C_{S_{best}}\right)/T}, & if\ C_{S_{cand}} > C_{S_{best}} \\ 1, & otherwise \end{cases} \quad (4.34)$$

The algorithm starts with the initial solution found in the construction phase. The main idea is to change the active solution iteratively by destroy and repair operations until a stopping criterion is met. The algorithm stops either when all route-vehicle pairs of the active solution are in the tabu list or when the iteration limit is reached. In either case, the best solution found up to that point is reported.

When a candidate solution is obtained by applying destroy and repair operators on the active solution, the solution is accepted with a probability which is calculated by the acceptance probability function. If the candidate solution is accepted, it is assigned as the new active solution. All newly activated route-vehicle pairs during this operation are added to the tabu list so that they are not ruined for a certain number of iterations. If the candidate solution is better than the best solution available, it is further assigned as the best solution. If the candidate solution is not accepted, then the set of route-vehicle pairs

which are removed from the active solution during the destroy operation are added to the tabu list, in order not to repeat the same type of destroy operation for a given number of steps. We update the value of the temperature parameter and the tabu list at the end of each iteration. If some route-vehicle pairs have been waiting in the tabu list for K iterations, then they are removed from the tabu list. Next we explain the destroy and repair operators used in detail.

Destroy Operator: In a typical solution to the LTLP-C, most of the route-vehicle pairs are dependent on other route-vehicle pairs due to transfers between each other. Therefore, removing one route-vehicle pair from a solution usually renders the solution infeasible. The main idea behind the destroy operator is to find a set of route-vehicle pairs such that no element of the set is dependent on some other route-vehicle pair outside the set. On the other hand, destroying and repairing a large set of route-vehicle pairs would create a higher possibility of improving the solution. In the destroy operation, we assign weights to each route-vehicle pair of the active solution. These weights are given based on the route-vehicle pair's dependency and relatedness on other route-vehicle pairs. Any route-vehicle pair that has a commonly assigned order with a given route-vehicle pair p in a certain solution is considered to be a *dependent* route-vehicle pair of p. The set of all dependent route-vehicle pairs of p is represented by DP_p. We further assume that any two route-vehicle pairs are *related* to each other if and only if their pickup zones and/or delivery zones are the same and they are not dependent on each other. If only one zone is in common, then the route-vehicle pairs are *weakly related*. If both pickup and delivery zones are in common, then they are considered to be *strongly related*. WR_p and SR_p denote the sets of weakly and strongly related route-vehicle pairs of p, respectively. Then the weight (W_p) of a route-vehicle pair $p \in P_{S_{act}}$ is calculated as:

$$W_p = \begin{cases} \alpha|DP_p| + \beta|SR_p| + \gamma|WR_p|, & if\ p \in P_{S_{act}} \setminus P_T \\ 0, & if\ p \in P_T \end{cases} \qquad p \in P_{S_{act}} \tag{4.35}$$

where α, β and γ are positive real valued algorithm parameters which satisfy $\alpha > \beta > \gamma$. These weights are converted into selection probabilities such that the probability of route-vehicle p to be selected is P_p where:

$$P_p = \frac{W_p}{\sum_{p_1 \in P_{S_{act}}} W_{p_1}} \qquad p \in P_{S_{act}} \qquad (4.36)$$

The first route-vehicle to be removed from the solution is chosen by the roulette wheel selection method with respect to the calculated selection probabilities. By assigning a higher probability to a route-vehicle pair with higher number of dependent and related route-vehicle pairs, we aim to increase the possibility of destroying a larger portion of the active solution. We expect this strategy to increase the effectiveness of our improvement algorithm.

As we remove a route-vehicle pair from the active solution, we remove its dependent route-vehicle pairs from the active solution recursively. This continues until no route-vehicle pair in the set of route-vehicle pairs to be removed has a dependent route-vehicle pair outside the set.

When we remove a route-vehicle pair from a solution, orders carried by the vehicle associated with that route-vehicle pair become unassigned. Let us denote the set of orders that become unassigned after a destroy operation by I_d. If the number of elements in I_d is too low, then the effect of a repair operation would be limited. In order to avoid this, we define a lower limit on the number of orders to be removed from the solution, which is represented by O_{min}. When $|I_d| < O_{min}$, we apply the following procedure to increase the number of elements in I_d:

Algorithm 4.3. Procedure for Increasing the Number of Orders in I_d

1: *Search the initially unassigned orders of S_{act}. If there is any unassigned order that has the same pickup or delivery zone as the first route-vehicle pair removed from the solution, append this order to I_d. Stop if $|I_d| \geq O_{min}$. Otherwise, proceed to the next step.*

2: *Form a subset of SR_p where p is the first route-vehicle pair removed from the solution such that only the route-vehicle pairs which are not yet removed from the active solution are included. If the newly formed set is empty, proceed to the next step. Otherwise randomly pick a route-vehicle pair from the set. Remove the selected route-vehicle pair and its subsequent dependent route-vehicle pairs from the set and the active solution. Append any orders that become unassigned to I_d. Stop if $|I_d| \geq O_{min}$. Otherwise, proceed to the next step.*

3: *Form a subset of WR_p, instead of SR_p and apply the same operations as the previous step. If at any time the newly formed set becomes empty or the stopping criterion $|I_d| \geq O_{min}$ holds, terminate the procedure.*

At the end of this process, we obtain the finalized set of orders I_d, which serves as the main input to our repair operators.

Repair Operators: In the repair operation we need to solve the LTLP-C for a subset of orders, denoted by I_d. We address three different repair operators which are named as *Repair-1*, *Repair-2* and *Repair-3*, respectively. We define a special parameter, O_{max}, which denotes the maximum size of an order set that can be solved by the exact solution algorithm proposed for the base problem. When $|I_d| \leq O_{max}$, we apply Repair-1 operator, which solves the LTLP-C model for order set I_d. If $|I_d| > O_{max}$, we choose either Repair-2 or Repair-3, in which different types of clustering mechanisms are applied to reduce the number of orders in a single MILP solution. Repair-2 and Repair-3 are equally likely to be selected. The three repair operators are defined as follows:

Algorithm 4.4. Repair-1

1: *Derive the initial order to route-vehicle pair assignments of the destroyed part of S_{act}. Set these assignments as a starting solution for the LTLP-C.*
2: *Solve the LTLP-C with the order set I_d. Obtain the set of newly activated route-vehicle pairs (P_{Rep}) and a set of unassigned orders $I_u \subseteq I_c$.*
3: *Append the set of route-vehicle pairs P_{Rep} and the set of unassigned orders I_u to the destroyed solution to get S_{cand}.*

An important feature of the Repair-1 operator is that it cannot return a worse solution than S_{act}, even if the MILP for the LTLP-C is not solved optimally due to the fact that the destroyed part of the solution is provided to the solver as a starting solution.

The main purpose of the Repair-2 operator is to find a single subset of orders $I_c \subseteq I_d$ with the size of O_{max} and solve the LTLP-C for orders in I_c. Afterwards, the matheuristic construction algorithm is executed for the unassigned orders obtained from the MILP solution and the orders of I_d which are not included in subset I_c. This procedure is explained in Algorithm 4.5.

Algorithm 4.5. Repair-2

1: *Find a subset of orders $I_c \subseteq I_d$ where $|I_c| = O_{max}$.*
2: *Apply the matheuristic construction algorithm for I_c. Set the solution of the construction algorithm as a starting solution for the* LTLP-C.
3: *Solve the LTLP-C with the order set I_c. Obtain the set of newly activated route-vehicle pairs (P_{Rep_1}) and a set of unassigned orders $I_{u_1} \subseteq I_c$.*

4: *Apply the matheuristic construction algorithm for* $(I_d \setminus I_c) \cup I_{u_1}$. *Obtain the set of newly activated route-vehicle pairs* (R_{Rep_2}) *and a set of unassigned orders* $I_{u_2} \subseteq \left((I_d \setminus I_c) \cup I_{u_1}\right)$

5: *Append the sets of route-vehicle pairs* R_{Rep_1} *and* R_{Rep_2} *and the set of unassigned orders* I_{u_2} *to the destroyed solution to get* S_{cand}.

We form the subset of orders (I_c) with a randomized selection mechanism which aims to bring highly related orders together in order to increase consolidation opportunities. To do this, we define an *order relatedness indicator*, $OR_{i_1 i_2}$ between orders i_1 and i_2. Orders i_1 and i_2 are considered to be not related if the sum of order sizes $(S_{i_1} + S_{i_2})$ is greater than the capacity of the largest vehicle. In this case, these two orders cannot be consolidated, and we assume that $OR_{i_1 i_2} = 0$. Otherwise, we address two other indicators, $PR_{i_1 i_2}$ and $DR_{i_1 i_2}$.

Assume that the distance between two locations, l_1 and l_2 is denoted by $H_{l_1 l_2}$. l_i^O and $l_i^{C_P}$ denotes the pickup location of order i and the cross-docking location in the pickup zone of order i, respectively. Similarly, l_i^D and $l_i^{C_D}$ stand for the delivery location of order i and the cross-docking location in the delivery zone of order i, respectively. $PR_{i_1 i_2}$ and $DR_{i_1 i_2}$ are defined as follows:

$$PR_{i_1 i_2} = min\left\{H_{l_{i_1}^O l_{i_2}^O} + H_{l_{i_2}^O l_{i_2}^{C_P}}, H_{l_{i_2}^O l_{i_1}^O} + H_{l_{i_1}^O l_{i_1}^{C_P}}\right\} \Big/ max\left\{H_{l_{i_1}^O l_{i_1}^{C_P}}, H_{l_{i_2}^O l_{i_2}^{C_P}}\right\} \quad (4.37)$$

$$DR_{i_1 i_2} = min\left\{H_{l_{i_1}^{C_D} l_{i_1}^D} + H_{l_{i_1}^D l_{i_2}^D}, H_{l_{i_2}^{C_D} l_{i_2}^D} + H_{l_{i_2}^D l_{i_1}^D}\right\} \Big/ max\left\{H_{l_{i_1}^{C_D} l_{i_1}^D}, H_{l_{i_2}^{C_D} l_{i_2}^D}\right\} \quad (4.38)$$

These indicators show how consolidating the two orders change the distance of the direct collection or last-mile delivery route of the order which has the farther pickup location from the cross-docking in the pickup zone or the farther delivery location from the cross-docking location in the delivery zone. Eventually, $OR_{i_1 i_2}$ is calculated as follows:

Algorithm 4.6. Calculation of $OR_{i_1 i_2}$

1: $OR_{i_1 i_2} := 0$

2: *If* $S_{i_1} + S_{i_2}$ *is less than or equal to the capacity of the largest vehicle:*

3: *If the pickup locations of orders* i_1 *and* i_2 *are in the same zone:*

4: $OR_{i_1 i_2} := OR_{i_1 i_2} + 0.5/PR_{i_1 i_2}$
5: *If the delivery locations of orders i_1 and i_2 are in the same zone:*
6: $OR_{i_1 i_2} := OR_{i_1 i_2} + 0.5/DR_{i_1 i_2}$

This procedure produces a relatedness score $OR_{i_1 i_2}$ in the range $[0,1]$. The total relatedness of order i_1 to a set of orders I_c is represented by CR_{i_1,I_c} where:

$$CR_{i_1,I_c} = \sum_{\substack{i_2 \in I_c \\ i_1 \neq i_2}} OR_{i_1 i_2} \qquad (4.39)$$

The subset I_c is formed via Algorithm 4.7.

Algorithm 4.7. Formation of Order Set I_c

1: $I_c := \{\ \}$
2: *Apply roulette wheel selection to pick the first order to be inserted to I_c with probabilities $CR_{i_1,I_d}/\sum_{i_2 \in I_d} CR_{i_2,I_d}$ for $i_1 \in I_d$.*
3: *Append the selected order to I_c, remove the selected order from I_d.*
4: *While $|I_c| \leq O_{max}$:*
5: *Apply roulette wheel selection to pick the next order to be inserted to I_c with probabilities $CR_{i_1,I_c}/\sum_{i_2 \in I_d \setminus I_c} CR_{i_2,I_c}$ for $i_1 \in I_d \setminus I_c$*
6: *Append the selected order to I_c*

For the Repair-3 operator, we define a new limit on the size of the order set to be solved by the MILP for LTLP-C. This value is denoted by O_{max2}, which satisfies $O_{max2} > O_{max}$. The idea of this operator is to cluster the set of orders I_d into n distinct subsets such that each cluster is denoted by I_{c_1} through I_{c_n}, where $|I_{c_k}| < O_{max2}$ for $k = 1,..,n$. The number of clusters, n is calculated as $n = \left\lceil \frac{|I_d|}{O_{max2}} \right\rceil$. Then, the MILP is solved with a restricted set of route types for each cluster. Route types that contain more than one cross-docking stops are not provided to the model. At the end we solve the matheuristic construction algorithm for the union of the sets of unassigned orders for each cluster (I_{u_1} through I_{u_n}). The steps of the Repair-3 operator are given in Algorithm 4.8.

Algorithm 4.8. Repair-3

1: *Cluster orders and obtain n mutually exclusive and collectively exhaustive subsets of I_d, which are denoted by I_{c_k} where $|I_{c_k}| \leq O_{max2}$ for $k = 1,..,n$.*
2: *For $k = 1$ through n:*

3: *Solve matheuristic construction algorithm for I_{c_k}. Set the solution of the construction algorithm as a starting solution for LTLP-C.*

4: *Solve the LTLP-C with the order set I_{c_k} and route types 1, 2, 3, and 4. Obtain the set of newly activated route-vehicle pairs (P_{Rep_k}) and a set of unassigned orders $I_{u_k} \subseteq I_{c_k}$.*

5: *Apply matheuristic construction algorithm for $\bigcup_{k=1}^{n} I_{u_k}$. Obtain the set of newly activated route-vehicle pairs ($P_{Rep_{n+1}}$) and a set of unassigned orders $I_{u_{n+1}} \subseteq \bigcup_{k=1}^{n} I_{u_k}$*

6: *Append the sets of route-vehicle pairs P_{Rep_1} through $P_{Rep_{n+1}}$ and the set of unassigned orders $I_{u_{n+1}}$ to the destroyed solution to get S_{cand}.*

For finding order clusters I_{c_1} through I_{c_n}, we make use of the same order relatedness indicators defined for Repair-2 operator. We define an *inverse relatedness score* IR_{i_1,I_c} between order i and order set I_c such that $IR_{i_1,I_c} = 1/CR_{i_1,I_c}$. Moreover the *average relatedness score* of I_c for order i is denoted by AR_{i_1,I_c} where $AR_{i_1,I_c} = CR_{i_1,I_c}/|I_c|$.

The resulting clustering algorithm is given as follows:

Algorithm 4.9. Randomized Order Clustering

1: $I_{c_1} \leftarrow \{\}$, $I_{c_2} \leftarrow \{\}$, …, $I_{c_n} \leftarrow \{\}$

2: *Randomly pick an order from I_d and append it to I_{c_1}*

3: *For $k = 2$ through n:*

4: *Apply roulette wheel selection to pick the order from the set I_d to be inserted to I_{c_k} with probabilities $IR_{i_1,\cup_{j=1}^{k-1} I_{c_j}} / \sum_{i_2 \in I_d} IR_{i_1,\cup_{j=1}^{k-1} I_{c_j}}$ for $i_1 \in I_d$.*

5: *Append the selected order to I_{c_k}, remove the selected order from I_d.*

6: *While $|I_d| > 0$:*

7: *Randomly pick an order i and remove it from I_d.*

8: *Apply roulette wheel selection to select a cluster for order i to be inserted with probabilities $AR_{i,I_{c_k}} \Big/ \sum_{\substack{j=1,\dots,n \\ |I_{c_j}|<O_{max2}}} AR_{i,I_{c_j}}$ for $k = 1, \dots, n$ and $|I_{c_k}| < O_{max2}$.*

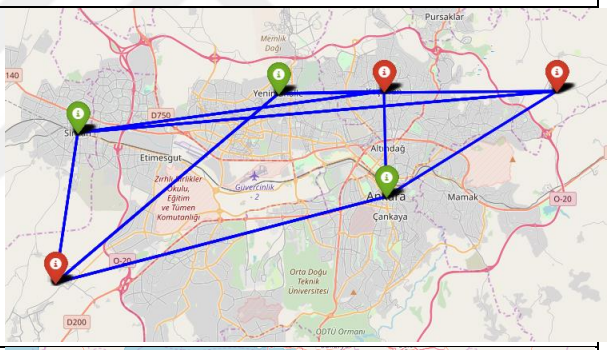
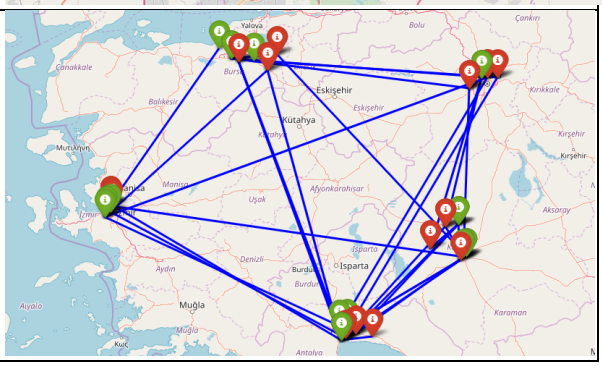
9: *Append order i to the selected cluster.*

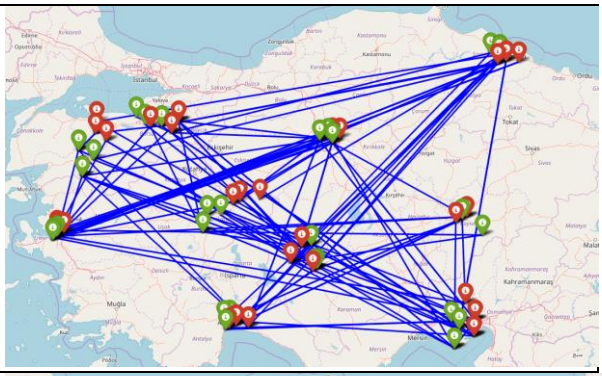
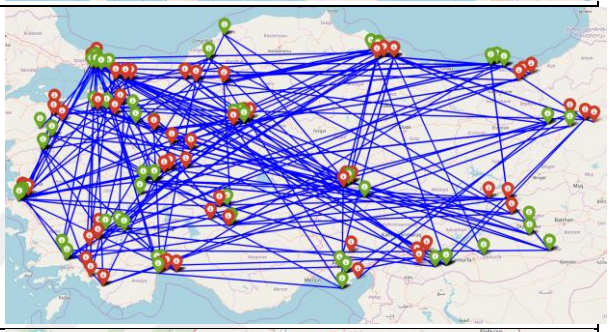
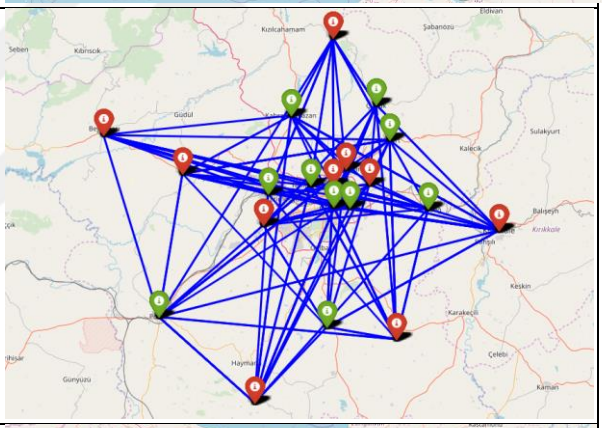
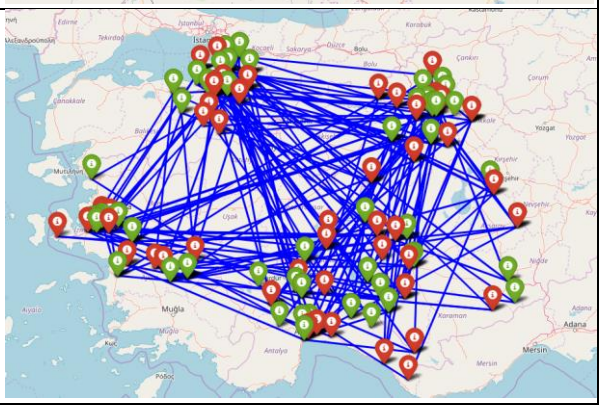
In this section we described the details of our 2-phase matheuristic algorithm which addresses the LTLP-C. The next section demonstrates the results of our computational experiments on instances generated using the real order data of the 3PL company.

4.2.3 Computational Experiments for the LTLP-C

In this computational study, compare the results of our matheuristic approach against exact MILP solutions for instances with different types and sizes.

We developed 10 types of instances. Each instance type is defined in terms of the number of zones and the number of possible pickup and delivery locations at each zone. For 9 of the instance types, orders are randomly generated by selecting a pickup and delivery location from the sets of possible pickup and delivery locations, respectively. Order sizes are obtained by random sampling from 3PL companies' real order database. Instances of the final type consist of a randomly selected subset of the 3PL company's daily orders. The description of instance types is provided in Figure 4.2.

Instance Type	**Description**	**Example Instance**
1Z-3L	Single zone, 3 possible pickup locations and 3 possible delivery locations	
5Z-3L	5 zones, 3 possible pickup locations and 3 possible delivery locations at each zone	

10Z-3L	10 zones, 3 possible pickup locations and 3 possible delivery locations at each zone	
20Z-3L	20 zones, 3 possible pickup locations and 3 possible delivery locations at each zone	
1Z-10L	Single zone, 10 possible pickup locations and 10 possible delivery locations	
5Z-10L	5 zones, 10 possible pickup locations and 10 possible delivery locations at each zone	

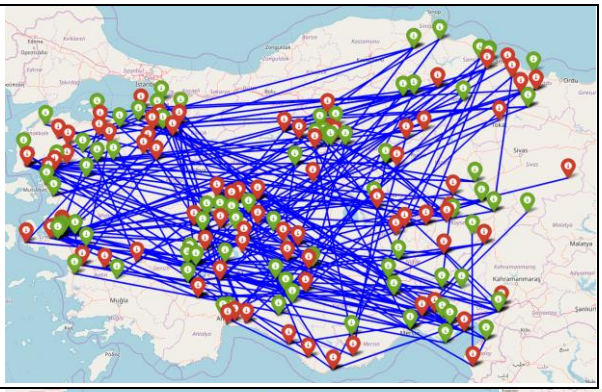
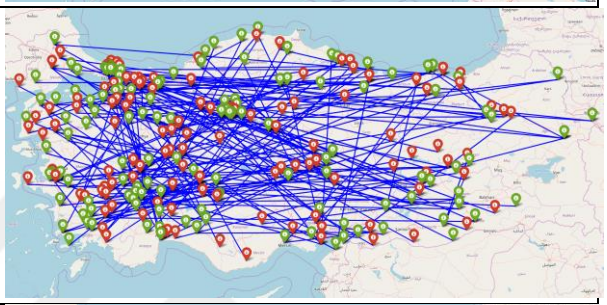
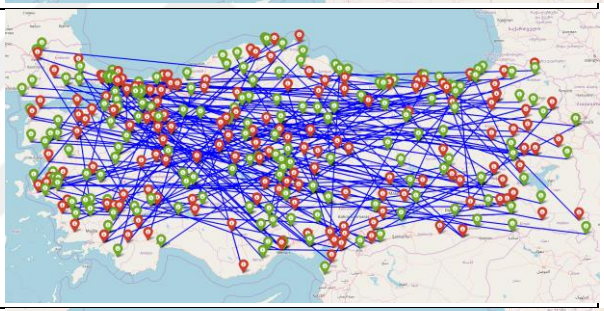
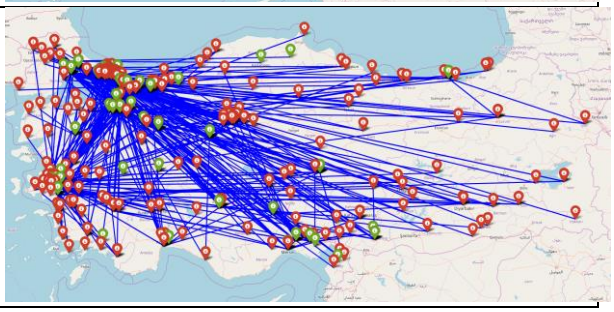
10Z-10L	10 zones, 10 possible pickup locations and 10 possible delivery locations at each zone	
20Z-10L	20 zones, 10 possible pickup locations and 10 possible delivery locations at each zone	
20Z-ALL	20 zones, all 957 locations in the network are possible pickup and delivery locations	
20Z-SUB	20 zones, randomly selected subset of real orders	

Figure 4.2. Sample Illustration of Test Instances.

In the example instance illustration, the green and red colored pins represent the pickup and delivery locations of the orders, respectively. Each blue line represents an order.

We assume that there exist two types of vehicles in terms of their physical attributes. All costs are proportionally altered from their original values. We use a distance matrix of real driving distances. Due dates for deliveries are assumed to be 96 hours for all orders.

Values of the rest of the common parameters that we use in both types of experiments are tabulated in Appendix B.

We generate 12 instances for each instance type, where the number of orders in each instance ranges from 10 to 500. Each instance is labelled as the name of the instance type, followed by the number of orders in the instance. For example, the instance of type 5Z-10L with 100 orders is named as 5Z-10L-100. For each instance type, instances grow cumulatively, that is, new orders are added to the existing set of orders to obtain larger instances.

The MILP for LTLP-C is solved with a time limit of 3 hours (10,800 seconds) using CPLEX 12.8. The LTLP-C-Subs within the construction matheuristic and the other MILP models within the improvement matheuristic are solved with a time limit of 10 minutes (600 seconds). The values of all matheuristic parameters are provided in Appendix C. All experiments were run on a 24-core workstation with 2.40 GHz processors and 128 GB RAM. Table 4.2 provides a description of the notation used in the results table. The results of the experiments are tabulated in Table 4.3.

Table 4.2. Description of Result Parameters.

Parameter	Description
C_{MILP}	MILP solution cost
GAP_{MILP}	MILP GAP
C^{I}_{MH}	Matheuristic initial solution cost
C^{F}_{MH}	Matheuristic final solution cost
$\%\Delta_{C}$	Percentage cost difference (Matheuristic - MILP Solution Cost)
T_{MILP}	MILP solution time
T^{init}_{MH}	Matheuristic initial solution time (seconds)
T^{total}_{MH}	Matheuristic total solution time (seconds)

Table 4.3. Comparison of MILP and Matheuristic Approaches for LTLP-C.

Instance	C_{MILP}	GAP_{MILP}	C^{l}_{MH}	C^{F}_{MH}	$\%\Delta_C$	T_{MILP}	T^{init}_{MH}	T^{total}_{MH}
1Z-3L-10	**4477.18**	**0.0%**	**4477.18**	**4477.18**	**0.0%**	**1.75**	**0.24**	**0.37**
1Z-3L-15	5595.26	0.0%	5663.40	5663.40	1.2%	7.98	0.23	0.75
1Z-3L-20	7976.99	0.0%	8905.48	8569.68	7.4%	24.58	0.37	1.88
1Z-3L-25	9725.94	0.0%	9917.43	9917.43	2.0%	107.92	0.53	1.46
1Z-3L-35	12610.17	0.0%	12917.73	12917.73	2.4%	394.76	1.36	3.82
1Z-3L-50	**13238.82**	**0.0%**	**13238.82**	**13238.82**	**0.0%**	**4020.29**	**6.45**	**12.47**
1Z-3L-75	18894.88	4.8%	20887.79	20887.79	10.5%	10808.55	85.83	92.97
1Z-3L-100	1024056.00	99.7%	28689.42	28689.42	-97.2%	10801.56	116.69	122.88
1Z-3L-200	-	-	43008.20	42996.16	-	-	605.34	611.16
1Z-3L-300	-	-	61684.05	61682.59	-	-	611.12	616.64
1Z-3L-400	-	-	83030.87	82040.30	-	-	622.52	628.62
1Z-3L-500	-	-	250648.21	154041.34	-	-	2620.38	3980.42
5Z-3L-10	**19523.14**	**0.0%**	**19561.50**	**19523.14**	**0.0%**	**14.21**	**0.82**	**2.61**
5Z-3L-15	**28161.32**	**0.0%**	**28333.68**	**28161.32**	**0.0%**	**72.47**	**1.15**	**4.03**
5Z-3L-20	33573.64	0.0%	34318.69	33816.40	0.7%	268.15	1.31	15.50
5Z-3L-25	40906.17	0.0%	42092.61	41340.27	1.1%	1332.32	1.48	271.17
5Z-3L-35	44433.61	3.8%	46892.17	46850.25	5.4%	10816.17	1.88	320.78
5Z-3L-50	474272.00	95.1%	66159.96	61693.75	-87.0%	10815.62	2.12	1637.44
5Z-3L-75	-	-	86279.76	84518.27	-	-	2.65	2334.45
5Z-3L-100	-	-	111450.75	105489.48	-	-	2.84	4026.80
5Z-3L-200	-	-	189775.58	181188.98	-	-	6.55	6507.87
5Z-3L-300	-	-	264611.79	261331.30	-	-	94.93	6546.94
5Z-3L-400	-	-	351462.40	345692.65	-	-	1639.64	7712.89
5Z-3L-500	-	-	447411.51	438194.81	-	-	3897.89	8011.71
10Z-3L-10	**22462.32**	**0.0%**	**22824.26**	**22467.15**	**0.0%**	**4.76**	**0.96**	**1.48**
10Z-3L-15	**30760.94**	**0.0%**	**31449.53**	**30760.94**	**0.0%**	**26.53**	**1.30**	**6.11**
10Z-3L-20	**42790.62**	**0.0%**	**43479.21**	**42790.62**	**0.0%**	**79.71**	**1.65**	**15.58**
10Z-3L-25	**55225.03**	**0.0%**	**57906.39**	**55225.03**	**0.0%**	**419.21**	**2.04**	**144.30**
10Z-3L-35	73374.07	0.0%	76863.38	73491.70	0.2%	1130.04	2.43	344.57
10Z-3L-50	107674.68	0.0%	111951.35	111951.35	4.0%	4568.25	5.27	432.27
10Z-3L-75	144387.65	15.9%	148875.95	146771.66	1.7%	10865.15	4.52	1754.09
10Z-3L-100	-	-	193628.98	193628.98	-	-	6.88	2330.12
10Z-3L-200	-	-	343898.43	343898.43	-	-	15.38	3434.69
10Z-3L-300	-	-	440722.42	440722.42	-	-	65.39	3546.65
10Z-3L-400	-	-	537813.69	537813.69	-	-	684.57	4552.55
10Z-3L-500	-	-	624233.82	624233.82	-	-	795.45	5394.98
20Z-3L-10	**26836.10**	**0.0%**	**26836.10**	**26836.10**	**0.0%**	**7.07**	**0.96**	**1.64**
20Z-3L-15	**43859.34**	**0.0%**	**43859.34**	**43859.34**	**0.0%**	**13.69**	**1.46**	**2.59**
20Z-3L-20	**57355.39**	**0.0%**	**57355.39**	**57355.39**	**0.0%**	**28.46**	**2.17**	**4.55**
20Z-3L-25	**70148.02**	**0.0%**	**70148.02**	**70148.02**	**0.0%**	**36.00**	**2.27**	**4.81**

20Z-3L-35	**99130.54**	**0.0%**	**99130.54**	**99130.54**	**0.0%**	**102.71**	**3.42**	**7.47**
20Z-3L-50	**139720.32**	**0.0%**	**139720.32**	**139720.32**	**0.0%**	**377.13**	**4.69**	**52.26**
20Z-3L-75	191202.85	0.0%	193239.92	191605.29	0.2%	2383.33	6.61	103.31
20Z-3L-100	252035.42	0.0%	254923.87	254708.36	1.1%	8396.02	7.80	117.96
20Z-3L-200	-	-	513362.20	513362.20	-	-	20.72	212.08
20Z-3L-300	-	-	731196.47	731196.47	-	-	59.76	519.02
20Z-3L-400	-	-	947007.82	947007.82	-	-	303.12	1554.73
20Z-3L-500	-	-	1105197.17	1105197.17	-	-	806.96	5496.77
1Z-10L-10	**6474.62**	**0.0%**	**6474.62**	**6474.62**	**0.0%**	**7.15**	**0.25**	**0.79**
1Z-10L-15	**8087.50**	**0.0%**	**8087.50**	**8087.50**	**0.0%**	**41.07**	**0.86**	**17.23**
1Z-10L-20	**9347.63**	**0.0%**	**9347.63**	**9347.63**	**0.0%**	**134.51**	**0.55**	**23.42**
1Z-10L-25	**9562.27**	**0.0%**	**9562.27**	**9562.27**	**0.0%**	**677.14**	**0.51**	**14.38**
1Z-10L-35	**11211.89**	**0.0%**	**11304.67**	**11213.86**	**0.0%**	**5885.30**	**1.02**	**134.66**
1Z-10L-50	20950.81	17.8%	18824.09	18824.09	-10.2%	10810.98	8.74	277.37
1Z-10L-75	-	-	25944.56	25937.10	-	-	7.98	660.80
1Z-10L-100	-	-	31383.89	31383.89	-	-	50.54	1157.57
1Z-10L-200	-	-	56256.43	55846.08	-	-	617.08	1863.10
1Z-10L-300	-	-	95439.09	93174.89	-	-	665.85	1974.02
1Z-10L-400	-	-	3835540.00	3835540.00	-	-	6779.72	6779.72
1Z-10L-500	-	-	4786531.00	4786531.00	-	-	20211.11	20211.11
5Z-10L-10	**20374.27**	**0.0%**	**21072.02**	**20374.27**	**0.0%**	**13.31**	**0.68**	**2.41**
5Z-10L-15	**29351.62**	**0.0%**	**29445.44**	**29351.62**	**0.0%**	**74.62**	**0.98**	**7.21**
5Z-10L-20	38544.91	0.0%	40602.09	38625.72	0.2%	345.16	1.24	13.93
5Z-10L-25	**44202.40**	**0.0%**	**47431.14**	**44202.40**	**0.0%**	**5616.14**	**1.37**	**76.60**
5Z-10L-35	67787.54	3.1%	72484.74	70564.82	4.1%	10817.51	1.90	821.08
5Z-10L-50	505914.00	100.0%	96982.23	93902.64	-81.4%	10914.60	2.10	1367.52
5Z-10L-75	-	-	128372.02	122511.39	-	-	3.37	1933.67
5Z-10L-100	-	-	167771.22	156430.75	-	-	8.22	2795.24
5Z-10L-200	-	-	289992.31	289992.31	-	-	32.23	5156.12
5Z-10L-300	-	-	407686.84	407686.84	-	-	61.32	8032.95
5Z-10L-400	-	-	513016.23	513016.23	-	-	323.74	9791.65
5Z-10L-500	-	-	587721.23	586329.34	-	-	500.74	10813.51
10Z-10L-10	**28879.10**	**0.0%**	**28879.10**	**28879.10**	**0.0%**	**1.65**	**0.87**	**1.20**
10Z-10L-15	41692.93	0.0%	42031.27	41722.27	0.1%	14.47	1.35	3.18
10Z-10L-20	55120.07	0.0%	55458.41	55149.41	0.1%	31.96	1.66	7.28
10Z-10L-25	**63776.76**	**0.0%**	**64115.10**	**63806.10**	**0.0%**	**47.95**	**1.92**	**9.42**
10Z-10L-35	**88957.85**	**0.0%**	**88962.28**	**88962.28**	**0.0%**	**251.78**	**2.34**	**49.83**
10Z-10L-50	123247.11	0.0%	123990.71	123747.07	0.4%	804.23	3.20	69.68
10Z-10L-75	169151.73	0.9%	173159.74	170526.29	0.8%	10834.44	4.47	172.52
10Z-10L-100	-	-	216459.53	215293.31	-	-	6.16	1154.91
10Z-10L-200	-	-	387941.56	387941.56	-	-	13.20	4813.68
10Z-10L-300	-	-	561510.30	561510.30	-	-	21.05	4932.95
10Z-10L-400	-	-	684757.88	684757.88	-	-	43.17	9433.98

10Z-10L-500	-	-	810394.44	810394.44	-	-	94.07	17691.87
20Z-10L-10	**28553.73**	**0.0%**	**28553.73**	**28553.73**	**0.0%**	**1.30**	**1.04**	**1.27**
20Z-10L-15	**43953.39**	**0.0%**	**43953.39**	**43953.39**	**0.0%**	**6.87**	**1.48**	**2.17**
20Z-10L-20	**57467.54**	**0.0%**	**57919.55**	**57467.54**	**0.0%**	**22.97**	**1.70**	**3.36**
20Z-10L-25	**74052.98**	**0.0%**	**78505.00**	**74052.98**	**0.0%**	**37.25**	**2.44**	**5.53**
20Z-10L-35	**101857.66**	**0.0%**	**107488.52**	**101857.66**	**0.0%**	**213.92**	**2.91**	**36.48**
20Z-10L-50	**155447.46**	**0.0%**	**160844.88**	**155447.46**	**0.0%**	**312.76**	**4.41**	**44.02**
20Z-10L-75	217213.66	0.0%	224052.91	218179.44	0.4%	6785.98	5.67	84.80
20Z-10L-100	285656.63	2.9%	297196.77	291697.89	2.1%	10819.97	8.06	262.04
20Z-10L-200	-	-	587513.78	587513.78	-	-	16.10	1638.56
20Z-10L-300	-	-	814799.56	814799.56	-	-	29.12	2002.08
20Z-10L-400	-	-	1041519.29	1041519.29	-	-	50.53	2044.45
20Z-10L-500	-	-	1227345.49	1227345.49	-	-	84.98	7214.94
20Z-ALL-10	**34097.78**	**0.0%**	**39769.06**	**34097.78**	**0.0%**	**1.26**	**1.00**	**1.26**
20Z-ALL-15	**49705.93**	**0.0%**	**54887.75**	**49705.93**	**0.0%**	**12.51**	**1.35**	**2.61**
20Z-ALL-20	**76455.57**	**0.0%**	**81675.71**	**76455.57**	**0.0%**	**18.13**	**1.82**	**3.85**
20Z-ALL-25	**90270.61**	**0.0%**	**95490.75**	**90270.61**	**0.0%**	**31.78**	**2.18**	**6.23**
20Z-ALL-35	**116130.38**	**0.0%**	**126294.43**	**116130.38**	**0.0%**	**65.14**	**2.87**	**7.49**
20Z-ALL-50	157160.32	0.0%	171078.04	160961.69	2.4%	306.35	4.01	36.04
20Z-ALL-75	236040.16	0.0%	246554.58	236367.28	0.1%	2203.79	5.59	51.35
20Z-ALL-100	321296.75	0.0%	325933.05	323392.09	0.7%	7203.97	6.45	143.53
20Z-ALL-200	-	-	600954.12	588119.79	-	-	15.29	372.43
20Z-ALL-300	-	-	853606.56	850969.13	-	-	27.75	715.10
20Z-ALL-400	-	-	1102998.35	1102998.35	-	-	63.36	2864.05
20Z-ALL-500	-	-	1288646.34	1288646.34	-	-	91.35	6307.88
20Z-SUB-10	**27147.53**	**0.0%**	**27147.53**	**27147.53**	**0.0%**	**5.26**	**1.09**	**1.59**
20Z-SUB-15	**42887.66**	**0.0%**	**43202.13**	**42887.66**	**0.0%**	**67.48**	**1.49**	**3.15**
20Z-SUB-20	46808.78	0.0%	47123.26	47123.26	0.7%	170.04	1.70	5.46
20Z-SUB-25	53496.80	0.0%	55339.69	54413.54	1.7%	1112.96	1.94	32.74
20Z-SUB-35	81356.61	6.3%	82045.95	81458.00	0.1%	10805.81	2.67	36.28
20Z-SUB-50	515287.00	85.1%	111532.53	109489.72	-78.8%	10809.63	3.77	125.99
20Z-SUB-75	-	-	153409.59	149555.59	-	-	5.61	773.09
20Z-SUB-100	-	-	200509.95	199842.96	-	-	11.06	2322.13
20Z-SUB-200	-	-	336410.76	336410.76	-	-	42.15	4648.85
20Z-SUB-300	-	-	481141.21	481141.21	-	-	226.95	4907.34
20Z-SUB-400	-	-	629431.93	629431.93	-	-	775.27	8675.51
20Z-SUB-500	-	-	782520.32	782520.32	-	-	2276.44	16796.57

The results chow that an optimum solution is found in 58 instances out of 120 by the exact MILP approach within the given time limit. In addition, 6 of the instances are solved with a gap below 10%. Instances 10Z-3L-75 and 1Z-10L-50 are solved with optimality gaps of 15.9% and 17.8%, respectively. However, in 4 instances, feasible solutions are

reported with a gap above 80%. In these cases, the solver produced a solution where all orders are unfulfilled. For the rest of the instances, either a feasible solution was not found, or the problem could not be generated by the solver due to an out-of-memory error.

The matheuristic algorithm is able to find the optimum solution in 38 out of the 58 instances where the optimum solution is available. In the remaining 20 instances, the average percentage difference between the matheuristic final solution cost and the optimum solution cost is found as 1.4%. For the 6 instances where the optimality gap is below 10%, the average percentage difference between the matheuristic final solution cost and the best integer feasible solution reported by the solver is 3.9%. The matheuristic solution costs are 1.4% higher and 10.2% lower than the best integer solution costs for instances 10Z-3L-75 and 1Z-10L-50, respectively.

We can easily conclude that the matheuristic performs significantly better than the exact MILP in terms of solution time. Moreover, it is able to find feasible solutions for large instances where the MILP fails to do so. Exceptions are instances 1Z-10L-400 and 1Z-10L-500, for which the matheuristic does not assign any orders to route-vehicle pairs since the LTLP-C-Subs become too difficult due to the high number of orders per sub-problem in a single zone. A similar difficulty is encountered in instance 1Z-3L-500, where the average optimality gap of LTLP-C-Subs solved is 89%. On the other hand, in this specific case, the solution cost is reduced significantly in the improvement phase.

Overall, the results indicate that the matheuristic algorithm can produce high quality solutions in significantly lower computation times compared to the exact solution approach.

Figure 4.3 shows how the solution cost found by the matheuristic algorithm for different instance types changes as the instances get larger. We exclude instances 1Z-10L-400 and 1Z-10L-500 from this comparison.

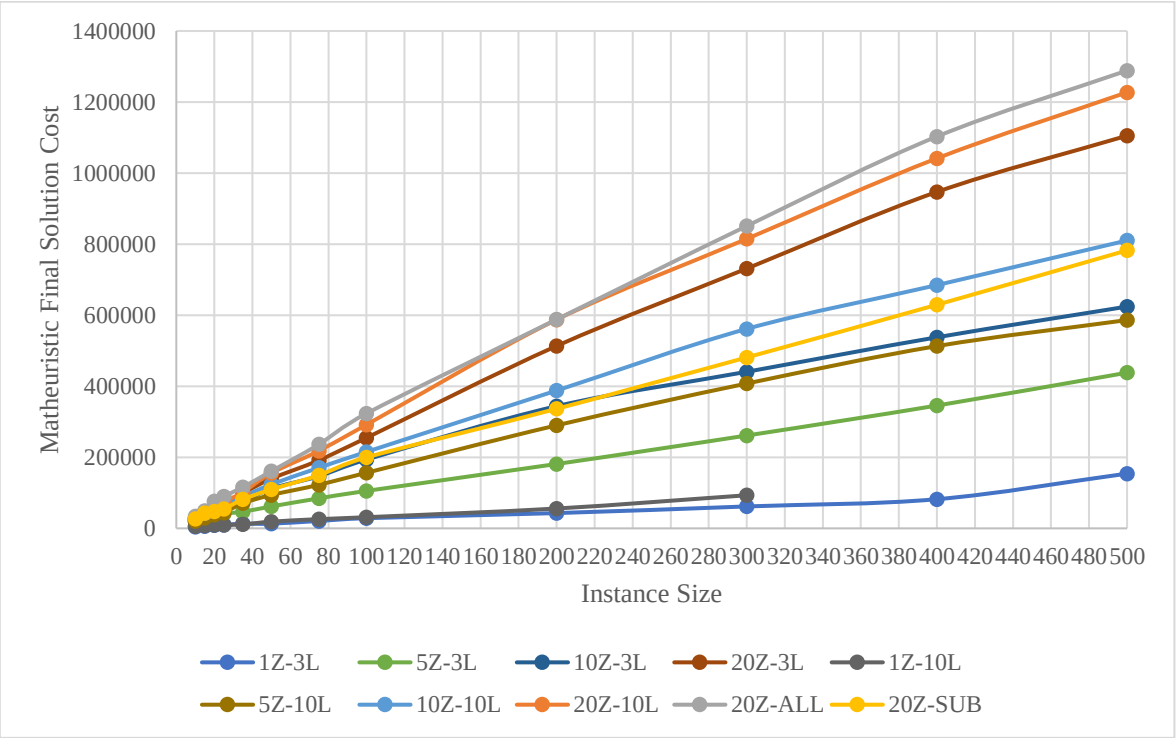


Figure 4.3. Matheuristic Final Solution Cost with Respect to Instance Size.

As expected, the total solution cost is lower when the number of zones in the problem network, and the number of possible pickup and delivery locations within the zones are low. Instance type 20Z-SUB has a moderate cost compared to other instance types. This is a special case where certain zones are significantly more congested than others. In this sense, it is a good representation of the actual operations of the 3PL company.

Comparing the results of the initial solution cost of the matheuristic, which is obtained through the matheuristic construction algorithm against the final solution cost achieved after the improvement phase, we deduce that the improvement matheuristic reduces the cost of the initial solution by 1.8%, on average. On the other hand, the solution time increases drastically when the improvement matheuristic is applied. Therefore, the solutions delivered by the constructive matheuristic algorithm can be acceptable in the case where the allotted time for solving the problem is limited.

The number of LTL orders in the system of the 3PL carrier may reach up to 1,500 on a daily basis, and the company wants to obtain planning results in two hours for the LTL process. Although the solution approach we presented for LTLP-C can be used in practice for small-size carriers to solve instances below 400 orders, an alternative strategy is required for implementing our algorithms for the specific 3PL company. When the results

of LTLP-C was analyzed with the relevant parties at the firm, it was realized that several simplifying assumptions could be done to not only reduce the problem complexity, but also increase the compliance with the practical considerations that are taken into account in the existing system. After conducting a second analysis, we have altered the scope of LTLP and defined a second version of the problem. The next section addresses this new version of LTLP with a new problem definition and solution approach.

4.3 The Less-than Truckload Planning Problem with Discrete Planning Horizon and Multi-period Considerations

In this section, we focus on the new version of LTLP, which we name as the *LTLP with Discrete Planning Horizon and Multi-period Considerations* (LTLP-DM). LTLP-DM differs from LTLP-C in several aspects.

The first main point where LTLP-DM diverges from LTLP-C is the structure of the planning horizon. LTLP-C assumes a continuous planning horizon, which may seem to be a more realistic representation of the actual system. However, it has been observed that the actual operations often deviate from the planned operations in terms of temporal projections due to unexpected events (e.g. delays) in the system. Therefore, accurately modelling temporal parameters and constraints such as travel times, loading / unloading times and synchronization requirements does not necessarily lead to better results in practice. In LTLP-DM, we switch to a discrete-time concept where all durations are represented in "days". LTL process involves both long-haul and short haul operations. We assume that intra-zone routes and some short-haul inter-zone routes with a distance below a predefined limit are completed on the same day. For long-haul routes, the travel duration in days is defined based on the distance of the route. We use the current method utilized by the planners to convert route distance to route duration.

Since new orders are received every day, planning for multiple days does not create a crucial advantage as plans will be refreshed on a daily basis. While we define LTLP-DM as a daily planning problem for this particular reason, we also take into account the impact of our decisions in the following days and incorporate the future considerations into our model.

Another important observation is that any item that is shipped to a cross-dock terminal is dispatched to its new route at least one they after its arrival at the transfer facility. Therefore, if an order will be transported to its final destination through multiple shipments, the future shipments of it does not necessarily need to be planned at the beginning. Since we produce transportation plans each day, we can make the simplifying assumption that an order can be assigned to at most one route-vehicle pair, which also lifts the synchronization requirements at the cross-dock terminals for the daily problem.

Although LTLP-C features hard deadlines, our new analysis has shown that late deliveries can be tolerated to some degree. We convert the hard deadlines to soft due dates, and our motivation is to keep the percentage of late deliveries below 2% in general.

Two matheuristic approaches are proposed for solving LTLP-DM. In the first method, called the *Decentralized Heuristic* (DH), we adopt the philosophy of the existing planning approach but optimize the related decisions in order to have a benchmark environment to test the performance of the new centralized solution methodology. We implement an algorithm that establishes a decomposition-based framework which solves a series of sub-problems, each representing an actual problem faced by the individual dispatchers of the 3PL firm on a daily basis. By following the existing planning approach and yet optimizing it according to the practical implications, not only we provide an automated methodology, but also we find a lower bound on the solution total cost that can be achieved without changing the planning strategy.

The *Centralized Heuristic* (CH) adds a certain degree of centralization to the solution approach, while aiming not to deviate from the current practice significantly and creates more consolidation opportunities without violating the practical requirements of the company. Both heuristics are decomposition-based methods that partition LTLP-DM into solvable sub-problems (LTLP-DM-Subs). We develop an integer programming (ILP) model associated with each LTLP-DM-Sub. A column generation-based heuristic is executed for LTLP-DM-Subs that cannot be solved optimally within a specified time limit. We test both approaches by using the real data of the 3PL company.

4.3.1 Problem Definition

We address a pickup and delivery problem with soft due dates and multi-echelon cross-docking opportunities in a multi-period environment. Our objective is to minimize the

total freight cost while keeping the rate of late deliveries as low as possible in the long run. LTLP-DM aims to produce daily route plans and make order fulfillment decisions by considering penalties for decisions that may render future plans inefficient.

Optimizing the daily LTL transportation operations does not guarantee that the system is optimized in the long-term, since the firm has the flexibility to transfer transportation orders to the subsequent days. We develop a multi-day simulation framework which solves planning problems day by day over a long period by simultaneously producing daily transportation plans and deciding on which orders to be passed on to the next day. The overall quality of any solution methodology can be evaluated based on the total freight cost and on-time delivery performance achieved in the long-run by running the simulation.

The flow of orders in a multi-day planning environment is illustrated in Figure 4.4. For any day n, the set of newly received orders is denoted by N^n. The set of orders which are not assigned to any route-vehicle pair on day n is represented by U^n. Note that these orders are transferred to day $n+1$. Any order that is assigned to a route-vehicle pair that will ship it to its final destination leaves the system. However, if an order is shipped to a cross-dock terminal, it will require at least one more transport in the future. Such orders are assumed to be ready for their next shipment after a lead time of t days, where t is an approximation of the sum of the travel time from the pickup terminal to the corresponding cross-dock terminal and the unloading time. $T^{n,n+t}$ is the set of orders inherited from the orders planned on day n, which are assigned to some route-vehicle pair, will be unloaded at cross-dock terminals and become available for their next shipment after t days. In the specific problem we address, $t \in \{1, 2, 3\}$ and its value is estimated for each order based on the distance between its pickup location and the cross-dock terminal it will be delivered to.

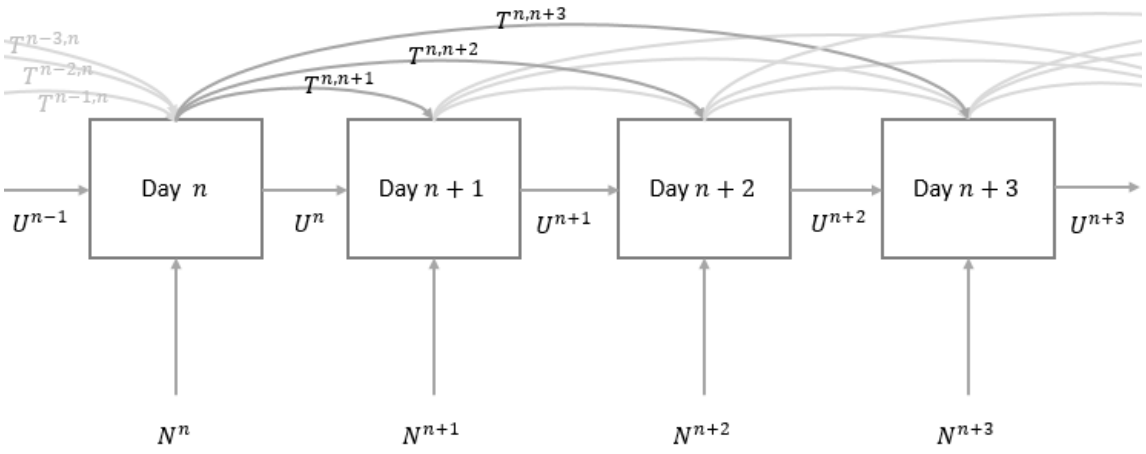


Figure 4.4. Order Flow in Multi-day Planning.

Since the planning is done on a daily basis, we need to explicitly define the LTLP-DM, which is going to be solved daily and propose a solution approach that aims to minimize the total freight cost in the long term. Next, we present a mathematical programming formulation for the LTLP-DM. We use a similar notation to the one used in LTL-C in the ILP model.

In the LTLP-DM, the set of all orders that can be planned on that day is denoted by I. All admissible routes that can be used with the given set of orders is enumerated using the algorithm presented in Section 4.1. The routes are matched with each vehicle type and route-vehicle type pairs are obtained. Each pair is duplicated based on the total potential volume that can be carried along the route associated with it and the vehicle capacity. This establishes the set of route-vehicle pairs P, where each $p \in P$ refers to the combination of a certain route and an individual vehicle of a certain type. Route-vehicle pair p has freight cost C_p^F and the capacity of the vehicle associated with it is denoted by Q_p. Recall that some route types may require a minimum vehicle capacity utilization so that they can be used in the transportation plan. The minimum capacity utilization for route-vehicle pair P is represented by M_p in the formulation of the LTLP-DM. The size of any order $i \in I$ is shown by S_i. We define a hypothetical cost of not assigning an order i to any route-vehicle pair and denote it by C_i^U. The binary parameter O_{ip} has the value of one, if order i can be assigned to route-vehicle pair p, and zero, otherwise. Note that an order can be assigned to a route-vehicle pair if the order's pickup location and delivery location (or a proper cross-dock terminal) is visited along the route associated with the

route-vehicle pair. If the order is going to be shipped to a cross-dock terminal, we know that there will be at least one extra shipment in the following days for delivering it to its final destination. In order to account for this, we define an artificial cost for orders-to-be-transferred, denoted by C_{ip}^{T} for order i and route-vehicle pair p. C_{ip}^{T} is equal to zero if order i is shipped to its final destination by route-vehicle pair p, and it has a positive value, otherwise. The binary variable y_p takes the value of one, if route-vehicle pair p is used in the daily transportation plan. If order i is assigned to route-vehicle pair p, the value of the binary variable x_{ip} will be set to one. In the case where order i is unassigned, the binary variable u_i will take the value of one. The integer linear programming (ILP) model associated with LTLP-DM is provided as follows:

$$Minimize \quad \sum_{p \in P} C_p^F y_p + \sum_{i \in I} C_i^U u_i + \sum_{i \in I} \sum_{\substack{p \in P \\ O_{ip}=1}} C_{ip}^T x_{ip} \tag{4.39}$$

$$subject\ to$$

$$\sum_{\substack{p \in P \\ O_{ip}=1}} x_{ip} + u_i = 1 \qquad i \in I \tag{4.40}$$

$$\sum_{\substack{i \in I \\ O_{ip}=1}} S_i x_{ip} \leq Q_p y_p \qquad p \in P \tag{4.41}$$

$$\sum_{\substack{i \in I \\ O_{ip}=1}} S_i x_{ip} \geq M_p Q_p y_p \qquad p \in P \tag{4.42}$$

$$x_{ip} \in \{0,1\} \qquad i \in I, p \in P, O_{ip} = 1 \tag{4.43}$$

$$u_i \in \{0,1\} \qquad i \in I \tag{4.44}$$

$$y_p \in \{0,1\} \qquad p \in P \tag{4.45}$$

The objective function of LTLP-DM consists of the total freight cost, cost of unassigned orders and cost of orders-to-be-transferred. Constraint (4.40) ensures that an order is either assigned to one route-vehicle pair, or it is unassigned. Constraint (4.41) guarantees that the total size of orders assigned to a route-vehicle pair cannot exceed the capacity of the vehicle linked with it. A route-vehicle pair can be used only if the

minimum capacity utilization can be attained. This is ensured via constraint (4.42). The domains of the decision variables are established by constraints (4.43), (4.44) and (4.45).

As we did LTLP-C model, we introduce symmetry breaking constraints to LTLP-DM in a similar fashion. We assign indices to route-vehicle pairs from 1 to $|P|$ and we define a binary parameter Y_{p_1,p_2} which is equal to one if route-vehicle pairs p_1 and p_2 have the same route, their vehicle types are the same and index of p_1 is smaller than that of p_2, and zero, otherwise. Then we have the constraint $y_{p_1} \geq y_{p_2}$ (7) where $Y_{p_1,p_2} = 1$.

The choice of unassigned order cost (C_i^U) and cost of order-to-be-transferred (C_{ip}^T) in the LTLP-DM is very important for the quality of solutions in the long term. The unassigned order cost, C_i^U is modeled as a function of: i) $C_i^{F_{max}}$, defined as the maximum freight cost (including cross-docking) for shipping order i to its final destination, ii) D_i, namely, the number of days until the latest departure date for order i, and iii) S_i, that is, the size of order i. Q^{max} stands for the capacity of the largest vehicle. The weight of the cost is controlled by the parameter λ_U. Equation (4.46) illustrates the computation of C_i^U.

$$C_i^U = \lambda_U \times C_i^{F_{max}} \times \frac{1}{(D_i + 1)} \times \left(1 + \frac{S_i}{Q^{max}}\right) \qquad (4.46)$$

C_i^U is inversely proportional with the number of days until the latest departure, which means that the cost increases as the due date for delivery gets closer. There is no hard constraint in LTLP-DM for ensuring that each order is delivered before its due date. However, one can easily impose this by setting λ_U to a high value.

The cost associated with an order-to-be-transferred (C_{ip}^T) should represent the marginal cost that may incur due to the subsequent shipments of order i. When order i is unloaded at a cross-dock terminal, we assume that a new order i' is created. The origin of i' is the cross-dock terminal to which order i is shipped by route-vehicle pair p, whereas its destination is the same as that of order i. $C_{i'}^{F_{max}}$ denotes the maximum freight cost for shipping order i' to its final destination. C_{ip}^T is defined as a function of $C_{i'}^{F_{max}}$ and order size (S_i), where λ_T is a weight parameter used to adjust the value of C_{ip}^T.

$$C_{ip}^T = \lambda_T \times C_{i'}^{F_{max}} \times \left(1 + \frac{S_i}{Q^{max}}\right) \qquad (4.47)$$

In the next section, we propose two decomposition-based heuristic approaches for solving the LTLP-DM.

4.3.2 Heuristics for Solving the LTLP-DM

One of the main concerns of this study is to develop a fast solution approach for LTLP-DM so that applicable and high quality solutions are found for practical instances within a reasonable amount of time. We can see that constructing the ILP for LTLP-DM and solving it exactly would require a substantial computation time based on our experiments with small-sized instances. Therefore, we focus on heuristic solution methods to generate daily LTL transportation plans quickly. The first approach we consider is an imitation of the current decentralized way of planning, in which we aim to solve the planning problems faced by the dispatchers of the 3PL firm separately. The results obtained by this method will serve a benchmark given that the current decentralized planning approach does not change. Afterwards, we propose a methodology to solve the problem by adding a certain degree of centralization to the matheuristic. We demonstrate the potential improvement in the LTL transportation costs by transitioning to a centralized planning approach, together with the benefits of optimization, in our simulation results.

Decentralized Heuristic

The *Decentralized Heuristic* (DH) is designed to solve LTLP-DM by decomposing it into tractable parts such that each sub-problem represents the actual daily planning task of a dispatcher working for the 3PL company. Any sub-problem of LTLP-DM, which is referred to as LTLP-DM-Sub, has the same structure as that of LTLP-DM, but is solved for a subset of orders to be planned on a specific day. DH embodies the current way of sharing information between dispatchers located at pickup depots and cross-dock terminals. Hence, it aims to solve LTLP-DM through a similar approach to the existing operations planning strategy of the firm. The notation related with DH and steps of the algorithm are provided as follows.

On any day n, we are given a set of orders (I) to be planned, which consists of the newly arrived orders (N^n), unassigned orders from the previous day (U^{n-1}), and orders

associated with the items unloaded at the cross-dock terminals before and ready to be shipped ($\bigcup_{j=n-3}^{n-1} T^{j,n}$). L^P, L^D and L^C denote the set of pickup locations, delivery locations and cross-dock terminals, while L stands for the set of all locations in the network ($L = L^P \cup L^D \cup L^C$). Each location belongs to a certain zone. The set of all locations in zone $z \in Z$ is represented by L_Z, where $L_Z = L_z^P \cup L_z^D \cup L_z^C$ and the set of pickup locations, delivery locations and cross-dock terminals in zone z are shown by L_z^P, L_z^D and L_z^C, respectively. In the current network structure of the 3PL company, each zone accommodates a single cross-dock facility. Therefore, $|L_z^C| = 1$ for $z \in Z$. $I_{S_1}^O \subseteq I$ represents the subset of orders whose origin location is in set S_1, where $S_1 \subseteq (L^P \cup L^C)$. Similarly, $I_{S_2}^D \subseteq I$ is the set of orders which have a destination location within S_2, where $S_2 \subseteq (L^D \cup L^C)$. The set of routes generated for any given set of orders I^* is shown by R_{I^*}. Solving LTLP-DM-Sub at day n for I^* and R_{I^*} produces three main outcomes: $K_{I^*}^n$, which consists of pairs of activated route-vehicle pairs and the set of orders assigned to them; $U_{I^*}^n$, the set of unassigned orders and $T_{I^*,c}^{n,j}$, the set of orders created based on the ones that will be shipped to cross-docking facility c and will be available for their next transport on day $j \in B^n$. B^n is the set of future days on which the newly created orders can be available for their next shipment. In the case of our specific problem, B^n is simply $\{n+1, n+2, n+3\}$. $K_{I^*}^n$ is named as the set of combinations, and each element of $K_{I^*}^n$ is in the form of (p, I_p^A) where p is a route-vehicle pair and $I_p^A \subseteq I^*$ is the set of orders assigned to p.

Algorithm 4.10. Decentralized Heuristic (DH)

1: *Let* $I := N^n \cup U^{n-1} \cup \left(\bigcup_{j=n-3}^{n-1} T^{j,n}\right)$, $K^n := \emptyset$, $U^n := \emptyset$, $T^{n,j} := \emptyset$,

2: *For all pickup locations* ($l \in L^P$)*:*

3: *Let* $I^* := I_{\{l\}}^O$

4: *Construct the set of admissible routes* R_{I^*} *such that:*

- *Generated routes are of type 1 (collection routes), 3 (pickup and delivery routes), 4 (pickup and delivery routes with single transfer) or 6 (trunk route with pickup)*
- *The only pickup location in the routes is* l

5: *Solve* LTLP-DM-Sub *with the order set,* I^* *and the set of routes,* R_{I^*}
Obtain $K_{I^*}^n$, $U_{I^*}^n$ *and* $T_{I^*,c}^{n,j}$ *for* $c \in L^C$ *and* $j \in B^n$

6: $K^n := K^n \cup K_{I^*}^n$

$T^{n,j} := T^{n,j} \cup T^{n,j}_{I^*,c}$ *for* $c \in L^C$ *and* $j \in B^n$

7: *For all zones* ($z \in Z$)*:*

8: *Let* $I^* := \left(\bigcup_{l \in L^P_z} U^n_{I^O_{\{l\}}}\right) \cap I^O_{L^P_z}$

9: *Construct the set of admissible routes* R_{I^*} *such that:*

- *Generated routes are intra-zone collection routes (of type 1)*
- *All routes terminate at the cross-dock terminal* $c \in L^C_z$

10: *Solve LTLP-DM-Sub with the order set,* I^* *and the set of routes,* R_{I^*}

Obtain $K^n_{I^*}$, $U^n_{I^*}$ *and* $T^{n,j}_{I^*,c}$ *for* $c \in L^C_z$ *and* $j \in B^n$

11: $K^n := K^n \cup K^n_{I^*}$

$T^{n,j} := T^{n,j} \cup T^{n,j}_{I^*,c}$ *for* $c \in L^C_z$ *and* $j \in B^n$

$U^n := U^n \cup U^n_{I^*}$

12: *Let* $I^* := I^O_{\{c\}}$ *where* $c \in L^C_z$

13: *Construct the set of admissible routes* R_{I^*} *such that:*

- *Generated routes are of type 2 (distribution routes), 5 (pickup and delivery routes), 5 (trunk route) or 7 (trunk route with delivery)*
- *All routes start from the cross-dock terminal* $c \in L^C_z$

14: *Solve LTLP-DM-Sub with the order set,* I^* *and the set of routes,* R_{I^*}

Obtain K_{I^*}, U_{I^*} *and* $T^{n,j}_{I^*,c}$ *for* $c \in L^C$ *and* $j \in B^n$

15: $K^n := K^n \cup K^n_{I^*}$

$T^{n,j} := T^{n,j} \cup T^{n,j}_{I^*,c}$ *for* $c \in L^C$ *and* $j \in B^n$

$U^n := U^n \cup U^n_{I^*}$

16: *Return* K^n, U^n *and* $T^{n,j}$ *for* $j \in B^n$

We remind that LTLP-DM needs to be solved each day within a multi-day planning environment. Step 1 of the DH forms the set of orders to be planned at a specific day (n) and initializes the set of combinations, unassigned orders and orders-to-be-transferred. Steps 3 through 6 represent the solution of the planning problem faced by dispatchers located at pickup terminals. Each dispatcher deals with the orders that have to be shipped from his/her facility and the type of routes used in the plans include a single pickup location. Therefore, each problem is defined as a separate LTLP-DM-Sub and solved independently. Information regarding the unassigned orders originating at each pickup location is transmitted to the cross-dock terminals. Dispatchers at the cross-dock terminals gather such orders from different pickup locations in their zones and plan the inbound shipments employing intra-zone collection routes. This process is demonstrated in steps 8 through 11 of DH. Afterwards, the planning of outbound shipments for each

cross-dock terminal is carried out via steps from 12 to 15. Inbound and outbound transportation planning problems encountered by the dispatchers at each cross-dock terminal can be modelled separately. Thus, each such problem corresponds to a distinct LTLP-DM-Sub for DH. The outcome of DH at day n is the set of combinations (K^n), set of unassigned orders (U^n) and the set of orders-to-be-transferred ($T^{n,j}$). In a multi-day planning setting, all orders in U^n will be transferred to day $n+1$. $T^{n,j}$ represents the set of new orders created based on the orders that will be shipped to cross-dock terminals via route-vehicle pairs planned on day n. Orders in $T^{n,j}$ are scheduled to arrive at the system on day j for $j \in B^n$. These orders will have the cross-dock terminal which they are shipped to as the origin location, whereas their destination location will be the same as that of the order they are inherited from.

The main advantage of DH is that it can be easily put into practice without changing the existing planning approach of the 3PL firm drastically. However, the highly decentralized structure leads to missed consolidation opportunities, and hence, loss of potential savings in costs. An advantage of DH is that steps 3 to 6 and 8 to 15 can be parallelized since each LTLP-DM-Sub can be solved independently. This has the potential to reduce the time required for solving LTLP-DM further with a proper implementation. Next, we propose the Centralized Heuristic (CH), which aims to add a certain degree of centralization to DH.

Centralized Heuristic

The optimal solution to LTLP-DM can be obtained by solving the associated ILP model. However, due to the complex nature of the problem and the need for a quick solution, solving ILP model to optimality is not possible in practice. The Centralized heuristic (CH) is designed to solve LTLP-DM in a more integrated manner compared to DH, by enhancing the consolidation opportunities in order to achieve lower freight costs. Similar to DH, CH also employs a decomposition approach and partitions the set of daily orders to solve LTLP-DM-Sub with a smaller set of orders and less number of route-vehicle pairs. However, the partitioning logic differs from that of DH. Instead of constructing sub-problems that are analogous to daily planning tasks of the dispatchers, we divide the set of orders based on their origin and destination zones. Let (z_1, z_2) represent a pair of

origin and destination zones where $z_1, z_2 \in Z$. We next form clusters. Any subset of orders that have their origin locations in zone z_1 and destination locations in zone z_2 are in the same cluster. In the first phase of the CH (steps 3 to 6 of Algorithm 2 given below), we solve LTLP-DM-Sub for each cluster separately. If the origin and destination zones for any given cluster are the same ($z_1 = z_2$), then intra-zone pickup and delivery routes (with or without transfer) are used in LTLP-DM-Subs, whereas all types of inter-zone route types are incorporated if the origin zone is different from the destination zone. The main reason for partitioning the orders based on their origin and destination zones is that the current types of routes employed by the 3PL firm are also associated with an origin zone, where all items are picked up from, and a destination zone, where all items are delivered to. This route structure allows decomposing the problem by keeping most of the consolidation opportunities available.

The orders which are not assigned to any route-vehicle pair in the first phase are re-clustered based on their origin zones. In steps 8 through 11 of Algorithm 2, separate LTLP-DM-Sub are solved for each of the new order clusters with intra-zone collection routes. This phase aims to bring items that belong to orders having the same origin zone but not necessarily the same destination zone, to the closest cross-dock terminal. The last phase of the algorithm is designed to plan the last mile deliveries for the items which have already been shipped to the cross-dock terminals at their destination zones. This requires the last mile delivery orders to be clustered based on their destination zones. LTLP-DM-Sub is then solved for each cluster with intra-zone distribution routes. CH allows parallelization as well. In steps 3 to 6 and 8 to 15, LTLP-DM-Subs can be solved in parallel since each sub-problem accommodates a set of orders that is independent from the others.

The pseudocode of the CH, which uses the same notation as that of DH, is provided below.

Algorithm 4.11. Centralized Heuristic (CH)

1: *Let* $I := N^n \cup U^{n-1} \cup \left(\bigcup_{j=n-3}^{n-1} T^{j,n}\right)$, $K^n := \emptyset$, $U^n := \emptyset$, $T^{n,j} := \emptyset$,

2: *For all origin-destination zone pairs* (z_1, z_2) *for* $z_1, z_2 \in Z$*:*

3: *Let* $I^* := I^O_{\left(L^P_{z_1} \cup L^C_{z_1}\right)} \cap I^D_{L^D_{z_2}}$

4: *If* $z_1 = z_2$*:*

Enumerate all admissible intra-zone routes of type 3 (pickup and delivery routes) and 4 (pickup and delivery routes with single transfer) for the given set of orders I^* and construct the set of routes, R_{I^*}

Else:

Enumerate all admissible inter-zone routes (of type 1 through 8) for the given set of orders I^* and construct the set of routes, R_{I^*}

5: *Solve LTLP-DM-Sub with the order set, I^* and the set of routes, R_{I^*}.*
Obtain $K^n_{I^*}$, $U^n_{I^*}$ and $T^{n,j}_{I^*,c}$ for $c \in L^C$ and $j \in B^n$

6: $K^n := K^n \cup K^n_{I^*}$
$T^{n,j} := T^{n,j} \cup T^{n,j}_{I^*,c}$ *for* $c \in L^C$ *and* $j \in B^n$

7: *For all zones ($z \in Z$):*

8: *Let* $I^* := \left(\bigcup_{l \in L^P_z} U^n_{I^O_{\{l\}}}\right) \cap I^O_{L^P_z}$

9: *Construct the set of admissible routes R_{I^*} such that:*
- *Generated routes are intra-zone collection routes (of type 1)*
- *All routes terminate at the cross-dock terminal $c \in L^C_z$*

10: *Solve LTLP-DM-Sub with the order set, I^* and the set of routes, R_{I^*}.*
Obtain $K^n_{I^*}$, $U^n_{I^*}$ and $T^{n,j}_{I^*,c}$ for $c \in L^C_z$ and $j \in B^n$

11: $K^n := K^n \cup K^n_{I^*}$
$T^{n,j} := T^{n,j} \cup T^{n,j}_{I^*,c}$ *for* $c \in L^C_z$ *and* $j \in B^n$
$U^n := U^n \cup U^n_{I^*}$

12: *Let* $I^* := I^O_{\{c\}} \cap I^D_{L^D_z}$

13: *Construct the set of admissible routes R_{I^*} such that:*
- *Generated routes are intra-zone distribution routes (of type 2)*
- *All routes start from the cross-dock terminal $c \in L^C_z$*

14: *Solve LTLP-DM-Sub with the order set, I^* and the set of routes, R_{I^*}.*
Obtain K_{I^*} and U_{I^*}

15: $K^n := K^n \cup K^n_{I^*}$
$U^n := U^n \cup U^n_{I^*}$

16: *Return K^n, U^n and $T^{n,j}$ for $j \in B^n$*

After the orders are partitioned into subsets and LTLP-DM-Subs are defined, the set of relevant routes should be constructed through the algorithm presented in Section 4.1. The next section describes the solution approach for tackling LTLP-DM-Subs.

4.3.3 A Matheuristic for Solving the LTLP-DM-Sub

In both DH and CH, the LTLP-DM is decomposed into sub-problems called LTLP-DM-Subs in order to reduce the computational effort and solution time. Preliminary experiments have shown that majority of the LTLP-DM-Subs instances obtained from practical size problems can be solved optimally by formulating each problem as an integer linear programming (ILP) solving it through commercial solvers. Although most of the instances can be solved to optimality within an acceptable duration, some instances become intractable due to the high number of orders and routes. The main reason for the different levels of difficulty between instances is the imbalanced structure of the network in the tested real-life instances. Density of orders are significantly higher in some regions compared to others due to non-uniform spread of the population and industrialized areas across the country. Furthermore, the LTLP-DM-Sub instances solved within CH are more difficult compared to those solved within DH as a result of larger order clusters and higher consolidation opportunities.

In order that both easy and difficult problem instances are handled properly within a reasonable solution time, we propose a generic solution methodology that combines an exact mathematical programming-based approach with heuristics. This type of approach is referred to as a "matheuristic" in the literature. The matheuristic (MH) for LTLP-DM-Sub starts by solving the ILP model for the LTLP-DM-Sub with a time limit (τ_{SUB}). The optimality gap of the solution found is denoted by G_{SUB}. The algorithm terminates if the optimal solution to LTLP-DM-Sub is achieved ($G_{SUB} = 0$) within the specified time limit. Otherwise, a column generation-based heuristic (CGH) is executed to obtain a near-optimal solution to LTLP-DM-Sub. The initial columns of CGH are acquired from the best feasible solution to LTLP-DM-Sub, which is found by the solver when the time limit is exceeded. If no feasible solution is found or all orders are unassigned in the feasible solution, then a constructive heuristic is run to generate the set of initial columns for CGH. This heuristic is called the *Potential Insertion Heuristic* (PIH), which selects route-vehicle pairs of higher potential and assigns orders to them in a greedy manner iteratively. The overall flow of the MH algorithm is given as follows.

Algorithm 4.12. A Matheuristic for Solving LTLP-DM-Sub

1: *Get algorithm inputs: I and R_I*
2: *Construct the set of route-vehicle pairs P, using I and R_I*
3: *Solve the ILP for the LTLP-DM-Sub for the order set I and route-vehicle pair set P with a time limit of τ_{SUB} minutes, obtain K_I, U_I, and G_{SUB}*
4: *If $G_{SUB} = 0$:*
5: *Terminate the algorithm and return K_I and U_I*
6: *Else:*
7: *If $|K_I| > 0$:*
8: *Let K_I and U_I form the initial set of columns for CGH*
9: *Else:*
10: *Run PIH and obtain new sets of K_I and U_I*
Let K_I and U_I form the initial set of columns for CGH
11: *Run CGH and obtain sets of activated combinations (K_I^*) and unassigned orders (U_I^*)*
12: *Return K_I^* and U_I^**

Once the set of route-vehicle pairs (P) is constructed, the ILP for LTLP-DM-Sub is solved with a time limit of τ_{SUB} minutes. Recall that the ILP has three types of binary decision variables: y_p indicates whether route-vehicle pair p is used, u_i shows whether order i is unassigned and finally x_{ip} takes the value of one if order i is assigned to route-vehicle pair p. Let the solution returned by the solver be denoted by y_p^* for $p \in P$, u_i^* for $i \in I$ and x_{ip}^* for $i \in I$, $p \in P$ and $O_{ip} = 1$, where O_{ip} shows whether order i can be assigned to route-vehicle pair p. The set of unassigned orders, U_I is simply a subset of I where $u_i^* = 1$. The set of activated combinations (K_I) is the collection of used route-vehicle pairs and orders assigned to them. Each element of K_I is in the form of $\left(p, I_p^A\right)$ for $p \in P$ such that $y_p^* = 1$ and I_p^A is a subset of I where $x_{ip}^* = 1$.

If the solver finds the optimal solution, which means the optimality gap (G_{SUB}) is equal to zero, then the solution found (K_I, U_I) is returned. Otherwise, CGH is executed to find a good solution heuristically. An initial set of columns is necessary for running CGH. Each column is a combination $\left(p, I_p^A\right)$ that contains a route-vehicle pair (p), and a set of orders assigned to it (I_p^A). If the solver finds a feasible solution with a positive number of combinations ($|K_I| > 0$), then these combinations are used as initial columns of CGH. If

the solver fails to detect a feasible solution or the set of combinations it produces is empty, then PIH is run to generate initial columns for CGH.

PIH is a constructive heuristic which aims to create feasible combinations by activating route-vehicle pairs with a high potential of load carrying and assigning orders to them in a randomized and greedy fashion. Let S_p^{potp} denote the potential of route-vehicle pair p and Q_p^* be the total size of loads assigned to p. Throughout the algorithm, we maintain a set of candidate route-vehicle pairs $P^* \subseteq P$. For each route-vehicle pair $p \in P^*$, we have $I_p^{P^*}$ which represents the set of orders which can potentially be assigned to p. The pseudocode of PIH is given in Algorithm 4.13.

Algorithm 4.13. Potential Insertion Heuristic

1: *Get algorithm inputs: I and P*
2: *Let $P^* := P$, $U_I := I$ and $K_I := \emptyset$. Calculate S_p^{potp} and let $Q_p^* := 0$ for $p \in P^*$*
3: *Remove any route-vehicle pair p from P^* if $S_p^{potp} < Q_p M_p$*
4: *While $|P^*| > 0$:*
5: *Pick a route-vehicle pair, p' from P^* by roulette-wheel selection with probabilities proportional to the potentials S_p^{potp}*
6: *Let $I_{p'}^A := \emptyset$. Construct $I_{p'}^{P^*}$ evaluating $i \in U_I$ such that $O_{ip} = 1$*
7: *While $\left|I_{p'}^{P^*}\right| > 0$:*
8: *Pick and order, i' from $I_{p'}^{P^*}$ by roulette-wheel selection with probabilities proportional to the order sizes S_i*
9: *If $S_{i'} \leq Q_{p'} - Q_{p'}^*$:*
10: *Append i' to $I_{p'}^A$ and let $Q_{p'}^* := Q_{p'}^* + S_{i'}$*
11: *Remove i' from $I_{p'}^{P^*}$*
12: *If $Q_{p'}^* \geq Q_{p'} M_{p'}$:*
13: *Append the combination $\left(p', I_{p'}^A\right)$ to K_I and let $U_I := U_I \setminus I_{p'}^A$*
14: *Remove p' from P^*.*
15: *Update potentials of route-vehicle pairs (S_p^{potp}) and remove any route-vehicle pair p from P^* if $S_p^{potp} < Q_p M_p$*
16: *Return K_I*

PIH employs roulette-wheel selection mechanism for determining: i) the route-vehicle pair to be activated (step 5) and ii) orders that will be assigned to the activated route-vehicle pair (step 8). A higher potential increases the probability of selection for any

route-vehicle pair whereas a bigger order size increases the chances of an order getting assigned to the selected route-vehicle pair. It is ensured that the vehicle capacity is not exceeded when an order is assigned to a route-vehicle pair. Any route-vehicle pair p that has a potential (S_p^{potp}) below the required capacity usage for satisfying the minimum utilization limit ($Q_p M_p$) is disregarded. Any combination created $\left(p', I_{p'}^A\right)$ is added to the set of generated combinations (K_I) if the minimum utilization limit is met. These inspections guarantee that only feasible combinations are detected by PIH.

Once the set of initial columns are constructed either via exploiting the best feasible solution returned by the solver or running PIH, the column generation procedure can be launched. We describe CGH in the next section in detail.

4.3.4 Column Generation Heuristic

Column generation is a technique for solving linear programming problems with huge number of variables in an efficient manner. We propose a heuristic method that utilizes this technique for finding a good solution to LTLP-DM-Sub within a short time frame. The *Column Generation Heuristic* (CGH) starts with an initial set of combinations and iteratively generates new combinations to improve the best feasible solution to LTLP-DM-Sub. We reformulate the LTLP-DM-Sub to incorporate variables associated with combinations rather than route-vehicle pairs. The new formulation, called the *Integer Master Problem* (IMP), is theoretically capable of solving any LTLP-DM-Sub to optimality given that the set of all feasible combinations are available. However, for problem sizes in practice, it is computationally impossible to generate all feasible combinations for LTLP-DM-Subs. CGH is designed to find the best feasible solution to IMP using the limited solution time effectively.

IMP can be also solved for a restricted set of columns, in which case it is called the *Restricted Integer Master Problem* (RIMP). To model an instance of RIMP, we need to have a set of orders (I) and set of feasible combinations (K). C_k^K is the cost associated with combination k. If $k = \left(p, I_p^A\right)$ is the combination of route-vehicle pair p and the set of orders assigned to it I_p^A, then we have:

$$C_k^K = C_p^F y_p + \sum_{i \in I_p^A} C_{ip}^T x_{ip} \qquad (4.48)$$

A_{ik} is a binary parameter which is equal to one if $i \in I_p^A$ where $k = (p, I_p^A)$ and zero, otherwise. We define the binary variable z_k to indicate whether combination k is activated or not. The binary variable u_i, and the unassigned order cost coefficient C_i^U are analogous to the same components in the formulation of ILP formulation of LTLP-DM-Sub. The resulting mathematical programming formulation of RIMP is as follows:

$$Minimize \quad \sum_{k \in K} C_k^K z_k + \sum_{i \in I} C_i^U u_i \qquad (4.49)$$

$$subject\ to$$

$$\sum_{k \in K} A_{ik} z_k + u_i = 1, \qquad i \in I \qquad (4.50)$$

$$z_k \in \{0, 1\}, \qquad k \in K \qquad (4.51)$$

$$u_i \in \{0, 1\}, \qquad i \in I \qquad (4.52)$$

The objective function is the summation of the cost of activated combinations and unassigned orders. Constraint (4.50) guarantees that an order is either unassigned or it is used in exactly one of the activated combinations. Constraints (4.51) and (4.52) define variables z_k and u_i as binary.

In the column generation procedure, we solve the linear relaxation of RIMP, which is referred to as the *Restricted Linear Master Problem* (RLMP), and obtain dual prices associated with each order. The mathematical model of RLMP can be demonstrated as:

$$Minimize \quad \sum_{k \in K} C_k^K z_k + \sum_{i \in I} C_i^U u_i \qquad (4.49)$$

$$subject\ to$$

$$\sum_{k \in K} A_{ik} z_k + u_i = 1, \qquad i \in I \qquad (4.50)$$

$$z_k \geq 0, \qquad k \in K \qquad (4.53)$$

$$u_i \geq 0, \qquad i \in I \qquad (4.54)$$

Using the dual prices obtained by solving RLMP, one can compute the reduced cost coefficient (C_k^R) for any combination (k). When combinations with negative reduced costs are incorporated into RLMP and the model is solved again, a new set of dual prices is obtained. This process can be repeated until no further combinations with negative reduced cost can be detected. At the end, the given instance of RLMP is solved optimally.

We denote the dual prices associated with each constraint in the constraint set (12) as μ_i for $i \in I$. The reduced cost of any combination $k = (p, I_p^A)$ is then calculated as:

$$C_k^R = C_p^F + \sum_{i \in I_p^A} (C_{ip}^T - \mu_i) \qquad (4.55)$$

The problem of finding combinations with negative reduced costs is called as the *Pricing Problem*. Both exact and heuristic approaches can be employed for determining which combinations will be introduced to RLMP in the next iteration. Deciding on how many columns will be added to the RLMP is an important algorithmic choice. It is possible to add one or more combinations with the least reduced costs, or even all the combinations with negative reduced costs at each iteration of the column generation procedure. We first formulate the pricing problem as an integer programming (IP) model, called the *Pricing Model* (PM), which aims to detect δ_{PM} most negative combinations based on a given set of orders (I), route-vehicle pairs (P) and the vector of dual prices with elements μ_i for $i \in I$. The PM shares several common constraints with the ILP model of LTLP-DM-Sub and it is presented as follows:

$$Minimize \quad \sum_{p \in P} C_p^F y_p + \sum_{i \in I} \sum_{\substack{p \in P \\ O_{ip}=1}} (C_{ip}^T - \mu_i) x_{ip} \qquad (4.56)$$

$$subject\ to$$

$$\sum_{p \in P} y_p \leq \delta_{PM}, \qquad p \in P \qquad (4.57)$$

$$\sum_{\substack{i \in I \\ O_{ip}=1}} S_i x_{ip} \leq Q_p y_p, \qquad p \in P \qquad (4.41)$$

$$\sum_{\substack{i \in I \\ O_{ip}=1}} S_i x_{ip} \geq M_p Q_p y_p, \qquad p \in P \qquad (4.42)$$

$$x_{ip} \in \{0,1\}, \qquad i \in I, p \in P, O_{ip} = 1 \qquad (4.43)$$

$$y_p \in \{0,1\}, \qquad p \in P \qquad (4.44)$$

The objective function of PM represents the summation of the reduced costs of all selected combinations. Constraint (4.57) ensures that at most δ_{PM} combinations are selected. One can simply lift constraint (4.57) to select all combinations with negative reduced costs. Constraints (4.41) and (4.42) are adopted from the ILP formulation of LTLP-DM-Sub, which guarantee the feasibility of the selected combinations in terms of not exceeding vehicle capacity and satisfying minimum capacity utilization limit.

Solving PM with the complete set of route-vehicle pairs might result in high computation times since the pricing problem will be solved many times within the column generation process. To avoid this, we propose a heuristic approach called the *Pricing Heuristic* (PH), to produce combinations with negative reduced costs efficiently. PH consists of four main steps. Firstly, a set of promising route-vehicle pairs ($\hat{P}$) is identified by picking the route-vehicle pairs with the least reduced cost potential. The maximum number of elements in set $\hat{P}$ is a predefined parameter denoted by δ_{PP}. For each $p \in \hat{P}$, a modified version of the potential insertion heuristic (MPIH) is executed to construct the set of combinations, K^*. If at least one combination has a negative reduced cost, then the algorithm terminates and the set of all negative cost combinations is returned. Otherwise, PM is solved for $\hat{P}$ with $\delta_{PM} = 1$ to identify the combination with the most negative reduced cost using only the set of promising route-vehicle pairs. If this step fails to find a combination with negative reduced cost, then PM is solved for the complete set of route-vehicle pairs (P) with $\delta_{PM} = 1$ as the last resort. This approach has two main advantages: i) by applying quicker methods first, combinations with negative reduced costs can be detected faster in most of the cases, and ii) if the efficient methods fail to find such combinations, the pricing problem is solved exactly so that the column generation process

is not aborted unnecessarily due to lack of new combinations. The overall flow of PH is as given in Algorithm 4.14.

Algorithm 4.14. Pricing Heuristic

1: *Get algorithm inputs: I, P and μ_i for $i \in I$*
2: *Identify the set of promising route-vehicle pairs ($\hat{P}$)*
3: *Run MPIH with $\hat{P}$ and obtain K^**
4: *Remove combinations with non-negative reduced costs from K^**
5: *If $|K^*| > 0$:*
6: *Return K^**
7: *Else:*
8: *Solve PM for route-vehicle pair set $\hat{P}$ and $\delta_{PM} = 1$, obtain K^**
9: *If $|K^*| > 0$:*
10: *Return K^**
11: *Else:*
12: *Solve PM for route-vehicle pair set P and $\delta_{PM} = 1$, obtain K^**
13: *Return K^**

The identification of promising route-vehicle pairs (step 2) is a randomized process to pick δ_{PP} route-vehicle pairs which are more likely to lead to combinations with negative reduced costs. Assume that I_p^P represents the subset of orders which can possibly be assigned to a route-vehicle pair p. For each $p \in P$ we calculate the potential metric E_p^{potp}, where

$$E_p^{potp} = C_p^F + C_{ip}^T - \frac{Q_p}{S_p^{potp}} \sum_{i \in I_p^P} \mu_i \qquad (4.58)$$

The overall process is demonstrated in Algorithm 4.15.

Algorithm 4.15. Identification of Promising Route-Vehicle Pairs

1: *Get algorithm inputs: I, P and μ_i for $i \in I$*
2: *If $|P| \leq \delta_{PP}$:*
3: *Let $\hat{P} := P$ and return $\hat{P}$*
4: *Else:*
5: *Let $\hat{P} := \emptyset$, $P^{cand} := P$ and calculate E_p^{potp} for each $p \in P^{cand}$*
6: *While $|\hat{P}| < \delta_{PP}$:*

7: *Pick a route-vehicle pair p' from P^{cand} by roulette-wheel selection with probabilities proportional to values of E_p^{potp}*
8: *Append p' to $\hat{P}$ and remove p' from P^{cand}*
9: *Update probabilities based on the remaining elements in P^{cand}*
10: *Return $\hat{P}$*

Modified Potential Insertion Heuristic (MPIH) aims to construct a feasible combination for each promising route-vehicle pair $p \in \hat{P}$. The mechanism for assigning orders to route-vehicle pairs is very similar to that in PIH, but instead of picking orders with probabilities proportional to order sizes (S_i), we use dual prices (μ_i).

Algorithm 4.16. Modified Potential Insertion Heuristic

1: *Get algorithm inputs: I, $\hat{P}$ and μ_i for $i \in I$*
2: *Let $K^* := \emptyset$. Calculate S_p^{potp} and let $Q_p^* := 0$ for $p \in \hat{P}$*
3: *Remove any route-vehicle pair p from $\hat{P}$ if $S_p^{potp} < Q_p M_p$*
4: *For each $p \in \hat{P}$:*
5: *Let $I_p^A := \emptyset$ and $I_p^{P^*} := I_p^P$*
6: *While $|I_p^{P^*}| > 0$:*
7: *Pick and order, i' from $I_{p'}^{P^*}$ by roulette-wheel selection with probabilities proportional to the dual prices μ_i*
8: *If $S_{i'} \leq Q_p - Q_p^*$:*
9: *Append i' to I_p^A and let $Q_p^* := Q_p^* + S_{i'}$*
10: *Remove i' from $I_p^{P^*}$*
11: *If $Q_p^* \geq Q_p M_p$:*
12: *Append the combination (p, I_p^A) to K^**
13: *Return K_I*

The iterative column generation process aims to solve the linear relaxation of the problem (LMP) optimally. Since the solution of LMP may contain fractional variables, a branch-and-price algorithm may be utilized to solve the IMP. However, preliminary experiments have shown that this approach consumes too much time for solving LTLP-DM-Subs. Instead, the *Column Generation Heuristic* (CGH) runs the column generation procedure for a predefined time limit τ_{CG} and solves the RIMP with the set of generated columns to obtain an integral solution to the given LTLP-DM-Sub. In our preliminary experiments, almost all of the practical instances of RIMP are solved optimally in a very

short computation time. Still, we specify a time limit τ_{RIMP} for the solver to account for larger instances that may occur in the future. The pseudocode of CGH is provided in Algorithm 4.17.

Algorithm 4.17. Column Generation Heuristic

1: *Get algorithm inputs: I, P and the set of initial columns, K_I*
2: *While elapsed time < τ_{CG}:*
3: *Solve RLMP for I and K_I. Obtain dual prices μ_i for $i \in I$*
4: *Run PH and obtain the set of generated combinations, K^**
5: *If $|K^*| > 0$:*
6: *Let $K_I := K_I \cup K^*$*
7: *Else, go to Step 8*
8: *Solve RIMP for I and K_I with a time limit of τ_{RIMP} minutes*
Obtain the set of activated combinations, K_I^* and the set of unassigned orders U_I^*
9: *Return K_I^* and U_I^**

CGH is an important component of the matheuristic algorithm that aims to solve the difficult instances of sub-problems within both decentralized and centralized solution approaches. In the next section we compare the performances of these two approaches through a comprehensive computational study and interpret the results from quality and efficiency perspectives.

4.3.5 Computational Experiments for the LTLP-DM

In this section, we present the results of our computational experiments performed on real order data of the 3PL company. The same data set is used for decentralized and centralized heuristics (DH and CH). A multi-day simulation framework is developed to observe the performance of both approaches in the long run. We simulate each day by solving daily planning problems and maintain flow balance for orders between days. First, we present the characteristics of the test data. Then, we compare the results of DH and CH for a one-month experiment period. Finally, we investigate how the overall results would change for different values of cost coefficients associated with unassigned orders and orders-to-be-transferred.

Description of the Test Case

The experiments are run on the real orders of the 3PL company that are collected for a full month. A total of 26,663 orders that account for half a million cubic meters of goods are included in the tests. Figure 4.5 illustrates both the daily number of orders and the total size of the orders received at each day of the experiment period. It is seen that the daily number of orders gets as high as 1400. The number of orders received in certain days are significantly lower compared to others. This is because only a few clients of the 3PL company work on Sundays. Our heuristics are designed to solve instances with thousands of orders. Therefore, a comparison of the heuristics with optimal solutions in very small size instances is not meaningful.

While running both DH and CH, the simulation algorithm manages the set of orders that will be planned in each day's LTLP-DM. On day one, it is assumed that no orders are transferred from the previous periods and the LTLP-DM is solved for the orders received on the first day only. After the last day (day 31) is planned, there can still be some unassigned orders remaining or orders scheduled to arrive in the following days due to transfers. We run the simulation for an extra week or until all orders are delivered to their final destinations.

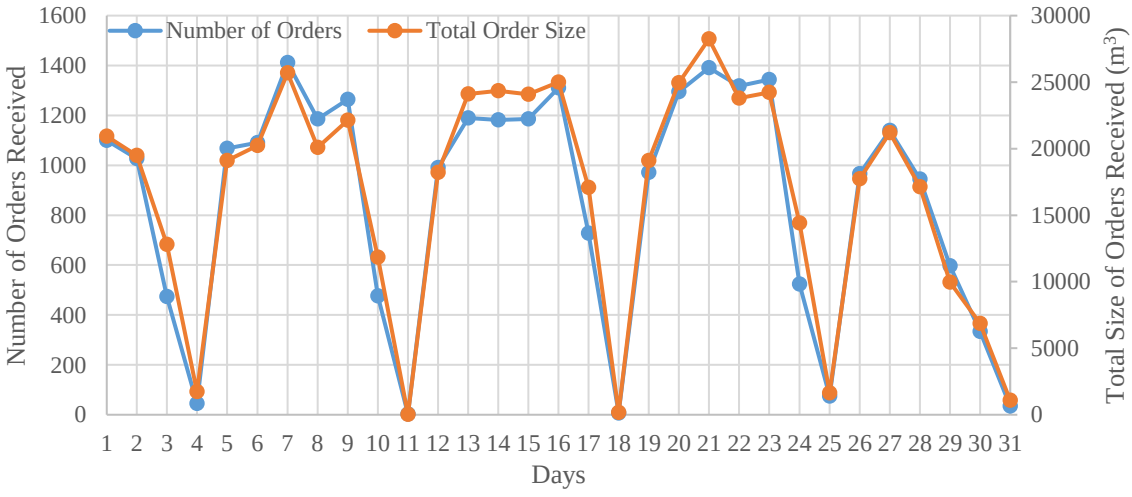


Figure 4.5. Total Quantity and Size of the Orders Received per Day.

The operation area is divided into 20 zones. Due to non-uniform distribution of the population and industrial areas across the country, some zones have a significantly higher frequency of orders compared to others. Figure 4.6 demonstrates the average number of orders per day from each zone (displayed in rows) to each zone (displayed in columns).

One can easily deduce from the figure that zone 13 is by far the busiest region in terms of both inbound and outbound shipments, followed by zones 14 and 11.

Two types of trucks with capacities of 50 and 90 cubic meters are available. For the sake of simplicity, hereafter we refer to these as small and large trucks. Each route-vehicle pair is associated with a fixed cost, which depends on the type of the truck as well as the origin and destination zones of the route. For any given route, fixed cost of a large vehicle is 1.25 times that of a small vehicle. Variable costs depend on the route distance and the number of visits to cross-dock terminals. Table 4.4 summarizes the values of parameters that belong to vehicle types.

Table 4.4. Vehicle Parameters.

Vehicle Type	***Capacity (m^3)***	***Cross-docking Cost (₺)***	***Fuel Cost (₺ / km)***
Small	50	150	2.5
Large	90	200	3

To Zone

From Zone	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	3.5	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.1	4.0	0.7	0.0	0.6	0.2	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.2	0.0	0.6	0.1	0.0	0.0	0.0	0.2	0.0	0.1	0.1	0.0	4.1	0.5	0.1	0.2	0.1	0.0	0.0	0.2
4	0.0	0.0	0.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.3	0.2	0.0	0.0	0.5	0.1	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.5	0.4	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.0	0.1	0.0	0.0	0.5	0.1	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.1	0.0	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	1.2	0.2	0.0	0.0	0.0	0.0	0.0	0.0
11	1.0	0.0	2.1	0.8	0.5	1.1	0.0	0.0	0.1	0.1	0.1	0.1	3.8	3.4	3.2	0.5	0.4	1.1	0.8	0.6
12	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8	0.7	0.2	0.0	0.0	0.0	0.0	0.0	0.0
13	45.9	3.2	96.7	35.9	11.3	37.8	6.4	8.1	12.6	16.2	16.8	26.3	109.0	119.9	77.7	19.2	10.5	15.0	24.6	11.1
14	6.4	0.4	8.4	4.6	2.4	4.9	1.2	5.5	3.5	5.6	1.9	5.1	12.8	11.8	7.6	3.0	3.0	1.1	4.3	2.7
15	0.0	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	4.2	0.2	0.5	0.0	0.0	0.1	0.0	0.0
16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.0	0.0	0.0	1.0	0.1	0.0	0.1	0.3	0.0	0.0	0.0
17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.7	0.1	0.1	0.0	0.0	0.0	0.3	0.1
20	0.2	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.2	0.1	0.0	0.0	0.0	0.0	0.1	0.0

Figure 4.6. Average Daily Number of Orders between Zones.

Recall that we identified eight main types of routes that can be used in the operations of the 3PL company. For the route generation process, we need to define the vector β_h for each route type h. We have two sets of route types: H^{intra} and H^{inter}, which represent intra-zonal and inter-zonal route types, respectively. The values of the parameters for route types in H^{intra} and H^{inter} are provided in Tables 4.5 and 4.6. Note that each specified route type can be used in DH, CH or both heuristics, depending on the general structure of the algorithm. The information regarding the solution approaches in which the given route types are used is also provided in the tables below.

Table 4.5. Intra-zonal Route Types and Their Parameters.

Route Type	***Used in***	$\beta^{minpick}$	$\beta^{maxpick}$	β^{mindel}	β^{maxdel}	$\beta^{xdorigin}$	β^{xddest}	$\beta^{maxspdev}$	$\beta^{minutil}$
1.1	CH & DH	1	5	0	0	True	False	0.25	0
2.1	CH & DH	0	0	1	5	True	False	0.25	0
3.1	DH	1	1	1	3	False	False	0.1	0.9
3.2	CH	1	3	1	3	False	False	0.1	0.9

Table 4.6. Inter-zonal Route Types and Their Parameters.

Route Type	***Used in***	$\beta^{minpick}$	$\beta^{maxpick}$	β^{mindel}	β^{maxdel}	$\beta^{xdorigin}$	β^{xddest}	$\beta^{maxspdev}$	$\beta^{minutil}$
1.2	DH	1	1	0	0	False	True	0.1	0.9
1.3	CH	1	3	0	0	False	True	0.1	0.9
2.2	CH & DH	0	0	1	3	True	False	0.1	0.9
3.3	DH	1	1	1	3	False	False	0.1	0.9
3.4	CH	1	3	1	3	False	False	0.1	0.9
4.1	CH & DH	1	2	1	3	True	False	0.1	0.9
4.2	CH	1	3	1	2	False	True	0.1	0.9
5.1	CH & DH	0	0	0	0	True	True	0.1	0
6.1	CH & DH	1	2	0	0	True	True	0.1	0.9
7.1	CH & DH	0	0	1	2	True	True	0.1	0.9
8.1	CH & DH	1	2	1	2	True	True	0.1	0.9

Two important coefficients, λ_U and λ_T are used to determine the cost of an unassigned order and the cost of delivering an order to a cross-dock terminal for a future shipment to its final destination. Both coefficients are set to one in the initial scenario, but further experiments are performed to observe the effect of higher and lower values of λ_U and λ_T.

Time limits set for solving the mathematical models and running the heuristics are critical parameters that affect both solution time and quality. We specify a time limit of 3 minutes (180 seconds) for solving the ILP model for each LTLP-DM-Sub (τ_{SUB}) and

solving RIMP at the end of CGH (τ_{RIMP}). The maximum time for running the column generation process within CGH (τ_{CG}) is also set to 3 minutes.

Lastly, the value of the maximum number of promising route-vehicle pairs (δ_{PP}) to be used in the pricing heuristic (PH) is set to 10.

Each parameter value is determined after a series of preliminary experiments to obtain the best possible solutions in general, ensuring that the overall solution time for each day allows the practical usage of the algorithms developed.

Comparison of Decentralized and Centralized Heuristics

We perform our experiments using a simulation framework that utilizes DH or CH to solve LTLP-DMs and organizes the flow of order information across the days within the planning window. We run two replications of a full-month simulation applying DH and CH each day. The simulation framework and the heuristics are coded in Python 3 and all mathematical programming models are solved by CPLEX 12.8 on an 8-core PC with 3.20 GHz processors and 16 GB RAM. We first compare the results of two simulation runs in terms of several performance metrics that give an overall idea about the differences between the decentralized and centralized approaches. Then we compare the quality of the solutions both on a day-by-day basis and cumulatively. Finally, we interpret the solution times and optimality gaps to evaluate the efficiency of our methods. Daily detailed simulation results for both DH and CH are reported in tabular form in Appendix D.

Table 4.7. A General Comparison of DH and CH.

Comparison Metric	***Decentralized Heuristic (DH)***	***Centralized Heuristic (CH)***
Number of trips	9,886	8,668
Vehicle usage by type	Small trucks: 29.1% Large trucks: 70.9%	Small trucks: 23.9% Large trucks: 76.1%
Average vehicle capacity utilization	88.69%	89.38%
Number transfers at cross-dock terminals	8,322	5,386
Average number of orders per trip	3.54	3.70
Average number of trips per order	1.31	1.20

Table 4.7 demonstrates the overall results of the simulations with DH and CH as the means of solving LTLP-DMs. Note that all orders are delivered to their final destinations by the end of day 36 in both simulations. With the enhanced consolidation opportunities, CH is able to deliver the same set of orders with a smaller number of trips using larger

trucks more frequently with a higher capacity utilization. The number of order transfers at the cross-dock terminals is 35% less in CH compared to DH. This figure indicates that direct routes are used more often in CH. Even though each order is shipped by a larger number of trips on average due to the higher number of transfers in DH, the average number of orders accommodated per each trip is still lower than that of CH. This result also demonstrates more efficient usage of trips by CH.

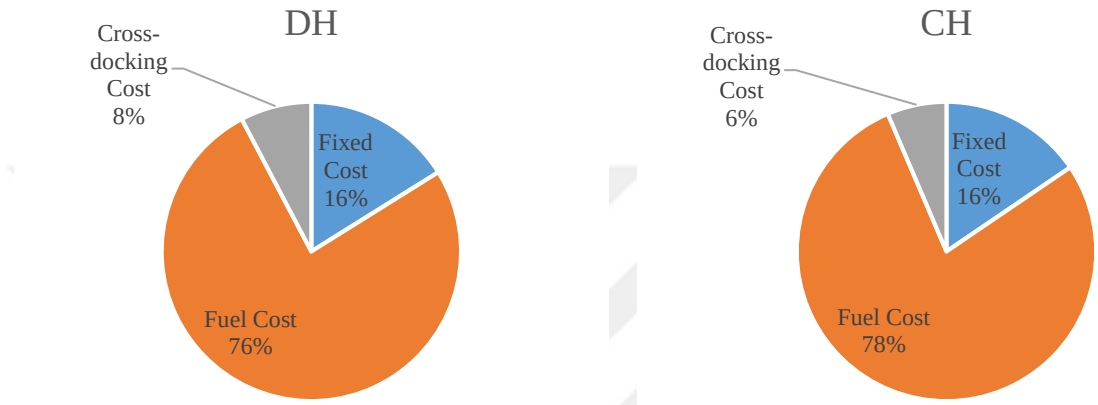


Figure 4.7. Composition of Freight Cost for DH and CH.

Figure 4.7 shows the composition of total freight cost for decentralized and centralized approaches. The portion of cross-docking costs is lower for CH thanks to a lower number of order transfers.

In each instance of LTLP-DM or LTLP-DM-Sub, the objective is to minimize the summation of total freight cost, unassigned order cost, and the cost of orders-to-be-transferred. Assume that we have a solution to an instance of LTLP-DM where P^* is the set of activated route-vehicle pairs, I_p^A is the set of orders assigned to route-vehicle pair p for each $p \in P^*$ and U^* is the set of unassigned orders. Then the total solution cost is calculated as:

$$\sum_{p \in P^*} C_p^F + \sum_{i \in U^*} C_i^U + \sum_{p \in P^*} \sum_{i \in I_p^A} C_{ip}^T \qquad (4.59)$$

The first cost component is the summation of freight costs (C_p^F) for each route-vehicle pair. The other two cost terms (C_i^U and C_{ip}^T) are not real costs that the company has to pay. These are artificial costs that are introduced to provide flexibility of not fulfilling an order

at the day of planning or shipping an order to a cross-dock terminal instead of its final destination. Although the daily objective is to minimize the total cost, in the long run the firm wants to minimize its total freight costs while all orders are delivered on time. We first compare DH and CH in terms of cumulative freight costs and then provide the results related to total costs.

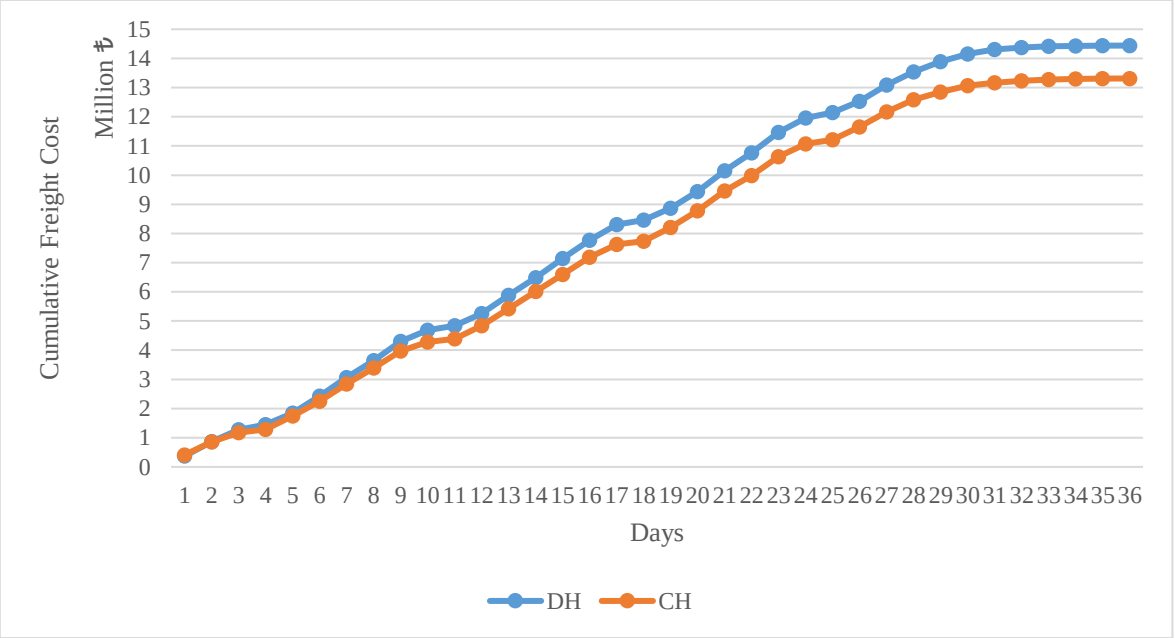


Figure 4.8. Cumulative Freight Costs.

Figure 4.8 illustrates the cumulative freight cost over the days, obtained by running DH and CH each day within the corresponding simulations. As it can be deduced from the graph, the gap between the cumulative freight costs of DH and CH increases as the simulation advances. At the end of day 36, which is the last day of both simulations, cumulative freight cost achieved by employing DH and CH are found to be ₺ 14.4 million and ₺ 13.3 million, respectively. These figures clearly demonstrate that the centralized approach we propose has a potential of creating a saving over ₺ 1 million per month, which accounts for a reduction in the freight cost by 7.8%. Figure 4.9 shows the change in the percentage difference in freight costs between the two approaches over the days.

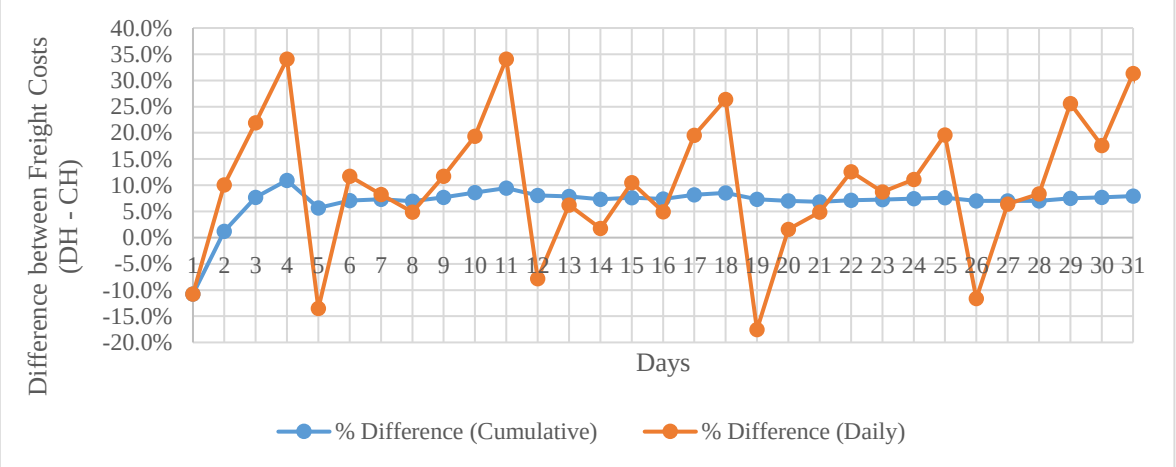


Figure 4.9. Percentage Difference between DH and CH in terms of Freight Cost.

The orange line represents the percentage difference in the freight costs of DH and CH each day, whereas the blue line shows the percentage difference between the cumulative freight costs. DH performs better than CH in terms of the freight costs only on the first day of the simulation and on Mondays (days 1, 5, 12, 19 and 26). This pattern can be explained by two main factors: unassigned orders and trip characteristics. At the beginning of the simulation or on the first day of a new week, portion of orders that can be left unfulfilled is higher since most of the orders are new and their due dates are far enough, which allows orders to be unassigned at relatively lower costs.

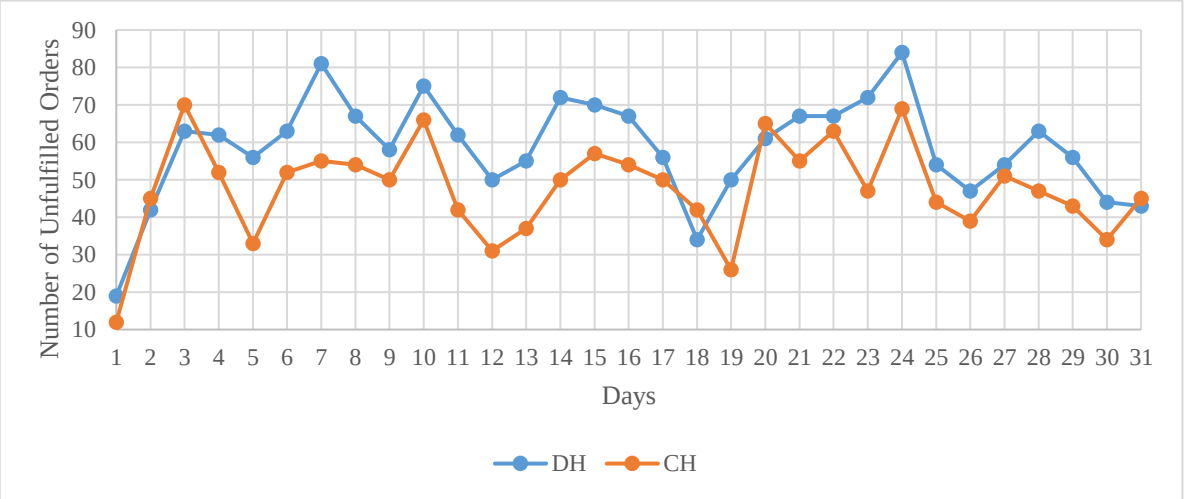


Figure 4.10. Number of Unfulfilled Orders.

Since less consolidation opportunities are present in DH, more orders remain unassigned at these days (see Figure 4.10), which leads to lower freight costs. The other reason that causes this pattern is the type of routes and the number of trips planned on these days. Figure 4.11 shows daily change in the relative difference between DH and CH results in terms of the number of trips planned and the average cost per trip.

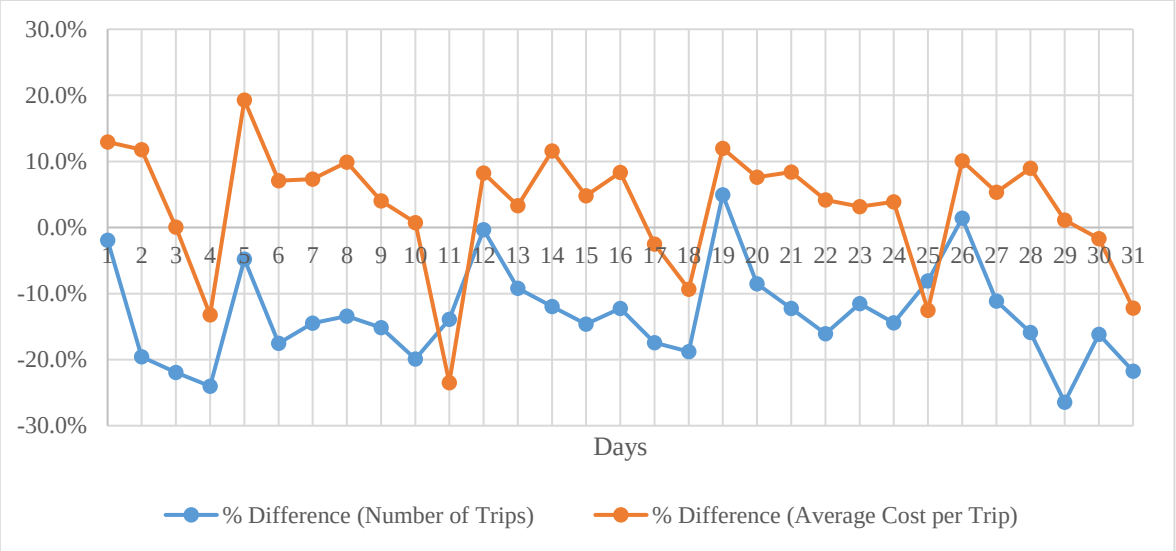


Figure 4.11. Percentage Differences in Number of Trips and Cost per Trip.

On days 1, 5, 12, 19 and 26, the difference in the number of trips is closer to zero unlike the other days. Moreover, trips produced by CH are relatively more costly on these days due to the possibility of employing longer routes with multiple pickup locations. These two factors combined lead to lower freight costs on these specific days. On any other day in the simulations, CH performs better in terms of the freight cost. The relative difference between the cumulative freight costs (see Figure 4.9) remains steady between 7% and 8% as the simulation proceeds, which demonstrates the superiority of CH over DH in the long run.

When we compare the two approaches in terms of total cost, which includes artificial costs in addition to the actual freight cost, it can be clearly seen that there is a significant difference between DH and CH on every single day of the simulation (Figure 4.12). The difference is relatively small on Sundays due to small number of orders available.

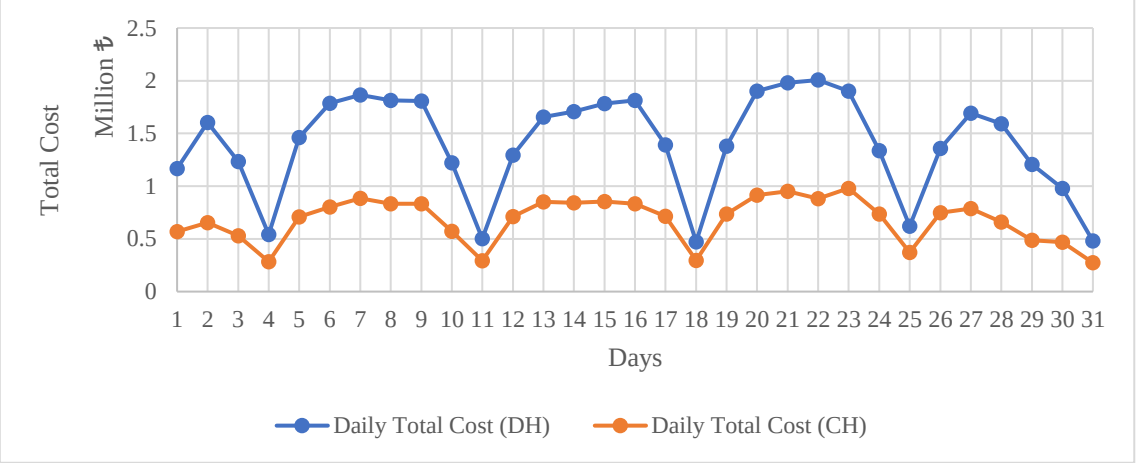


Figure 4.12. Daily Total Costs.

Overall, CH produces 51% lower total cost than DH. We know that 7.8% of this gap comes from the freight cost. The rest of the difference is caused by the higher number of unassigned orders and order transfers at cross-dock terminals in the decentralized approach. Hence, we observe a much better service performance with the CH.

Our next analysis is in terms of our solution methodology. In the description of our solution approaches, we had mentioned that LTLP-DMs are solved by decomposition, where the main problem is partitioned into tractable sub-problems called LTLP-DM-Subs. Figure 4.13 illustrates the number of LTLP-DM-Subs solved on each day by DH and CH.

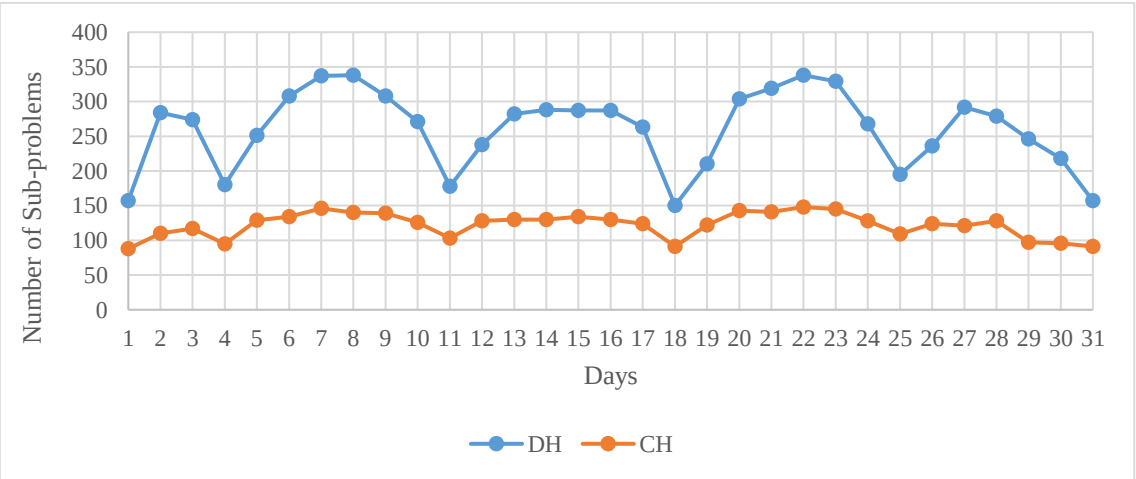


Figure 4.13. Daily Number of LTLP-DM-Subs Solved.

In total 8,230 LTLP-DM-Subs are solved by DH whereas 3,966 LTLP-DM-Subs are solved by CH. This large difference is a result of adding a degree of centralization to the existing solution strategy embodied by DH. Instead of solving a higher number of smaller size LTLP-DM-Subs, CH solves a smaller number of sub-problems for larger order clusters. However, the case is exactly opposite for the solution times.

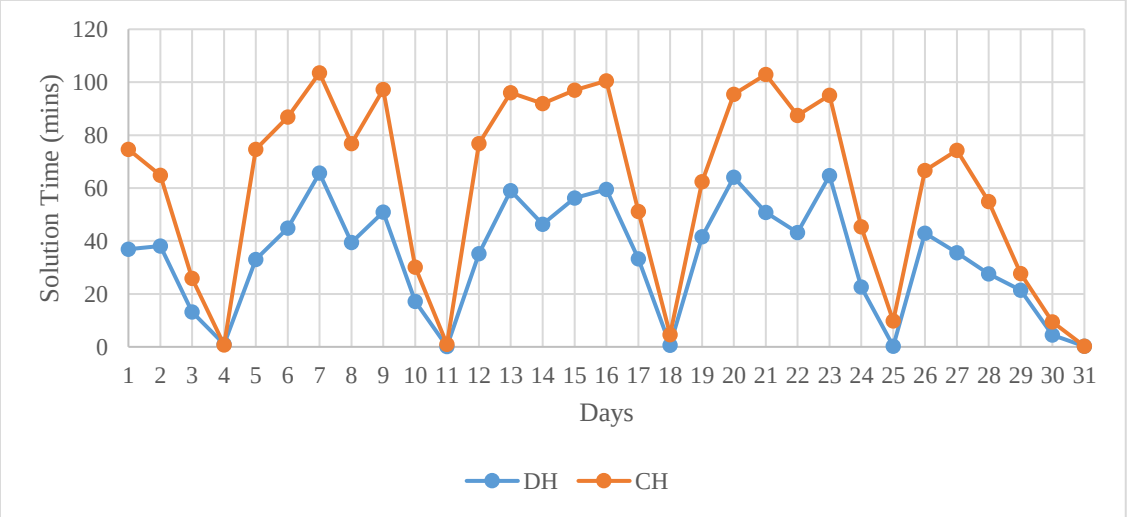


Figure 4.14. Solution Times for LTLP-DMs.

Figure 4.14 represents the total solution times for solving LTLP-DMs on each day. Whereas the number of LTLP-DM-Subs solved by DH is more than twice those solved by CH, the total computation time spent for solving LTLP-DMs by CH is 79.6% higher compared to DH. The average time for solving a LTLP-DM-Sub is 7.7 seconds for DH while it is 28.5 seconds for CH, which explains the reason for the gap between the total solution times of two approaches. Note that the total solution time is below two hours on all days for both approaches, which makes it possible to use them in practice.

The matheuristic that solves LTLP-DM-Sub is composed of two main steps. First, for the given order set the exact ILP model is solved with a short time limit. If the optimal solution to LTLP-DM-Sub is not found, then the Column Generation Heuristic (CGH) is executed. In the exceptional case where ILP solver is not able to find a positive number of initial columns for CGH, then Potential Insertion Heuristic (PIH) is executed to generate them. The total simulation time can be divided into three parts; namely, the time spent for solving the ILP for the LTLP-DM, running CGH and running PIH, since the computation time for all other data processing operations within the simulation are

negligible. Figure 4.15 illustrates the composition of the total simulation time for DH and CH in terms of the times allocated to ILP, PIH and CGH.

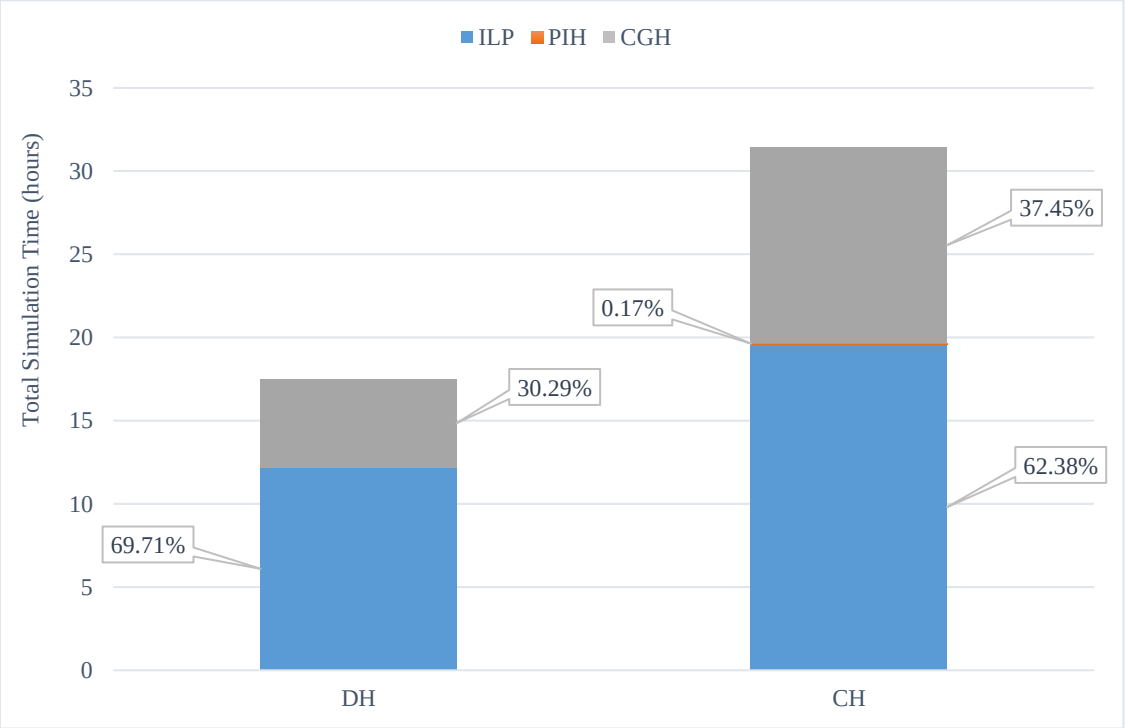


Figure 4.15. Composition of Total Simulation Time

PIH is executed only in CH and it accounts for only 0.17% of the total simulation time. Percentage of time spent for CGH is 30.29% and 37.45% of the total solution times for DH and CH, respectively. The higher portion of CGH in CH is an indication of a lower rate of optimally solved LTLP-DM-Subs compared to DH. We support this argument by investigating the number and percentage of sub-problems solved to optimality by the ILP solver for both approaches: 8,020 out of 8,230 LTLP-DM-Subs (97.4%) are solved optimally by the ILP solver in DH, whereas 3,634 out of 3,966 instances (91.6%) are solved optimally in CH. These figures also imply that CGH is performed for 210 sub-problems in DH, while this number is 332 for CH. The impact of CGH in both approaches is demonstrated in Figure 4.16.

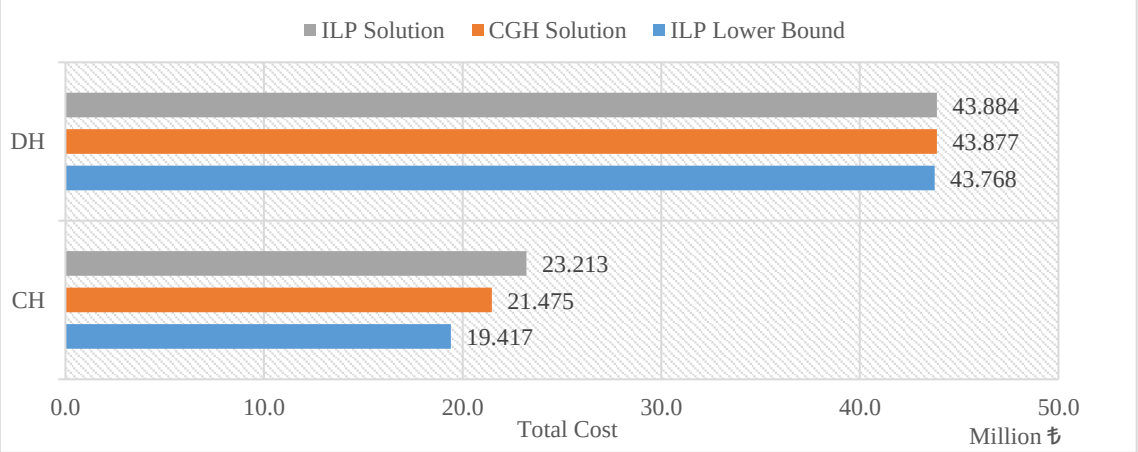


Figure 4.16. Total Solution Costs Found by ILP and CGH with ILP Lower Bounds.

In Figure 4.16, the ILP solution cost values represent the summation of the objective function values of best feasible solutions reported by the solver for each LTLP-DM-Sub. CGH solution cost is the summation of total costs of each LTLP-DM-Sub after the execution of CGH. ILP lower bound is the summation of lower bounds detected by the solver within its time limit for each LTLP-DM-Sub, which theoretically designates the minimum total cost value that can be achieved by solving each LTLP-DM-Sub optimally. The relative difference between ILP solution cost and ILP lower bound is the optimality gap of LTLP-DM in terms of the total cost (G^T_{SUB}), whereas G^T_{CGH} stands for the gap between the CGH solution cost and ILP lower bound. Table 5 summarizes the optimality gaps for the two solution approaches.

Table 4.8. Optimality Gaps of LTLP-DM-Sub and CGH in DH and CH.

	G^T_{SUB}	G^T_{CGH}
DH	0.27%	0.25%
CH	19.55%	10.60%

Since the sub-problems solved in DH are relatively easy, G^T_{SUB} is already quite low and applying CGH has a very small impact (0.02%) on the total solution cost. However, the gap for CH is high at the beginning, and applying CGH brings this gap from 19.55% to 10.6%, which is a significant improvement. Even though CH has an optimality gap of 10.6% in terms of total cost, it is still a more efficient approach as it is 7.8% better than DH in freight costs and 51% better in total cost.

CGH involves an iterative column generation step where dual prices are obtained by solving the Restricted Linear Master Problem (RLMP) and new columns to be introduced to RLMP are discovered through the Pricing Heuristic (PH). PH first tries to detect columns with negative reduced costs via the so-called Modified Potential Insertion Heuristic (MPIH). If it fails to find such columns, the Pricing Model (PM) is solved with the set of promising route-vehicle pairs ($\hat{P}$). If still no columns with negative reduced costs are discovered, then PM is solved with the whole set of route-vehicle pairs for the given LTLP-DM-Sub. Figure 4.17 illustrates the number and portion of PH iterations per type of applied pricing method.

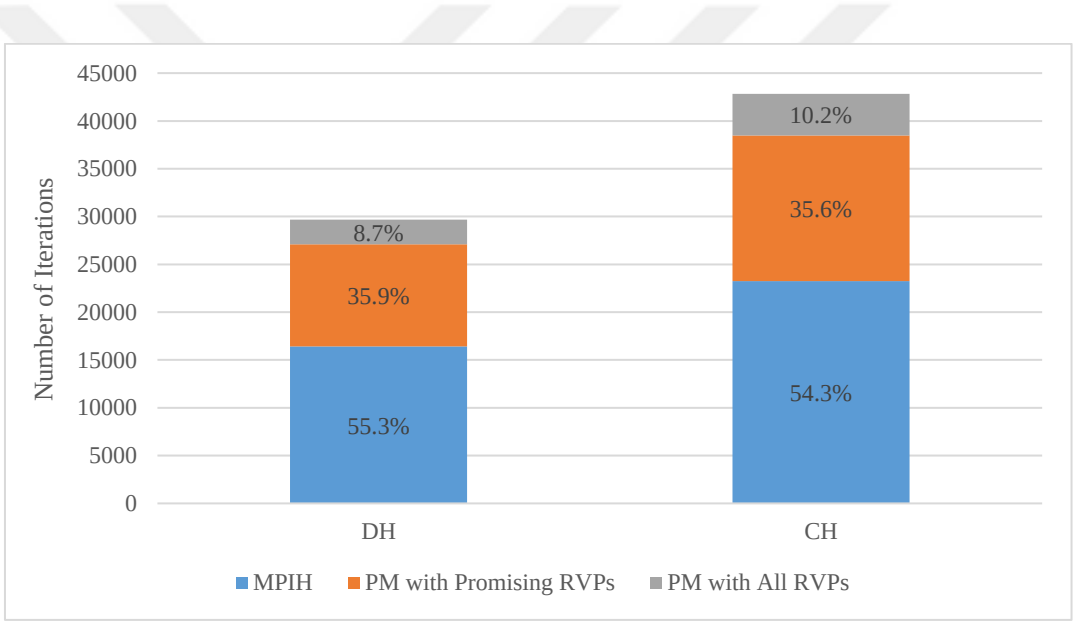


Figure 4.17. Number of PH Iterations per Type.

The heuristic method (MPIH) is applied in more than 50% of the iterations within PH for both decentralized and centralized solution approaches. Although MPIH is used more often compared to the others in terms of the number of iterations, it accounts for only 1.68% of the total time spent for pricing (see Figure 4.18).

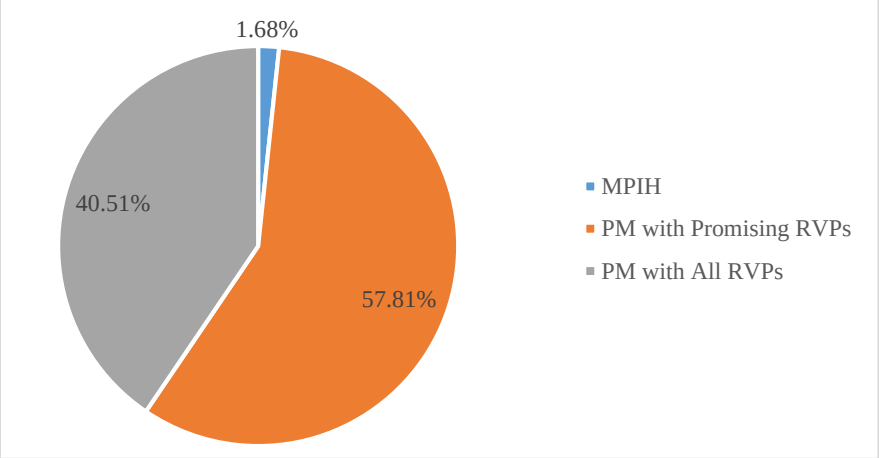


Figure 4.18. Distribution of Total Time Spent for Pricing.

These two figures combined demonstrate the efficiency of MPIH as it is used as the pricing method in more than half of the pricing iterations by consuming only a tiny amount of time.

In the next section we interpret the effect of changing cost coefficients of artificial components on the overall solution quality.

Experiments of Artificial Costs

The objective of the planning problems solved each day is the minimization of a total cost function which is composed of the freight cost and two artificial cost components. The artificial costs are introduced to make our solution approaches work effectively in a multi-day planning environment. The first one is the unassigned order cost (C_i^U), which represents the cost of not assigning an order to any of the planned trips on a single day. The second one is the cost of orders-to-be-transferred (C_{ip}^T), which takes a positive value for the orders which are assigned to trips that will drop them at cross-dock terminals, rather than their final destinations. This cost term can be viewed as an estimation of the cost of shipping this order from the cross-dock facility it is unloaded at to its actual destination in the upcoming days. Both cost components are defined as functions of several different parameters, details of which can be found in Section 4.3.1. For each function, we defined multipliers (λ_U and λ_T) that control the weight of these costs against each other and the freight cost. In the experiment results we have presented up to now,

we set the value of both coefficients to one. In this section, we analyze how the overall solution is impacted when the simulations are run with different values of λ_U and λ_T.

In addition to our base scenario where $\lambda_U = 1$ and $\lambda_T = 1$, we run six new simulations by solving LTLP-DM-Sub through the better performing centralized heuristic (CH) with λ_U values 0.5 and 2, and λ_T values 0.25, 0.5, 2 and 4. We compare the scenario results in terms of performance indicators such as the freight cost, unfulfilled orders and late deliveries. We also analyze several other metrics such as the number of trips, number of transfers and the usage rates of vehicle types to ease the identification of reasons for the differences between scenarios. The overall simulation results for the seven scenarios are tabulated in Table 4.9, where the first row (Scenario 1) represents the base scenario. The day-by-day results of each scenario are provided in Appendix E.

In scenario 2 the unassigned order cost is halved by setting λ_U to 0.5. This results in 244 unfulfilled orders at the end of the simulation. Since unassigned order cost is relatively low, many orders remain unassigned throughout the simulation until a cost efficient shipment opportunity arises. This also leads to late deliveries. Scenario 2 has the highest rate of late deliveries (2.85%) and the highest average lateness (2.43 days) among the orders which are delivered after their designated due dates. The freight cost of scenario 2 is 9.7% lower than that of base scenario thanks to the lower number of trips and higher usage of large trucks. However, the high number of unfulfilled orders might be a major problem for the service quality of the 3PL firm.

Table 4.9. Overall Results of the Scenarios.

Scenario	λ_U	λ_T	*Freight Cost (₺)*	*# of Unfulfilled Orders*	*# of Late Deliveries*	*Late Delivery Rate*	*Average Lateness (Days)*	*# of Trips*	*# of Transfers*	*Vehicle Type Usage Rates (S: Small, L: Large)*	*Average Vehicle Capacity Utilization*
1	**1**	**1**	13,313,734	0	401	1.50%	1.33	8,668	5,386	S: 23.9% L: 76.1%	89.4%
2	**0.5**	**1**	12,026,558	244	759	2.85%	2.43	7,508	4,882	S: 14.3% L: 85.7%	94.7%
3	**2**	**1**	13,917,831	0	452	1.70%	1.35	9,248	5,611	S: 31.8% L: 68.2%	87.3%
4	**1**	**0.25**	13,543,730	0	504	1.89%	1.32	9,217	8,509	S: 20.0% L: 80.0%	90.5%
5	**1**	**0.5**	13,353,188	0	443	1.66%	1.24	8,867	6,623	S: 22.0% L: 78.0%	89.9%

6	**1**	**2**	13,251,581	1	361	1.35%	1.21	8,586	4,889	S: 24.8% L: 75.2%	89.1%
7	**1**	**4**	13,059,407	44	356	1.34%	1.60	8,411	4,370	S: 25.2% L: 74.8%	89.6%

In scenario 3, we double the cost for unassigned orders ($\lambda_U = 2$). This change forces the daily solution algorithm to dispatch orders as quickly as possible. The consequences of this behavior is twofold: firstly, the planned trips are relatively inefficient due to more trips with lower capacity utilization and a larger usage rate of small trucks employed, which leads to a higher freight cost. Secondly, the number of transfers is higher compared to the base scenario, which causes higher lead times for orders to reach their destinations and eventually larger number of late deliveries. The reason for the high number of transfers is that the algorithm prefers to assign orders more frequently to route-vehicle pairs with intra-zonal collection routes, which ship the orders to cross-dock terminals rather than their final delivery locations. Since the minimum utilization limit is zero for this type of routes (see Table 4.5) and the unassigned order cost is high, the algorithm chooses to ship many orders to cross-dock terminals regardless of the vehicle capacity utilization instead of leaving them unassigned with the possibility of a direct shipment on the next day.

In scenarios 4 and 5, cost of order-to-be-transferred (λ_T) is set to 0.25 and 0.5, respectively. This causes an increase in the number of transfers compared to the base scenario. The number of transfers is 8,509 and 6,632 in scenarios 4 and 5, which are 58% and 23% higher than that of the base scenario. More transfers lead to a larger number of orders that are delivered later than their due dates. These two scenarios also produce slightly more costly solutions compared to the base scenario in terms of the freight cost.

In scenarios 6 and 7 we set the multiplier associated with order transfers (λ_T) to 2 and 4, respectively. Results of scenario 6 are promising in the sense that important metrics such as freight costs, number of late deliveries, average lateness and number of transfers are lower than those of the base scenario. However, one of the orders is not fulfilled at the end of the simulation, which might be an indication of a possibility of higher number of unfulfilled orders in other instances. In the last scenario where $\lambda_T = 4$, the number of unfulfilled orders raise up to 44, which makes this scenario even more problematic than scenario 6. Since transfer cost gets too high, it becomes more profitable to leave these

orders unfulfilled than shipping them to cross-dock terminals for transfer. Although scenario 7 produces better results than scenario 6 in most of the performance metrics, the high number of unfulfilled orders makes the results with λ_T equal to 4 unacceptable.

We finally compare the scenarios based on the total simulation time, its composition and the number of LTLP-DM-Subs solved per each scenario. Figure 4.19 demonstrates that in general there is no significant difference among the scenarios in terms of the portions of time spent to each solution step within the matheuristic that solves LTLP-DM. The only noticeable divergence is in the percentage of time spent to Potential Insertion Heuristic (PIH) in scenario 2. Since the unassigned order cost is lowered in this scenario, it is more likely that the ILP model for the LTLP-DM-Sub produces feasible solutions with all orders being unassigned. This causes the algorithm to call PIH for generating initial columns for the next step. The larger number of calls to PIH explains the higher portion of time spent to PIH in scenario 2.

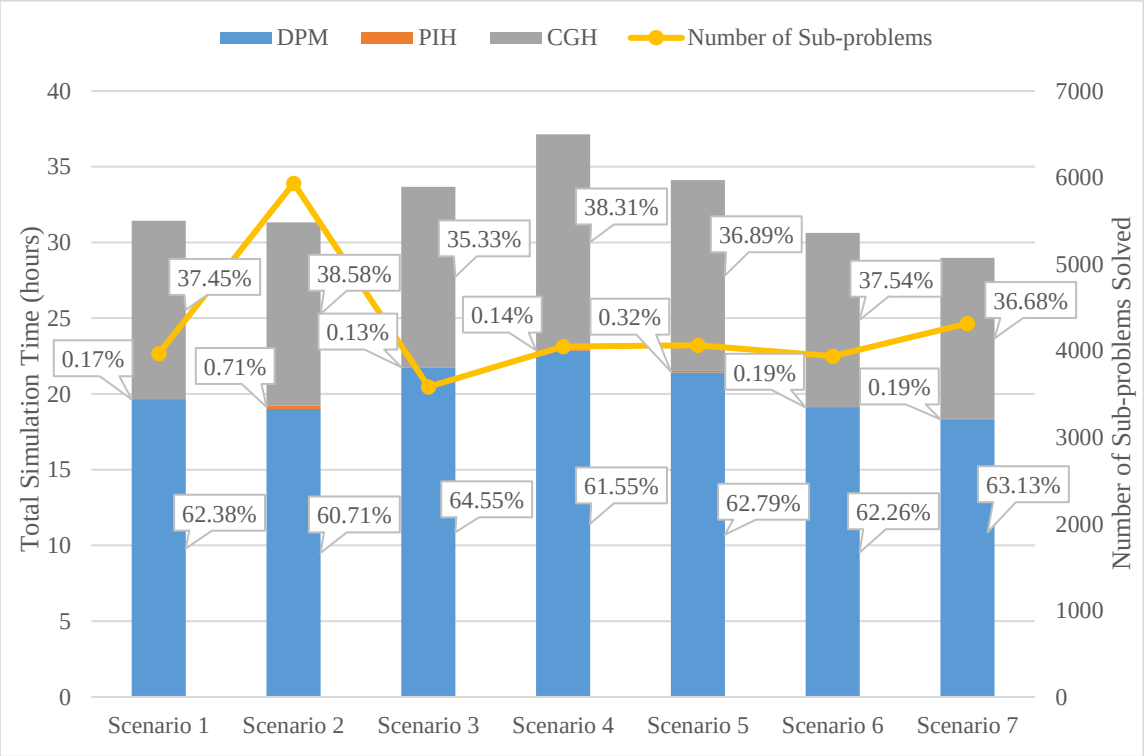


Figure 4.19. Total Simulation Time and Number of Sub-problems Solved.

Scenario 2 also deviates from the rest of the scenarios in terms of the number of LTLP-DM-Subs solved. Since a higher number of unassigned orders are carried over to the

subsequent days, LTLP-DM-Subs associated with these orders are solved repeatedly throughout the simulation which raises the total number of sub-problems solved.

Scenario 4 has the highest total simulation time among all scenarios, despite having a similar number of LTLP-DM-Subs solved as the others. This implies that the solution time per sub-problem is higher for scenario 4 compared to the rest of the scenarios. Since transfers are less costly, a high number of transfers are performed in scenario 4. For each order shipped to a cross-dock terminal for transfer, a new order is created from that terminal to the original order's destination location. Therefore, having a larger number of transfers increases the overall number of orders that need to be dealt with. Solution times eventually become longer due to higher number of orders per LTLP-DM-Sub.

In general, the selection of coefficients in the base scenario seems to be reasonable in terms of the overall indicators of solution quality as well as the solution time. The results suggest that it might be possible to improve on-time delivery performance without producing any unfulfilled orders by setting λ_T to a value greater than 1 and less than 2. The appropriate value of λ_T can be determined via further experimentation by testing alternative values between 1 and 2.

In Chapter 4, we represented two versions of the *LTL Planning Problem* (LTLP). We provided detailed problem definitions and description of solution approaches. The computational results of LTLP-DM applied on the real-data have demonstrated that the freight costs can be reduced by 7.8% by employing the approach we propose in the long run.

The next chapter focuses on the Dedicated Vehicle Assignment Problem (DVAP), in which we aim to assign a limited number of dedicated vehicles to planned LTL routes and FTL orders.

Chapter 5

THE DEDICATED VEHICLE ASSIGNMENT PROBLEM

Logistics companies follow different strategies in terms of the ownership of the vehicles that will perform their transportation operations. While some firms prefer to own the whole vehicle fleet, others may completely outsource the trucking to third-party businesses. Our case represents a mixture of both strategies. The 3PL company we address has a limited number of dedicated vehicles, but it also utilizes trucks from the spot market to fulfill the transportation orders.

Dedicated vehicles can be used in both LTL and FTL operations. While FTL routes are planned by the clients of the 3PL company and the stops on the route are known beforehand, planning of LTL routes is performed by in-house planners. In Chapter 4, we presented a solution approach for the LTL route planning process. In this chapter we focus on The *Dedicated Vehicle Assignment Problem* (DVAP), which aims to assign dedicated vehicles to LTL and FTL routes in the most efficient way.

We first provide a problem definition and present the mathematical formulation of the model. Then we represent our solution approach, which involves a constructive heuristic to find a starting solution and the solution of the mathematical model in through commercial solvers using the heuristic result as a starting solution. In the computational study, we show the potential saving that can be achieved through solving the DVAP on instances adopted from LTLP-C results.

5.1 Problem Definition

The purpose of DVAP is to assign dedicated vehicles, which can perform multiple routes, to previously planned routes with vehicle type information. Once we the planned route-vehicle type information is acquired, the earliest and latest start times of the routes can be calculated by finding room for pushing the start time of the route backward or forward. This can be done by examining the deadlines and planned pickup and delivery times of

each order assigned to the route. There exists an interdependence between routes which interchange orders at cross-dock terminals. This constraint should be taken into consideration in DVAP in order to maintain the synchronization.

Another important input of DVAP is the information regarding the dedicated vehicles. The current geographical locations of these vehicles are obtained from GPS devices. Each dedicated vehicle also has a home location. We assume that the vehicles will return to their home locations upon the completion of their routes. Dedicated vehicles have a non-use cost, which is incurred when no orders are assigned to them in a certain transportation plan. The non-use costs are used to reflect the empty travel cost of dedicated vehicles for returning their home locations.

A sample input for the DVAP is illustrated in Figure 5.1. Routes displayed in different colors are either FTL routes or they are derived from the outputs of the LTLP. For simplicity, we assume that two types of trucks (small and large) are available. The routes represented by yellow, orange, red and dark red lines illustrate the routes assigned to small vehicles, which are of the same physical type as purple and black trucks. The remaining two routes (light and dark blue) are assumed to be assigned to large vehicles. Note that the initial locations and home locations of the vehicles are displayed by the same colors. Our aim is to find a solution similar to the one depicted in Figure 5.2, where the dashed lines represent movements of the empty vehicles. There are three types of empty vehicle movements, namely, the movement of a vehicle from its initial location to the starting location of its first assigned route, from the ending location of a route to the starting location of the next route and finally from the ending location of its last assigned route to its home location.

We formulate an MILP model which includes all the practical constraints and requirements associated with the DVAP. This model takes route attributes such as route duration, earliest and latest route start times and route dependency relationships as fixed parameters from the solution of the base problem.

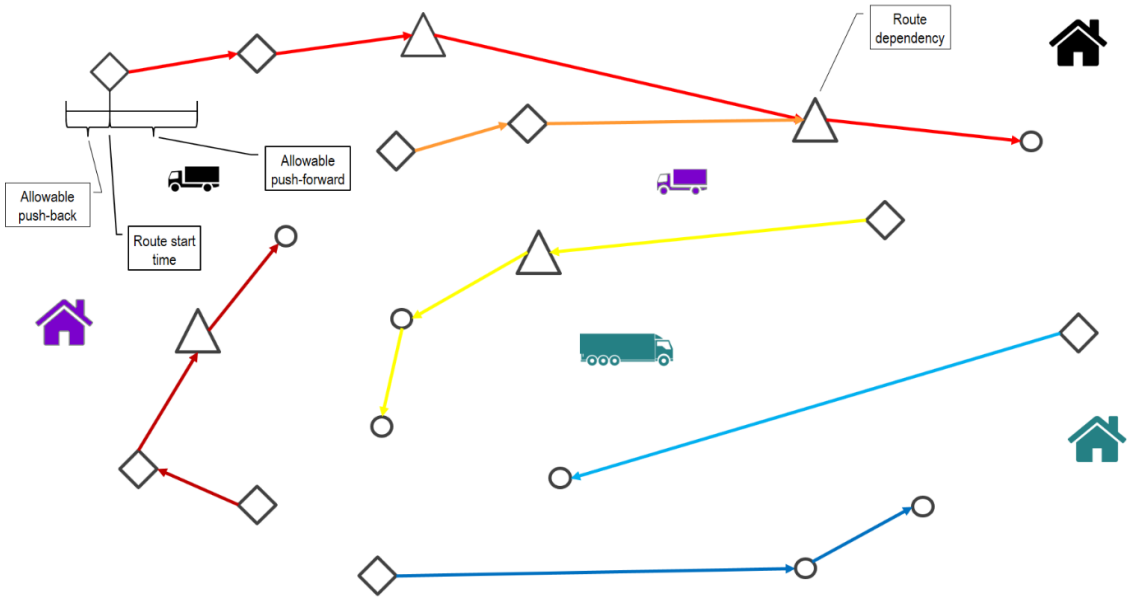


Figure 5.1. Sample DVAP Inputs.

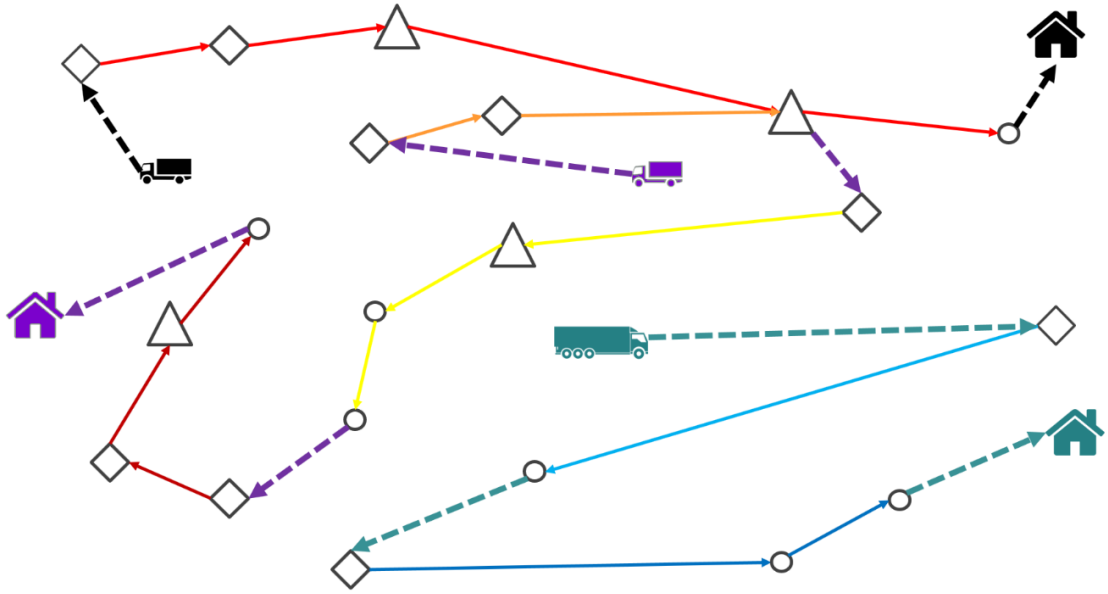


Figure 5.2. Sample Solution to the DVAP.

Assume that we have a set of planned routes (R), a set of dedicated vehicles (V) and a set of locations (L). Let the initial and home locations of vehicle v be denoted by i_v and h_v, respectively. The cost of assigning route r_2 right after route r_1 to vehicle v is shown by $C_{vr_1r_2}$. We define two other cost terms I_{rv} and H_{rv}, which are associated with the

travel of vehicle v from its initial location to the first location of route r and the travel of vehicle v to its home location from the last location of route r, respectively. The saving generated by replacing the spot-hired vehicle with dedicated vehicle v for route r is given as S_{rv}. N_v stands for the non-use cost of vehicle v. RV_{rv} is a binary parameter which indicates whether vehicle v can be assigned to route r. TI_{rv} represents the travel time between the initial location of vehicle v and the first location of route r, whereas TH_{rv} denotes the travel time between the last location of route r and the home location of vehicle v. Similarly, $TR_{r_1 r_2}$ is the travel time between the last location of route r_1 and the first location of route r_2. RD_r is the total duration of route r and RT_r is the required rest time after completing route r. ES_r and LS_r are the earliest and latest start times of route r. AL_{lr} denotes the time until arrival to location l in route r and UT_l is the unloading time at location l. Finally, we introduce the binary parameter $TR_{lr_1r_2}$, which takes the value of one if there is a transfer from route r_1 to route r_2 at location l and zero, otherwise. We assume that M is a sufficiently large number.

Decision variables of the model are defined as follows:

$$x_{vr_1r_2} : \begin{cases} 1, if\ vehicle\ v\ executes\ route\ r_2\ right\ after\ r_1 \\ 0, otherwise \end{cases} \quad v \in V, r_1, r_2 \in R, r_1 \neq r_2$$

$$f_{rv} : \begin{cases} 1, if\ r\ is\ the\ first\ route\ executed\ by\ vehicle\ v \\ 0, otherwise \end{cases} \quad v \in V, r \in R$$

$$l_{rv} : \begin{cases} 1, if\ r\ is\ the\ last\ route\ executed\ by\ vehicle\ v \\ 0, otherwise \end{cases} \quad v \in V, r \in R$$

$$y_{rv} : \begin{cases} 1, if\ route\ r\ is\ assigned\ to\ vehicle\ v \\ 0, otherwise \end{cases} \quad v \in V, r \in R$$

$$u_v : \begin{cases} 1, if\ vehicle\ v\ is\ used \\ 0, otherwise \end{cases} \quad v \in V$$

$$a_r : Start\ time\ of\ route\ r \quad r \in R$$

The resulting MILP formulation is given as follows:

$$Maximize \quad \sum_{r\in R}\sum_{v\in V} S_{rv}y_{rv} + \sum_{v\in V} N_v u_v - \sum_{r_1\in R}\sum_{\substack{r_2\in R\\ r_1\neq r_2}}\sum_{v\in V} C_{vr_1r_2}x_{vr_1r_2} - \sum_{r\in R}\sum_{v\in V}(I_{rv}\, f_{rv} + H_{rv}l_{rv}) \tag{5.1}$$

subject to

$$f_{rv} \leq 1 \qquad v \in V \tag{5.2}$$

$$f_{rv} + \sum_{\substack{r_2\in R\\ r_1\neq r_2}} x_{vr_2r_1} = l_{rv} + \sum_{\substack{r_2\in R\\ r_1\neq r_2}} x_{vr_1r_2} \qquad v \in V, r_1 \in R \tag{5.3}$$

$$l_{rv} + \sum_{\substack{r_2\in R\\ r_1\neq r_2}} x_{vr_1r_2} = y_{r_1v} \qquad v \in V, r_1 \in R \tag{5.4}$$

$$y_{rv} \leq RV_{rv} \qquad v \in V, r \in R \tag{5.5}$$

$$u_v \leq \sum_{r\in R} y_{rv} \qquad v \in V \tag{5.6}$$

$$-a_r + TI_{rv} \leq M(1 - f_{rv}) \qquad v \in V, r \in R \tag{5.7}$$

$$a_{r_1} - a_{r_2} + RD_{r_1} + RT_{r_1} + TR_{r_1r_2} \leq M\left(1 - \sum_{v\in V} x_{vr_1r_2}\right) \qquad r_1, r_2 \in R, r_1 \neq r_2 \tag{5.8}$$

$$ES_r \leq a_r \leq LS_r \qquad r \in R \tag{5.9}$$

$$a_{r_1} + AL_{lr_1} + UT_l \leq a_{r_2} + AL_{lr_2} \qquad r_1, r_2 \in R, r_1 \neq r_2, \; l \in L, TR_{lr_1r_2} = 1 \tag{5.10}$$

$$x_{vr_1r_2} \in \{0,1\} \qquad v \in V, r_1, r_2 \in R, \; r_1 \neq r_2 \tag{5.11}$$

$$f_{rv} \in \{0,1\} \qquad v \in V, r \in R \tag{5.12}$$

$$l_{rv} \in \{0,1\} \qquad v \in V, r \in R \tag{5.13}$$

$$y_{rv} \in \{0,1\} \qquad v \in V, r \in R \tag{5.14}$$

$$u_v \in \{0,1\} \qquad v \in V \tag{5.15}$$

$$a_r \geq 0 \qquad r \in R \tag{5.16}$$

The objective function (5.1) is composed of four components. The first component represents the saving created by replacing the initial spot-hired vehicle of the route with a dedicated one. In the initial case, none of the dedicated vehicles are used. Therefore, we assume that a vehicle non-use cost is incurred for all vehicles. The second component of the objective function increases the total saving by adding the non-use cost of vehicles which are used. Three types of empty travels are encountered for a dedicated vehicle: the travel of the vehicle from the last location of one of its assigned routes to the first location of its succeeding route, the travel of the vehicle from its home location to the starting location of its first assigned route and the travel of the vehicle from the last location of its final route to its home location. The third and fourth components of the objective function account for empty travel costs in these cases.

Constraint (5.2) ensures that a dedicated vehicle can be assigned to at most one route. Constraint (5.3) indicates that once a vehicle performs a route, then it either switches to a new route or travels to its home location. Constraint (5.4) requires that a vehicle is assigned to a route if and only if it switches to a new route or travels to its home location upon the completion of the route. A vehicle can be assigned to a route only if the route is initially assigned to a vehicle type which has the same or smaller capacity. This requirement is imposed by constraint (5.5). Constraint (5.6) implies that a vehicle is said to be used only if it is assigned to at least one route. Constraint (5.7) guarantees that the starting time of a route is greater than or equal to the travel time between the origin location of the vehicle assigned to it and its starting location. Constraint (5.8) ensures that there is a resting time and an empty travel time between two successive routes performed by a vehicle. Constraint (5.9) states that the starting time of any route should be between predetermined time windows, which can be specified by the customers. Synchronization of the routes is maintained through constraint (5.10). This constraint requires that in case of a transfer from one route to another, the arrival time of the receiving vehicle is greater than or equal to the time at which the associated orders are unloaded at the cross-docking location. Constraints (5.11) to (5.15) and (5.16) ensures the binarity and non-negativity of decision variables.

Another real-life constraint dictates that the dedicated vehicles should return to their home locations before a designated deadline denoted by D_v. In order to incorporate this requirement into our model, we introduce a new variable, h_v, which represents the arrival

time of vehicle v at its home location. We add two new sets of constraints to our model to ensure that this requirement is fulfilled:

$$a_r - h_v + RD_r + RT_r + TH_{rv} \leq M(1 - l_{rv}) \qquad v \in V, r \in R \tag{5.17}$$

$$h_v \leq D_v \qquad v \in V \tag{5.18}$$

The presented problem accommodates a selective approach, that is, not all routes are necessarily assigned to a dedicated vehicle. Furthermore, route dependency constraints are defined to attain synchronization of the orders. The next section describes the solution approach we propose for addressing the DVAP

5.2 Solution Approach for the DVAP

The solution algorithm for the DVAP intends to solve the problem by assigning vehicles to the route obtained through the output of the LTLP and FTL orders. An exact solution to the DVAP problem can be found by solving the MILP model developed for the problem. However, preliminary experiments show that the MILP model is computationally difficult to solve for instances with several hundred routes. In fact, in the case with 10 dedicated vehicles, the solver is not able to find the optimal solution within 3 hours for instances with more than 30 routes. In order to handle this issue to some extent, we develop a constructive heuristic algorithm to supply the MILP solver with a feasible initial starting solution. The aim of the constructive heuristic is to improve the overall performance of the solver when it is run with a time limit.

The general flow of the constructive algorithm is illustrated in Figure 5.3.

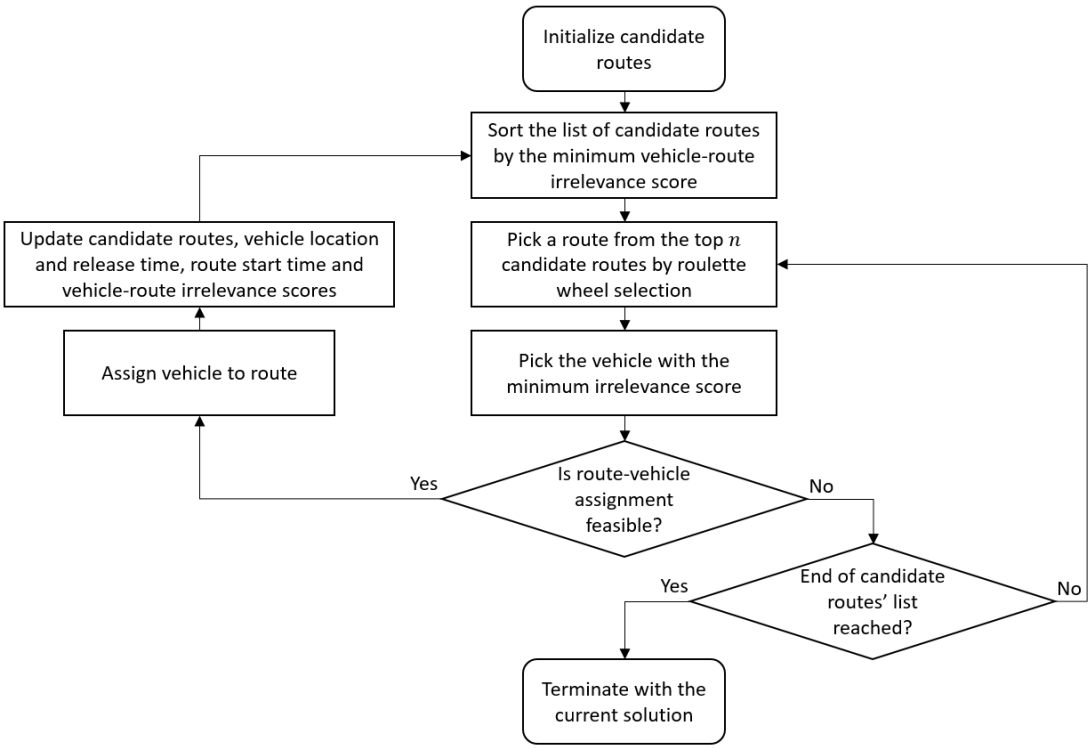


Figure 5.3. Constructive Heuristic for the DVAP.

We assume that there is a predecessor-successor relationship between some of the routes derived from the solution of the LTLP. This relationship exists if one route (successor) acquires one or more orders from another route (predecessor) at a cross-dock terminal. At the beginning of the algorithm, we initialize the list of candidate routes with routes having no predecessors.

Note that the list of candidate routes for a vehicle consists of only routes which are compatible with that vehicle. A vehicle compatible with a route if the total order size carried along the route is no greater than the capacity of the vehicle.

In the following step, we sort the list of candidate routes by first the total route distance (in descending order) and next the minimum vehicle-route irrelevance score (in ascending order). *Vehicle-route irrelevance* score (IS_{vr}) is calculated based on the current location of the vehicle (CL_v), vehicle release time (T_v) and the deadline (D_v) for the vehicle to return to its home location (HL_v). The value of vehicle-route irrelevance scores is updated as the algorithm proceeds. Let the first and last locations within a route be represented by FL_r and LL_r. The distance between two locations, l_1 and l_2 is given by $H_{l_1 l_2}$. The vehicle-route irrelevance score for vehicle v and route r is calculated as:

$$IS_{vr} = H_{CL_vFL_r} + \left(\frac{T_v}{D_v}\right)^2 H_{LL_rHL_v} \qquad (5.19)$$

The main idea behind multiplying the return distance from the last location of the route to the home location of the vehicle ($H_{LL_rHL_v}$) by the coefficient $\left(\frac{T_v}{D_v}\right)^2$ is to increase the effect of $H_{LL_rHL_v}$ on IS_{vr} as the deadline for returning to the home location gets closer.

Once the list of candidate routes is sorted, we pick a route from the top n routes in the list by a roulette wheel mechanism, where the probability of being selected for any route is proportional to minimum vehicle-route irrelevance score among all compatible vehicles of the route. After selecting the route, we pick the vehicle which has the minimum irrelevance score with the route and check if their assignment is feasible. The following conditions must hold for a feasible route-vehicle assignment:

- Start time of the route is between its earliest and latest start times.
- There is enough time left for the vehicle to return to its home location upon completing the route.
- If the route has any predecessors, arrival time at associated cross-docks is no earlier than the unloading time of orders to be acquired.
- If the route has any successors, unloading time of the orders to be transferred is no later than the maximum arrival time of the vehicle executing the succeeding route at the associated cross-docks.

These conditions ensure that the route dependency constraints hold. If the assignment is not feasible, the algorithm continues by picking another route from the list of candidate routes with the same roulette wheel selection mechanism. The algorithm terminates when the end of the list is reached and reports the assignments formed up to that point. The route-vehicle assignment is established if it is found to be feasible. In this case, location and the release time of the vehicle are updated, and the route start time is set to the earliest possible time. The route is removed from candidate routes' list and it is labeled as "processed". If all predecessors of a non-processed route become processed, then this route is added to the list of candidate routes. Afterwards, vehicle-route irrelevance scores

are updated for all elements in the list. Then the list is re-sorted, and the algorithm continues.

We performed several tests to understand the effect of employing the constructive heuristic before solving the MILP model on the best integer solution value and optimality gap for time-limited solutions. Since the heuristic is quite fast, we run many different replications in parallel and feed the MILP solver the solution with the maximum objective function value. Results of these tests indicate that having an initial starting solution reduces the optimality gaps by 20% to 60%. It also improves the best integer solutions by 10% to 30%, depending on the size of the test instance. These results reveal that when the constructive heuristic is employed to provide an initial feasible solution to the MILP, the overall quality of the solutions obtained increases significantly.

In the next section, we present the results of our computational experiments conducted for the DVAP.

5.3 Computational Experiments for the DVAP

In this computational study, we solve the DVAP through the solution approach presented in the previous section. We aim to understand to what extend the DVAP is capable of reducing the solution cost of the LTLP. We run our experiments by using several LTLP-C outputs as inputs to the DVAP. We solve the LTLP-C instances using the matheuristic algorithm presented in Section 4.2.2. We impose a time limit of 10 minutes (600 seconds) on each LTLP-C-Sub. Results obtained from the matheuristic are passed on to the DVAP algorithm. We provide an initial starting solution to the MILP model for the DVAP by running the constructive heuristic. Then we solve the MILP model for the DVAP with a time limit of one hour (3,600 seconds). We run 100 replications of the constructive heuristic in parallel and pick the solution with the maximum solution profit. Three instance types 1Z-10L, 5Z-10L and 10Z-10L, with four instances from each type are used in our experiments. For each such instance, we test two alternative scenarios where 5 and 10 dedicated vehicles of two physical vehicle type are available. For instances with 5 and 10 dedicated vehicles, there is an initial total vehicle non-use cost of 600 and 1,250, respectively. Values of the parameters related to the DVAP algorithm are provided in

Appendix F. The notation associated with the results are tabulated in Table 5.1 while the actual results are presented in Table 5.2.

Table 5.1. Description of Result Parameters.

Parameter	Description
n_{veh}	Number of dedicated vehicles available
C_{LTLP}	Total cost of LTLP solution
P_{DVAP}^{init}	Initial solution profit achieved by running the constructive heuristic
P_{DVAP}^{total}	Total solution profit achieved by solving DVAP model
C_{DVAP}	Final solution cost after deducting DVAP saving from LTLP cost
$\%\Delta_{DVAP}$	Percentage improvement achieved by DVAP
T_{DVAP}^{init}	Solution time for constructive heuristic (sec)
T_{DVAP}^{MILP}	Solution time for running the MILP for the DVAP (sec)
G_{DVAP}	Optimality GAP of MILP for the DVAP

Table 5.2. Results of DVAP Algorithm.

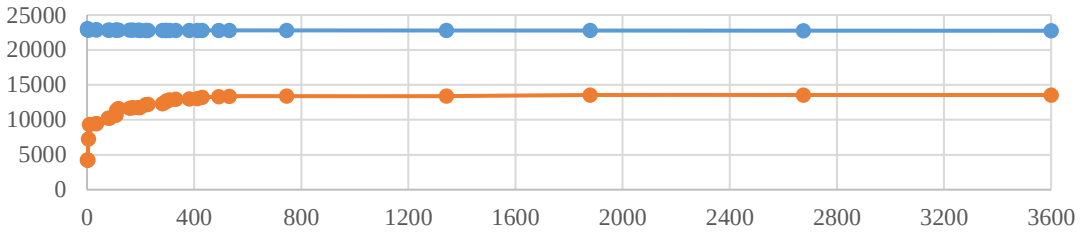
Instance	n_{veh}	C_{LTLP}	P_{DVAP}^{init}	P_{DVAP}^{total}	C_{DVAP}	$\%\Delta_{DVAP}$	T_{DVAP}^{init}	T_{DVAP}^{MILP}	G_{DVAP}
1Z-10L-20	5	9347.63	3257.20	3529.85	6417.78	35.5%	0.85	0.92	0.0%
1Z-10L-50	5	18824.09	6731.39	7384.46	12039.63	38.0%	1.51	3613.37	2.3%
1Z-10L-100	5	31383.89	6659.20	7524.99	24458.91	23.5%	2.61	3603.84	55.6%
1Z-10L-200	5	55241.80	6312.88	7621.88	48219.92	13.6%	4.80	3607.59	161.4%
5Z-10L-20	5	40602.09	1413.70	5923.01	35279.08	14.4%	1.27	3607.13	16.2%
5Z-10L-50	5	96982.23	4255.26	13552.64	84029.59	13.9%	2.91	3601.02	67.8%
5Z-10L-100	5	167771.22	5354.10	11185.61	157185.60	6.6%	5.37	3600.78	276.7%
5Z-10L-200	5	289992.31	5675.72	10617.99	279974.32	3.7%	12.09	3607.86	629.9%
10Z-10L-20	5	55458.41	2478.11	7358.32	48700.09	13.1%	1.59	3606.37	44.4%
10Z-10L-50	5	123990.71	2091.65	10118.23	114472.48	8.1%	2.26	3601.37	147.1%
10Z-10L-100	5	216459.53	4025.84	12242.46	204817.07	5.6%	5.48	3601.77	307.9%
10Z-10L-200	5	387941.56	4838.65	12944.71	375596.85	3.3%	12.97	3606.78	603.1%
1Z-10L-20	10	9347.63	3272.16	3979.85	6617.78	37.6%	0.84	1.78	0.0%
1Z-10L-50	10	18824.09	7488.26	7951.43	12122.66	39.6%	2.65	3603.22	0.7%
1Z-10L-100	10	31383.89	11245.01	11759.31	20874.58	36.0%	4.90	3607.02	3.4%
1Z-10L-200	10	55241.80	12847.67	14561.50	41930.30	25.8%	9.33	3613.98	39.9%
5Z-10L-20	10	40602.09	2395.02	7488.79	34363.30	17.9%	2.11	376.77	0.0%

5Z-10L-50	10	96982.23	4842.70	19844.14	78388.08	20.2%	5.27	3610.95	18.2%
5Z-10L-100	10	167771.22	10181.07	20380.37	148640.85	12.1%	10.90	3601.58	110.0%
5Z-10L-200	10	289992.31	12276.08	18691.31	272551.00	6.4%	22.08	3601.54	318.1%
10Z-10L-20	10	55458.41	1825.11	9823.11	46885.30	17.3%	2.25	3605.47	15.7%
10Z-10L-50	10	123990.71	2883.86	3529.85	112586.04	10.1%	5.01	3601.73	103.2%
10Z-10L-100	10	216459.53	2570.91	7384.46	204138.10	6.2%	9.87	3601.95	273.3%
10Z-10L-200	10	387941.56	7786.03	7524.99	373655.16	4.0%	21.81	3602.34	489.9%

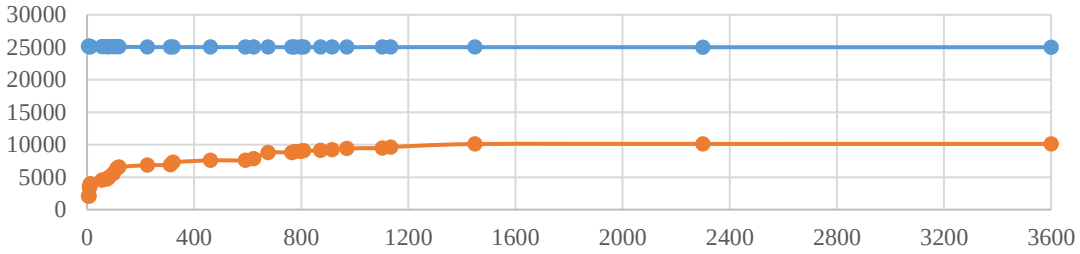
The MILP for the DVAP models a computationally difficult problem and evidently it cannot be solved optimally by commercial solvers for most of the instances with tested sizes. Since we provide a feasible initial starting solution via our constructive heuristic, a certain amount of saving is generated even for the largest instances. Since DVAP is a maximization problem, the optimality gap represents the percentage difference between the objective function upper bound and the best integer solution value. The optimality gaps of the MILP are quite high, especially for instances with more than 20 orders.

In order to understand the behavior of the solver when the MILP is solved and the reason for the high optimality gaps, we analyze the upper bound and lower bound values obtained throughout the solution period. Figure 5.4 represents the changes in the upper bounds and lower bounds for selected instances which are 5Z-10L-50 (a), 10Z-10L-50 (b) and 10Z-10L-100 (c) with 5 dedicated vehicles and 5Z-10L-100 (d) and 1Z-10L-200 (e) with 10 dedicated vehicles. In all figures, the y-axis denotes the solution cost and the x-axis shows the solution time. The blue lines and orange lines designate the upper bound and lower bound values, respectively.

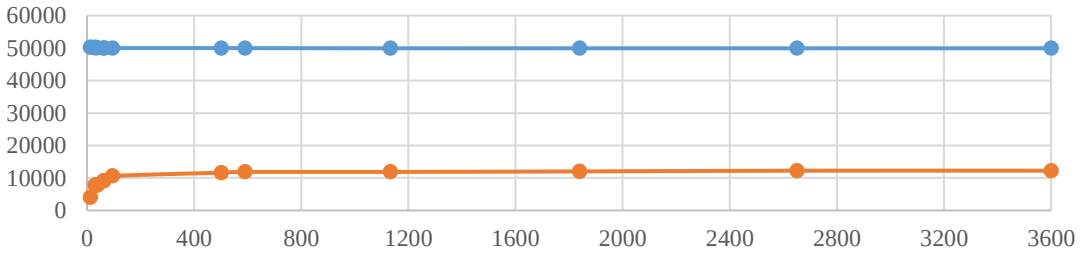
The figures associated with the solution of the selected instances demonstrate that while the solver was capable of improving the lower bounds to some degree throughout the solution period by finding better feasible solutions, the improvement in the upper bounds is quite limited. Therefore the high solution upper bounds is the main driver behind the high optimality gaps in the related DVAP instances.



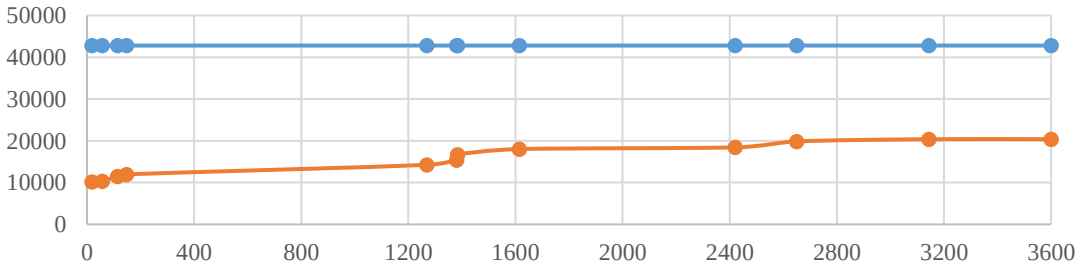
(a)



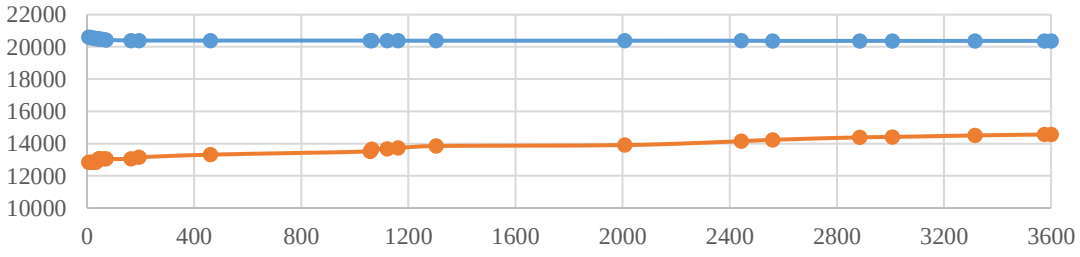
(b)



(c)



(d)



(e)

Figure 5.4. Progression of the Lower Bound and Upper Bound Values During MILP Solution for Instances with 5 Dedicated Vehicles (a) 5Z-10L-50, (b) 10Z-10L-50, (c) 10Z-10L-100 and 10 Dedicated Vehicles (d) 5Z-10L-100, (e) 1Z-10L-200.

An interesting remark can be made about the effect of the number of dedicated vehicles on the optimality gaps. Although a higher number of vehicles renders the MILP model more complex in terms of the number of decision variables and constraints, the optimality gap of each instance with 10 dedicated vehicles lower than that of the associated instance with 5 dedicated vehicles (except instance 1Z-10L-20, which is solved optimally in both cases). The reason for this can be understood by examining Figure 5.5, which illustrates the final values of the solution lower bounds and upper bounds for each instance. Solution upper bounds are nearly the same for variants with 5 and 10 dedicated vehicles of each instance whereas the lower bounds are higher for instances with 10 dedicated vehicles. This results in lower optimality gaps for instances with 10 dedicated vehicles, although the underlying MILP models of these instances are larger than those of the instances with 5 dedicated vehicles in terms of problem variables and constraints associated with the vehicles.

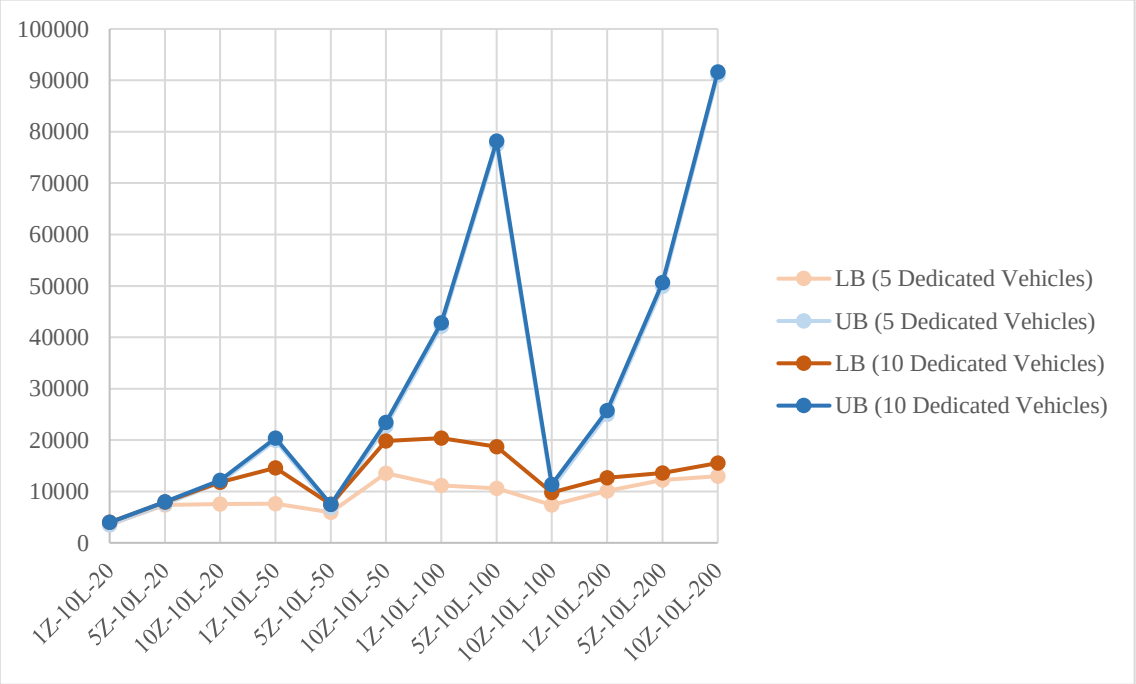


Figure 5.5. Solution Lower Bound (LB) and Upper Bound (UB) Values.

Figure 5.6 shows that the solution profit obtained within the solver time limit unexpectedly decreases as the instance size gets larger than 100 in 5Z-10L instances with

5 and 10 dedicated vehicles. This indicates that the quality of the best integer solutions is low for large instances.

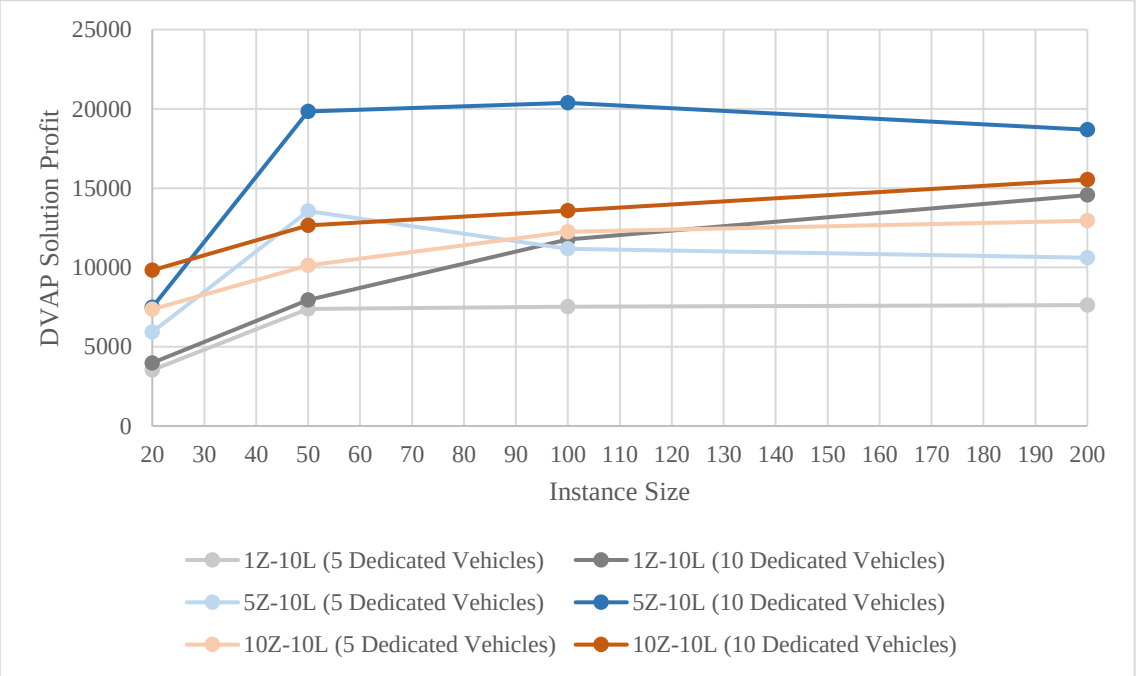


Figure 5.6. DVAP Solution Profit with Respect to Instance Size.

The results of the experiment demonstrate that the percentage improvement achieved by solving DVAP problem gets lower as the number of zones in the problem network get higher. Moreover, having more dedicated vehicles results in better improvement rates in general. This deduction can be supplemented by analyzing the ratio of routes performed by dedicated vehicles to the total number of routes obtained in the solution.

Figure 5.7 illustrates the percentage of routes performed by dedicated vehicles with respect to the problem size for each instance type. There is a higher use rate of dedicated vehicles in problems with lower number of zones. As expected, having more vehicles results in a higher coverage of the routes with dedicated vehicles.

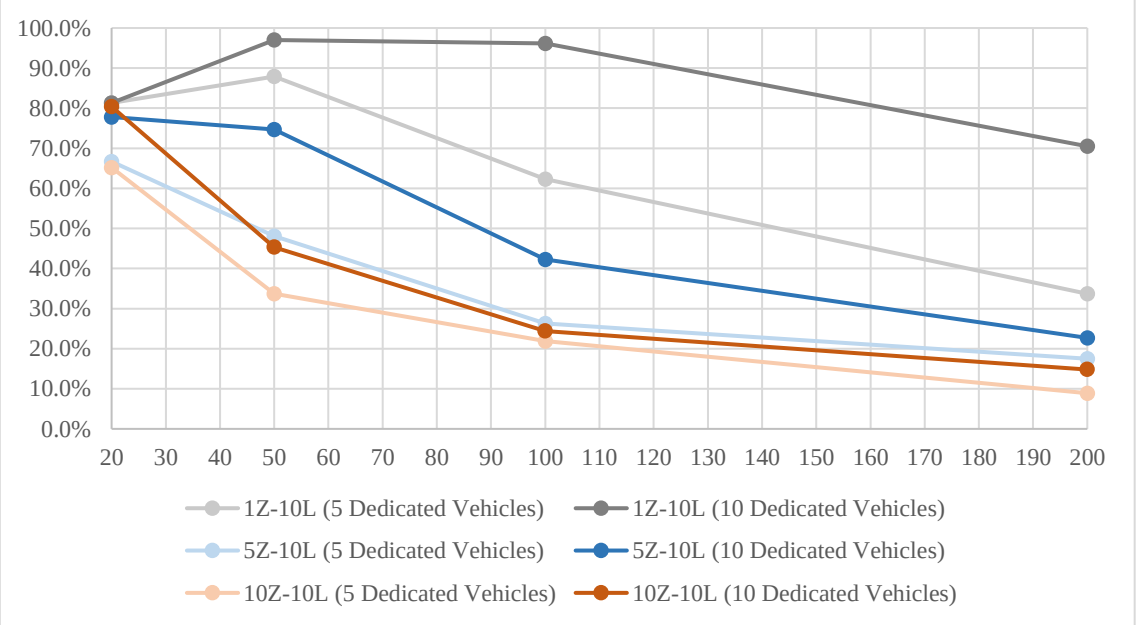


Figure 5.7. Percentage of Routes Performed by Dedicated Vehicles.

In this chapter we presented the DVAP, which can either be used as an extension to the LTLP or can be solved on its own for assigning dedicated vehicles to previously defined routes for FTL carriers. In the next chapter we focus on the last-mile delivery problem, which possess another important challenge for the road transport operations of the 3PL company.

Chapter 6

THE LAST-MILE DELIVERY PLANNING PROBLEM

This chapter addresses the last-mile delivery planning problem (LMDP) of the 3PL carrier we are engaged with. From perspective of the company, last-mile delivery operations represent the shipment of products from a central depot, which can be a regional warehouse or distribution center, to end customers and small shops. The 3PL company operates in over thirty last-mile delivery networks. These operations are independent and each employs a dedicated fleet to fulfill the customers' delivery requests.

Last-mile delivery is one of the main challenges in city logistics, and the problem we deal with can be easily extended to other logistics service providers, or any carriers that are involved in distribution of goods to customers from a central depot in urban areas. This type of problems are addressed by Vehicle Routing Problems (VRPs) by researchers. The problem we aim to handle incorporates many practical rules and constraints and has several distinctive characteristics arising from real-life requirements. In that respect, it can be classified as a "rich" VRP.

In the rest of the chapter, we provide a definition of the problem along with a mathematical programming formulation. We propose a *Hybrid Evolutionary Algorithm* (HEA) to solve practical instances of the problem. In the computational study, we first compare the results of HEA against the mathematical programming solver results for small instances. Then we run a comparison with benchmark instances of a special version of the problem.

6.1 Problem Definition

In the Rich VRP that we address, deliveries are carried out by different types of vehicles. There is a fixed cost associated with the daily usage of a vehicle. Additionally, a variable cost is incurred in proportion to the distance traveled by each vehicle. Fixed

and variable cost coefficients and vehicle capacities vary from one vehicle type to another. The vehicle fleet size is limited and the maximum number of vehicles that can be used in a single day is known for each vehicle type. A trip is defined as a sequence of visits to delivery locations and it starts and ends at a given depot location. Vehicles can perform more than one trip in a single day. In accordance with this property, we name the problem can be classified as a *Multi-trip Rich VRP* (MTRVRP).

Daily shift start and end times are defined for drivers of the vehicles. An overtime compensation is paid if the drivers work beyond their shift end times. Vehicles have specific final locations, which usually represents the home of the truck driver. Any used vehicle returns to its final location at the end of the shift. There is a time window associated with each delivery, that is, the vehicles are expected to arrive at each customer location within a designated time interval. Early arrivals are not allowed. Therefore, even if the vehicle arrives at a customer location before the earliest delivery time, it has to wait until the time window start time. Late arrivals are possible, but a penalty fee is paid based on the amount of delay. There exists a service time for unloading operations at the customer locations depending on the size of delivery. Similarly, a certain amount of time is consumed for reloading operations at the depot. Each customer can be served by a restricted set of vehicle types, which are referred to as the accessibility restrictions. This limitation arises from two practical requirements: Large vehicles cannot enter congested areas and some vehicles can only serve customers in specific regions. It is possible that a customer is left unserved, in which case an unfulfillment cost arises. This cost depends on the size and the importance of the order.

The problem is defined on a directed graph with the set of nodes (N^+) and set of arcs (A^+). We formulate the problem as an MILP model. The description of sets, parameters, decision variables and the mathematical model formulation is provided as follows:

Sets:

N: Customer nodes, $N = \{1,2, \dots, n\}$

N^+: All graph nodes including depot $\{0\}$, dummy trip-ending node $\{n+1\}$ and vehicle final nodes defined as $E = \{e_k : k \in V\}$ where $N^+ = N \cup \{0\} \cup \{n+1\} \cup E$

A: Set of all arcs (i, j) where $i \in N \cup \{0\}$, $j \in N \cup \{0\} \cup E$ and $i \neq j$

A^+: Set of all arcs including a dummy arc between the dummy-trip ending node and depot, $A^+ = A \cup \{(n+1,0)\}$

V: Vehicles

R: Trip indices

Parameters:

β_{ik}	:	$\begin{cases}1, if\ customer\ i\ can\ be\ served\ by\ vehicle\ k\\0, otherwise\end{cases}$	$i \in N, k \in V$
c_{ijk}^t	:	$Traveling\ cost\ on\ arc\ (i,j)\ by\ vehicle\ k$	$(i,j) \in A, k \in V$
c_k^f	:	$Fixed\ cost\ of\ using\ vehicle\ k$	$k \in V$
c_k^o	:	$Overtime\ cost\ per\ hour\ per\ vehicle\ k$	$k \in V$
c_i^l	:	$Late\ delivery\ penalty\ per\ hour\ for\ customer\ i$	$i \in N$
c_i^u	:	$Cost\ of\ not\ visiting\ customer\ i$	$i \in N$
q_i	:	$Demand\ of\ customer\ i$	$i \in N$
Q_k	:	$Capacity\ of\ vehicle\ k$	$k \in V$
s_i	:	$Unloading\ time\ for\ customer\ i$	$i \in N$
σ_k	:	$Loading\ time\ of\ vehicle\ k\ at\ the\ depot$	$k \in V$
t_{ij}	:	$Travel\ time\ on\ arc\ (i,j)$	$(i,j) \in A$
a_i	:	$Time\ window\ start\ for\ customer\ i$	$i \in N$
b_i	:	$Time\ window\ end\ for\ customer\ i$	$i \in N$
d_k	:	$Normal\ working\ hours\ for\ vehicle\ k$	$k \in V$

Decision Variables:

x_{ijkr}	:	$\begin{cases}1, if\ vehicle\ traverses\ arc\ (i,j)\ on\ its\ r^{th}\ trip\\0, otherwise\end{cases}$	$(i,j) \in A^+,$ $k \in V, r \in R$
y_{ikr}	:	$\begin{cases}1, if\ customer\ i\ is\ visited\ in\ the\ r^{th}\ trip\ of\ vehicle\ k\\0, otherwise\end{cases}$	$i \in N, k \in V,$ $r \in R$
z_{kr}	:	$\begin{cases}1, if\ the\ r^{th}\ trip\ of\ vehicle\ k\ is\ used\\0, otherwise\end{cases}$	$k \in V, r \in R$

f_{kr} : $\begin{cases}1, if\ the\ r^{th}\ trip\ of\ vehicle\ k\ is\ its\ final\ trip\\0, otherwise\end{cases}$ $k \in V, r \in R$

v_k : $\begin{cases}1, if\ vehicle\ k\ is\ used\\0, otherwise\end{cases}$ $k \in V$

u_i : $\begin{cases}1, if\ customer\ i\ is\ not\ visited\\0, otherwise\end{cases}$ $i \in N$

p_{ikr} : $Arrival\ time\ at\ customer\ i\ by\ the\ vehicle\ k\ in\ its\ r^{th}\ tri$ $i \in N, k \in V, r \in R$

l_i : $Lateness\ of\ arrival\ at\ customer\ i$ $i \in N$

o_k : $Overtime\ work\ for\ vehicle\ k$ $k \in V$

The resulting mathematical programming formulation of LMDP is given as follows:

$$Minimize \quad \sum_{(i,j)\in A}\sum_{k\in V}\sum_{r\in R} c^t_{ijk}x_{ijkr} + \sum_{k\in V} c^f_k v_k + \sum_{k\in V} c^o_k o_k + \sum_{i\in N} c^l_i l_i + \sum_{i\in N} c^u_i u_i \tag{6.1}$$

$subject\ to$

$$\sum_{j\in N^+} x_{ijkr} = y_{ikr} \qquad i \in N, k \in V, r \in R \tag{6.2}$$

$$\sum_{k\in V}\sum_{r\in R} y_{ikr} + u_i = 1 \qquad i \in N \tag{6.3}$$

$$y_{ikr} \le \beta_{ik} v_k \qquad i \in N, k \in V, r \in R \tag{6.4}$$

$$\sum_{i\in N\cup\{0\}} x_{ihkr} - \sum_{j\in N^+} x_{hjkr} = 0 \qquad h \in N, k \in V, r \in R \tag{6.5}$$

$$\sum_{i\in N^+} x_{0ikr} = 1 \qquad k \in V, r \in R \tag{6.6}$$

$$\sum_{i\in N^+} x_{i(n+1)kr} + \sum_{i\in N} x_{ie_kkr} = 1 \qquad k \in V, r \in R \tag{6.7}$$

$$\sum_{i \in N} q_i y_{ikr} \leq Q_k \qquad k \in V, r \in R \qquad (6.8)$$

$$p_{ikr} + s_i + t_{ij} - M\left(1 - x_{ijkr}\right) \leq p_{jkr} \qquad (i,j) \in A^+, k \in V, r \in R \qquad (6.9)$$

$$a_i y_{ikr} \leq p_{ikr} \leq b_i y_{ikr} + l_i \qquad i \in N, k \in V, r \in R \qquad (6.10)$$

$$p_{(n+1)kr} + \sigma_k \leq p_{0k(r+1)} \qquad k \in V, r \in R \setminus \{|R|\} \qquad (6.11)$$

$$p_{e_k kr} \leq d_k + o_k + M(1 - f_{kr}) \qquad k \in V, r \in R \qquad (6.12)$$

$$y_{ikr} \leq z_{kr} \qquad i \in N, k \in V, r \in R \qquad (6.13)$$

$$z_{kr} \leq \sum_{i \in N} y_{ikr} \qquad k \in V, r \in R \qquad (6.14)$$

$$z_{k(r+1)} - z_{kr} \leq 0 \qquad k \in V, r \in R \setminus \{|R|\} \qquad (6.15)$$

$$\sum_{r \in R} z_{kr} \leq \sum_{r \in R} r f_{kr} \qquad k \in V \qquad (6.16)$$

$$f_{kr} \leq z_{kr} \qquad k \in V, r \in R \qquad (6.17)$$

$$\sum_{i \in N} x_{ie_k kr} = f_{kr} \qquad k \in V, r \in R \qquad (6.18)$$

$$\sum_{r \in R} f_{kr} \leq 1 \qquad k \in V \qquad (6.19)$$

$$x_{ijkr} \in \{0,1\} \qquad (i,j) \in A^+, k \in V, r \in R \qquad (6.20)$$

$$y_{ikr} \in \{0,1\} \qquad i \in N, k \in V, r \in R \qquad (6.21)$$

$$v_k,\ o_k \geq 0 \qquad k \in V \qquad (6.22)$$

$$p_{ikr} \geq 0 \qquad i \in N, k \in V, r \in R \qquad (6.23)$$

$$l_i \geq 0 \qquad i \in N \qquad (6.24)$$

The objective function of the model combines five different cost components. The first cost component represents the travel cost on an arc by a specific vehicle. In practice, this component embodies the total fuel cost where the coefficient c_{ijk}^t is the cost of fuel

consumed by vehicle k when traversing arc (i, j). The second cost component is associated with the fixed cost of using a certain vehicle. The third and fourth cost terms demonstrate the overtime cost incurred due to delays in total route completion time and the penalty cost for late arrivals at customer locations, respectively. The cost of leaving a customer unserved is established in the last cost component.

Constraint (6.2) ensure that each customer is visited at most once. A customer is considered to be unserved if it is not visited by any vehicle (6.3). Constraint (6.4) enforces the accessibility restriction for each vehicle. Flow balance within each route is guaranteed by constraints (6.5) – (6.7). Vehicle capacity limits are imposed by constraint (6.8). The time schedule within a trip is established by constraint (6.9), which also eliminates possible subtours. Constraint (6.10) demonstrates the time windows for customer arrivals whereas constraint (6.12) defines the soft duration limit on the total route time. The time-synchronization of trips within a route is satisfied through constraint (6.11). Constraints (6.13) and (6.14) ensure that the variable z_{kr} is assigned to its appropriate value. Constraint (6.15) is a symmetry breaking valid inequality. This constraint ensures that a trip with a large index cannot be initiated unless the trips of the same route with the smaller indices (if any) are performed. Constraints (6.16) – (6.19) guarantee that each used vehicle returns to its designated final location upon completing its last trip.

Our preliminary experiments have shown that the MILP model we propose is intractable for practical size problem instances. Although it differs from region to region, the 3PL company processes around 200 last-mile delivery orders on average per day per network. Therefore, an alternative strategy is required for solving LMDP instances. In the next section we present a hybrid evolutionary algorithm, which aims to provide high quality solutions to practical size problems.

6.2 Hybrid Evolutionary Algorithm

Evolutionary algorithms have been widely used for solving different variants of VRPs. Hybridization techniques aim to improve the performance of evolutionary algorithms for real-world problems with high complexity. We propose a *Hybrid Evolutionary Algorithm* (HEA) which exploits solutions with an *Adaptive Large Neighborhood Search* (ALNS) heuristic within a parallelized genetic algorithm framework. Each element of the

algorithm is designed in such a way that the practical requirements of the LMDP are handled. In this section provide an overview of the HEA followed by the details of each algorithm component.

6.2.1 Overview of the Algorithm

The pseudocode of the HEA is provided in Algorithm 6.1. Firstly, the initial population is formed in steps 1 through 5. Algorithm starts with the construction of n_{pop} initial chromosomes in the form of customer node arrays via a randomized greedy approach. Each chromosome then goes through a splitting procedure which partitions the arrays into vehicle trips. Each solution obtained is exposed to mutation via destroy and repair operators of the ALNS algorithm. The resulting pool of modified solutions constitute the initial population of the problem. The construction, splitting and mutation operations are carried out in a parallel fashion using $n_{process}$ distinct threads.

Steps 6 through 15 represent the improvement phase of the algorithm. The improvement phase continues for n_{gen} generations. At each generation, n_{cross} new individuals are added to the population. At the beginning, n_{cross} pairs of parents are selected. Two child chromosomes are generated for each pair via crossover operation. Each child chromosome goes through splitting and mutation. The one with the lower fitness value (f) is inserted into the population. This procedure is executed in $n_{process}$ parallel threads.

After all new individuals are explored, the weights of destroy and repair operators used in mutation are updated for the next generation. Once the improvement phase is over, n_{boost} best distinct solutions in the population undergo a boosting procedure in a parallel fashion in order to improve the solution quality further.

Algorithm 6.1. Overview of HEA

1: *Initialize weights of operators used in mutation algorithm*
2: *Initialize the population in $n_{process}$ parallel threads:*
3: *Construct the pool of initial chromosomes where the pool size is n_{pop}*
4: *Split chromosomes*
5: *Mutate solutions and obtain initial population*
6: *For $i = 1$ through n_{gen}:*
7: *Select n_{cross} pairs of parents from the population*

8: *Distribute n_{cross} pairs into $n_{process}$ groups evenly for parallel processing*
9: *For each group, execute steps 10 through 13 in parallel:*
10: *Generate child chromosomes C_1 and C_2 for each pair of parents through crossover*
11: *Split chromosomes C_1 and C_2 and obtain solutions S_1 and S_2*
12: *Mutate solutions S_1 and S_2 and obtain S_1^* and S_2^**
13: *If $f_{S_1^*} \leq f_{S_2^*}$, insert S_1^*, otherwise insert S_2^* into population*
14: *Update weights of operators used in mutation algorithm*
15: *If $i < n_{gen}$ form the next generation by selecting n_{pop} survivors, otherwise select n_{boost} solutions for boosting*
16: *Boost n_{boost} solutions in $n_{process}$ parallel threads through ALNS algorithm and report the best solution*

6.2.2 Construction of Initial Chromosomes

Any chromosome used within the hybrid evolutionary algorithm has two elements: A permutation of customer nodes (C_{nodes}) and a permutation of vehicles ($C_{vehicles}$). Before constructing the initial chromosomes, we first calculate a modified distance figure between each customer pair, denoted by δ_{ij} for customer nodes i and j. δ_{ij} depends on the distance between the nodes (d_{ij}), whether the nodes have the same allowed vehicles or not (g_{ij}) and the difference between the time windows of the two nodes (w_{ij}).

g_{ij} equals to 1 if i and j can be served by the same set of vehicles and μ_1, otherwise. w_{ij} equals to 1 if the time windows of i and j overlap and $\mu_2\left|(a_i + b_i)/2 - (a_j + b_j)/2\right|$, otherwise. μ_1 and μ_2 are coefficients normalized for a proper modified distance measure.

The modified distance is calculated as:

$$\delta_{ij} = d_{ij} g_{ij} w_{ij} \qquad (6.25)$$

The algorithm for construction an initial chromosome (C) is given in Algorithm 6.2.

Algorithm 6.2. Construction of Initial Chromosome

1: *Initialization: $C_{nodes} := ( )$, $C_{vehicles} :=$ A random permutation of vehicles*
2: *Pick a random node, remove it from N and append it to C_{nodes}*
3: *While $|N| > 0$, do:*
4: *Sort the nodes $j \in N$ in ascending order by δ_{ij} where i is the last node in C_{nodes}*

5: *Randomly pick a node from the top n_{top} elements of N, remove it from N and append it to C_{nodes}*

6: Let $C := (C_{nodes}, C_{vehicles})$ and return C

6.2.3 Split Algorithm

The purpose of split algorithm is to partition C_{nodes} of a chromosome C into vehicle trips feasible in terms of capacity constraints with respect to the given vehicle ordering $C_{vehicles}$. The output of the split algorithm is a solution $S = (T, U)$ where T represents the trips and U is the set of unserved customers. Assume that $C_{vehicles} = (1, \dots, |V|)$ and the trip indices range from 1 to r_i for vehicle i. T_{ij} denotes the ordered set of customer nodes in the j^{th} trip of vehicle i. Then, T takes the following form:

$$T = \left(\left(v_1, \left(T_{11}, T_{12} \ \dots, T_{1r_1}\right)\right), \dots, \left(v_{|V|}, \left(T_{|V|1}, T_{|V|2} \ \dots, T_{|V|r_{|V|}}\right)\right)\right) \qquad (6.26)$$

Let us consider the following example. $C = \left((1,2,3,4,5,6,7,8,9,10), (A, B, C)\right)$. Let us assume that in the solution (S) we obtain by splitting C, customers 1-3 are assigned to the first trip of vehicle A, customers 4-5 are assigned to the second trip of vehicle A, customers 6-8 are assigned to the first trip of vehicle B and customers 9-10 are unserved. The solution representation for the given example would be as follows:

$$S = \left(\left(\left(A, \left((1,2,3), (4,5)\right)\right), \left(B, \left((6,7,8)\right)\right), \left(C, ( )\right)\right), \{9,10\}\right) \qquad (6.27)$$

We model the splitting problem as a shortest path problem from a dummy source node s to a dummy sink node t. Every other node in the problem network is represented by the index of the node and the ordered list of unused vehicles such that $n = (I_n, V_n)$ where I_n represents the index of the node associated with n and V_n is a permutation of vehicles that are not used until arriving at n. We employ only forward arcs in our network, that is, the index of the origin node of an arc is always smaller than that of the destination node. We define three types of arcs. Trip arcs are associated with the list of trips to be performed by a certain vehicle. These are represented by the origin and destination nodes, vehicle and trip information. Any trip arc a is denoted by $a = \left(org_a, dest_a, v_a, \left(T_{a1}, T_{a2} \ \dots, T_{ar_a}\right)\right)$ where org_a is the origin node, $dest_a$ is the destination node, v_a is the vehicle associated with arc a, r_a is the number of trips within

a and T_{aj} is the ordered list of visited customer nodes in the j^{th} trip of a. The next arc type is called vehicle-skipping arcs, which is a special case of trip arcs where the trips are empty, hence the vehicle is not used. Arcs of this type are shown as $a = (org_a, dest_a, v_a, (\,))$. The final arc type, which is named as the non-serving arcs, represents unvisited customers. All arcs of this type has the destination node t, which is the dummy sink node. This means that non-serving arcs cannot be succeeded by trip arcs and only the customers closer to the end of the chromosome can be unserved. A non-serving arc is denoted by $a = (org_a, t, U_a)$ where U_a stands for the set of unserved customers by arc a.

There can be more than one trip arcs between origin and destination nodes depending on the number of trips and the distribution of customer nodes among trips. Including only the arc with the minimum cost between each origin and destination node would be sufficient for solving the shortest path problem optimally.

Cost of trip arc $a = \left(org_a, dest_a, v_a, \left(T_{a1}, T_{a2} \ldots, T_{ar_a}\right)\right)$ can be calculated as follows:

$$c_a = \sum_{r=1}^{r_a}\left(c^t_{0T_{ar1}v_a} + \sum_{i=1}^{|T_{ar}|-1} c^t_{T_{ari}T_{ar(i+1)}v_a}\right) + \sum_{r=1}^{r_a-1} c^t_{T_{ar|T_{ar}|}0v_a} + c^t_{T_{ara|T_{ara}|}e_{v_a}v_a} + c^f_{v_a} + c^o_{v_a}o_a + \sum_{r=1}^{r_a}\sum_{i=1}^{|T_{ar}|} c^l_i l_i + \sum_{r=1}^{r_a}\sum_{i=1}^{|T_{ar}|}\left(1-\beta_{iv_a}\right)c^v \tag{6.28}$$

$c^t_{0T_{ar1}v_a}$ is the travel cost from the depot to the first customer node in the r^{th} trip of arc a, where 0 represents the depot and T_{ari} denotes the i^{th} customer node in the r^{th} trip of arc a. Similarly, $c^t_{T_{ari}T_{ar(i+1)}v_a}$ is the travel cost from the i^{th} customer node to the $(i+1)^{th}$ customer node in the r^{th} trip of arc a. At the end of every trip except for the last one ($r_a{}^{th}$ trip), the vehicle is assumed to return to the depot. The cost associated with this travel is represented by $c^t_{T_{ar|T_{ar}|}0v_a}$. On the other hand, after its last trip, the vehicle returns to a specific final location, e_{v_a}. Cost of this trip is shown by $c^t_{T_{ara|T_{ara}|}e_{v_a}v_a}$. Fixed vehicle cost is also incorporated into the arc cost, which is denoted by $c^f_{v_a}$. Overtime (o_a) and late arrivals for each customer (l_i) are calculated for each arc. The respective cost terms are $c^o_{v_a}o_a$ for overtime and $c^l_i l_i$ for late arrivals. β_{iv_a} is the binary parameter which takes the value of one if customer i can be served by vehicle v_a and zero, otherwise. We

penalize assignments violating the accessibility restrictions with a cost term of c^v for each customer node.

Cost of vehicle-skipping arcs are zero whereas cost of a non-serving arc $a = (org_a, t, U_a)$ is calculated as $c_a = \sum_{i \in U_a} c_i^u$ where c_i^u is the cost of not visiting customer i.

Let us recall the example chromosome $C = \big((1,2,3,4,5,6,7,8,9,10),(A,B,C)\big)$. Assume that we have three different split solutions which are given as:

$$S_1 = \Big(\big(A,((1,2,3))\big),\big(B,((4,5))\big),\big(C,(6,7,8)\big)\Big),\{9,10\}\Big) \tag{6.29}$$

$$S_2 = \Big(\big(A,((1,2,3),(4,5))\big),\big(B,((6,7,8))\big),\big(C,()\big)\Big),\{9,10\}\Big) \tag{6.30}$$

$$S_3 = \Big(\big(A,((1,2,3),(4,5,6))\big),\big(B,((7,8))\big),\big(C,(9,10)\big)\Big),\{\}\Big) \tag{6.31}$$

Each solution can be represented as a path from s to t in the problem network. S_1, S_2 and S_3 are illustrated as blue, red and green paths in Figure 1, respectively.

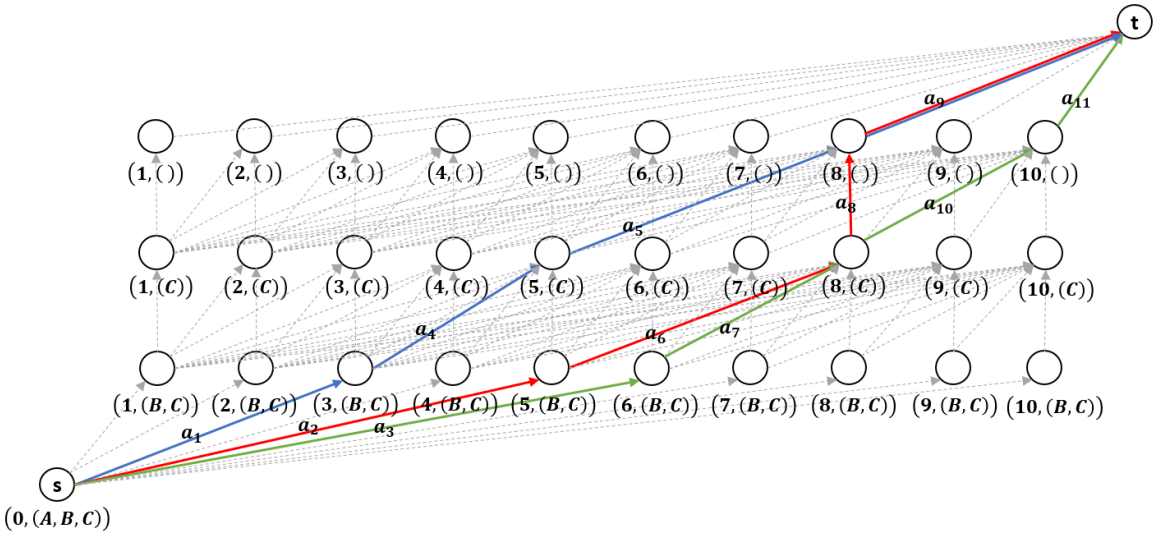


Figure 6.1. Example Solutions to Splitting Problem.

In the example provided, arcs $a_1 - a_7$ and a_{10} represent trip arcs, whereas a_8 is a vehicle-skipping arc and a_9 and a_{11} are non-serving arcs.

Calculating arc costs is computationally difficult and generating the whole problem network would be time consuming. Therefore, we propose a heuristic approach for the splitting problem in which only arcs that satisfy a minimum capacity utilization (ξ) for

each trip are considered. We also exclude the arcs in which the vehicle capacity is exceeded in order to maintain the feasibility of the solution.

Pseudocode of split algorithm is provided in Algorithm 6.3. The approach we take is similar to Dijkstra's shortest path algorithm. The main difference is that rather than constructing the whole problem network at the beginning, we add new nodes to the network as we explore the arcs that satisfy the conditions we defined.

Throughout the algorithm, we maintain a list called *process*, which includes the nodes to be processed. *cost(n)* represents the minimum total cost of arcs up to node n. *pred(n)* shows the predecessor of node n.

Algorithm 6.3. Split Algorithm

Initialization: $s := (0, C_{vehicles})$ *process* $:= \{s\}$, *cost(s)* $:= 0$, *cost(t)* $:= \infty$, *pred(s)* $\leftarrow$ *Null*
While process is not empty do:
    Select the first node n *from process and remove it from the list*
    If $|V_n| > 0$*:*
        For $r = 1$ *through* $|R|$*:*
            Create feasible trip arcs emanating from node n *with* r *trips*
            For each arc a *created:*
                If $dest_a$ *has not been created before:*
                    V_{dest_a}*: All elements of* V_n *except the first vehicle*
                    $dest_a := (I_{dest_a}, V_{dest_a})$
                    pred($dest_a$*)* $:= n$
                    cost($dest_a$*)* $:=$ *cost(n)* $+ c_a$
                    Append $dest_a$ *to process*
                Else if cost(n) $+ c_a <$ *cost(*$dest_a$*):*
                    pred($dest_a$*)* $:= n$
                    cost($dest_a$*)* $:=$ *cost(n)* $+ c_a$
    Else:
        $U_a = \{m \in C_{nodes}: I_m > I_n\}$
        Create non-serving arc $a = (n, t, U_a)$
        If cost(n) $+ c_a <$ *cost(t):*
            pred(t) $:= n$
            cost(t) $:=$ *cost(n)* $+ c_a$
Obtain solution S using predecessor information going from t *to* s
Return S

6.2.4 ALNS Operators

ALNS algorithm relies on the use of several removal and insertion heuristics, which are also known as destroy-repair or ruin-recreate operators. Using more than one heuristics increase the robustness of the methods and ease the handling of various types of problem characteristics.

We use ALNS operators in the mutation and boosting phases of our hybrid evolutionary algorithm. We employ six destroy and four repair operators and their variations.

Destroy Operators

Destroy operators remove a number of customers from the solution at each iteration. The operators used within our method are violation of accessibility restriction removal (VARR), random removal (RR), worst removal (WR), modified distance removal (MDR), largest earliness removal (LER) and largest lateness removal (LLR).

By applying a destroy operator to a solution, we obtain a new solution where the removed customers are added to the set of unserved customers. Recall the example solution from the previous section:

$$S = \Big(\big(\big(A, \big((1,2,3),(4,5)\big)\big), \big(B, \big((6,7,8)\big)\big), \big(C, (\,)\big)\big), \{9,10\}\Big) \qquad (6.32)$$

If we remove customers 2, 4 and 8 from S, we obtain a new solution S^*:

$$S^* = \Big(\big(\big(A, \big((1,3),(5)\big)\big), \big(B, \big((6,7)\big)\big), \big(C, (\,)\big)\big), \{2,4,8,9,10\}\Big) \qquad (6.33)$$

The change in the total fitness value is simply $f_{S^*} - f_S$. Next, let us describe the destroy operators in detail.

Violation of Accessibility Restriction Removal (VARR):

Split algorithm does not guarantee that the customers are always served by their allowed vehicle types. Therefore, we remove the customers that violate accessibility restrictions from the solution in order to make the solution feasible at the beginning of the mutation procedure.

Random Removal (RR):

The RR operator simply selects D% of the customer nodes that are assigned to a trip and makes them unserved.

Worst Removal (WR):

The aim of the WR operator is to remove customers that are most expensive to keep in their current trips. The total cost of a vehicle route, which may consist of one or more trips, is calculated in the same way as the cost of trip arcs in the split algorithm. Given a route $T_i = \left(v_i, \left(T_{i1}, T_{i2} \ \ldots, T_{ir_i}\right)\right)$ performed by vehicle v_i, the total cost of route (c_{T_i}) can be found as:

$$c_{T_i} = \sum_{r=1}^{r_i} \left(c^t_{0T_{ir1}v_i} + \sum_{j=1}^{|T_{ir}|-1} c^t_{T_{irj}T_{ir(j+1)}v_i} \right) + \sum_{r=1}^{r_i-1} c^t_{T_{ir}|T_{ir}|0v_i} + c^t_{T_{ir_i}|T_{ir_i}|e_{v_i}v_i} + c^f_{v_i} + c^o_{v_i} o_i + \sum_{r=1}^{r_i} \sum_{j=1}^{|T_{ir}|} c^l_j l_j + \sum_{r=1}^{r_i} \sum_{j=1}^{|T_{ir}|} \left(1 - \beta_{jv_i}\right) c^v \tag{6.34}$$

By removing a customer from its trip, we obtain a new route T_i^* with the cost of $c_{T_i^*}$. The WR operator removes D% of the nodes from their trips iteratively by selecting the one of which removal results in the maximum reduction in the route cost.

Modified Distance Removal (MDR):

The MDR operator makes use of the modified distance between two customer nodes, which is denoted by δ_{ij} for nodes i and j. The MDR procedure starts with the removal of a random customer from its trip. Iteratively, the customer having the lowest modified distance with the last removed customer is selected and pulled out from its trip until D% of the nodes are removed.

Largest Earliness Removal (LER):

The LER operator aims to remove the nodes which cause the vehicles to wait the most due to arriving earlier than the time window start times. p_i denotes the arrival time of vehicle to the customer whereas a_i is the earliest time to start service for node i. Nodes

are sorted in descending order based on the value of $a_i - p_i$ and up to D% of the customers are removed from the top to bottom provided that $a_i - p_i > 0$.

Largest Lateness Removal (LLR):

The purpose of the LLR operator is to remove nodes which cause the utmost violation of the latest arrival time. b_i stands for the latest arrival time for node i. Similar to LER operator, we rank the nodes based on the value of $p_i - b_i$ and remove up to D% of them, which satisfy $p_i - b_i > 0$, from their trips.

Repair Operators

Repair operators are used for inserting unserved customers into trips. We employ four different types of repair operators, namely, greedy insertion (GI), greedy insertion with noise (GIN), greedy insertion with random selection (GIRS) and regret insertion (RI).

The feasibility of the solution in terms of vehicle capacity and accessibility restrictions is maintained in all insertion operations.

After each insertion, the change in the solution cost is calculated for each cost component. $\Delta c^t, \Delta c^f, \Delta c^o, \Delta c^l$ and Δc^u stand for the change in the travel cost, fixed vehicle cost, overtime cost, late arrival cost and non-serving cost, respectively. Assume that node n is to be inserted at the s^{th} position in the r^{th} trip of vehicle route $T_i = \left(v_i, \left(T_{i1}, T_{i2} \ \ldots, T_{ir_i}\right)\right)$.

If node n will be inserted into a trip with existing customers, Δc^t will be calculated as follows:

$$\Delta c^t = \begin{cases} c^t_{0nv_i} + c^t_{nT_{ir1}v_i} - c^t_{0T_{ir1}v_i}, & if\ s = 1 \\ c^t_{T_{ir(s-1)}nv_i} + c^t_{nT_{irs}v_i} - c^t_{T_{ir(s-1)}T_{irs}v_i}, & if\ 1 < s \leq |\,T_{ir}| \\ c^t_{T_{ir|\,T_{ir}|}nv_i} + c^t_{n0v_i} - c^t_{T_{ir|\,T_{ir}|}0v_i}, & if\ s = |\,T_{ir}| + 1\ and\ r < r_i \\ c^t_{T_{ir|\,T_{ir}|}nv_i} + c^t_{ne_{v_i}v_i} - c^t_{T_{ir|\,T_{ir}|}e_{v_i}v_i}, & if\ s = |\,T_{ir}| + 1\ and\ r = r_i \end{cases} \quad (6.35)$$

If n will be the only node in the trip it is inserted, Δc^t is calculated as follows:

$$\Delta c^t = \begin{cases} c^t_{0nv_i} + c^t_{n0v_i}, & if\ r \leq r_i \\ c^t_{0nv_i} + c^t_{ne_{v_i}v_i} + c^t_{T_{ir|\,T_{ir}|}0v_i} - c^t_{T_{ir|\,T_{ir}|}e_{v_i}v_i}, & if\ r = r_i + 1 \end{cases} \quad (6.36)$$

The change in the fixed vehicle cost, Δc^f is equal to $c^f_{v_i}$ if vehicle v_i has not been used before the insertion and zero, otherwise.

Let the set of nodes in T_i that will be visited later than n after its insertion be denoted by $N_{>n}$. We calculate the change in the arrival times for each node $j \in N_{>n}$, denoted by Δp_j, as well as the change in the route completion time Δp_{T_i}. Δc^o and Δc^l can be then calculated as:

$$\Delta c^o = max\{0, c^o(p_{T_i} + \Delta p_{T_i} - d_{T_i})\} \quad (6.37)$$

$$\Delta c^l = \sum_{j \in N_{>n}} max\{0, c^l_i(p_j + \Delta p_j - b_j)\} \quad (6.38)$$

Since customer n will no longer be unserved after it is inserted into a trip, $\Delta c^u = -c^u_n$.

The total change in the solution cost, Δc is simply equal to $\Delta c^t + \Delta c^f + \Delta c^o + \Delta c^l + \Delta c^u$. If the value of Δc^u is found to be positive for some node, then there is no need for inserting it into the trip since keeping it unserved is less costly. The detailed description of repair operators is provided as follows:

Greedy Insertion (GI):

The purpose of GI operator is to find the best insertion location for all unserved customers. The value of Δc is calculated for all nodes in U for each feasible insertion position in the solution $S = (T, U)$. The node with the minimum insertion cost is removed from U and placed in its best position. This procedure is repeated iteratively until either no further feasible insertion is possible or the minimum insertion cost is found to be positive.

Greedy Insertion with Noise (GIN):

GIN is similar to GI and the only difference is that a noise term is added to Δc in order to allow diversification in the solutions. Let η be the noise coefficient and c^t_{max} denote the maximum possible travel cost among any two customers where $c^t_{max} = \max_{(i,j) \in A, k \in V} c^t_{ijk}$. We generate a random number ε between the interval $[-1,1]$. The noised cost change (Δc^η) is calculated as $\Delta c^\eta = \Delta c + \eta \varepsilon c^t_{max}$. The rest of the method is the same as the GI operator.

Greedy Insertion with Random Selection (GIRS):

In GIRS method, rather than finding the best insertion position for all unserved nodes, we randomly pick a node from U and insert it into the position with the minimum cost at each iteration. This operator is also expected to increase the level of diversification.

Regret Insertion (RI):

RI operator aims to select customers for insertion based on the difference between the cost of inserting them to their best and second best positions. This means that, we will regret the most if the customer with the largest difference is not inserted in its best position. Assume that Δc_1 and Δc_2 represent the change in the solution cost for the best and the second best insertions of a node. We calculate $\Delta c_2 - \Delta c_1$ for all nodes in U. We pick the node with the highest $\Delta c_2 - \Delta c_1$ value and insert it in its best position. This continues iteratively until all nodes in U are evaluated.

6.2.5 Mutation

Once a chromosome C goes through the split algorithm, we obtain a solution S, which is usually far from optimum. The purpose of the mutation process is to alter solution S iteratively applying ALNS operators and obtain a better solution at the end. We use a simulated annealing approach to allow worsening operations more often at the earlier stages of the algorithm for increasing the level of diversification. The solution undergoing a mutation is expected to converge to a local optimum through the end of the algorithm.

Mutation is applied both in the initial solution construction and the improvement phases of the hybrid evolutionary algorithm, with the iteration limits κ_{init} and κ_{impr}. Once the split solution is achieved, VARR operator is applied first, in order to make the solution feasible. This operator is no longer used throughout the mutation process, since the other ALNS operators maintain the feasibility of the solutions.

Destroy and repair operators have initially assigned weights, denoted by ω_{m_d} and ω_{m_r}, respectively where $m_d \in DO$, $DO = \{RR, WR, MDR, LER, LLR\}$ and $m_r \in RO$, $RO = \{GI, GIN, GIRS, RI\}$. At each iteration of the algorithm, destroy and repair operators m_d and m_r are selected with probabilities ρ_{m_d} and ρ_{m_r}, where $\rho_{m_d} = \omega_{m_d} / \sum_{i \in DO} \omega_i$ and $\rho_{m_r} = \omega_{m_r} / \sum_{i \in RO} \omega_i$.

We define five different notations for solutions: S_{init}, S_{act}, S_{dest}, S_{cand} and S_{best}. S_{init} denotes an initial solution obtained by splitting. S_{act} is the active solution in the algorithm. S_{act} represents the destroyed solution achieved by applying a destroy operator to S_{act}. S_{cand} is the candidate solution that is being evaluated for acceptance and finally, S_{best} stands for the solution with the lowest fitness value ($f_{S_{best}}$), since the fitness represents the total cost of our solution.

The parameters related with the simulated annealing framework are τ_{start}, τ_{cur} and θ, which denote starting temperature, current temperature and cooling rate. We define acceptance probability function $AP\left(f_{S_{cand}}, f_{S_{best}}, \tau_{cur}\right)$, which returns the probability of solution S_{cand} to be accepted given that the best solution found so far is S_{best} and the current temperature is τ_{cur}. This function is formulated as:

$$AP\left(f_{S_{cand}}, f_{S_{best}}, \tau_{cur}\right) = \begin{cases} e^{-\left(f_{S_{cand}} - f_{S_{best}}\right)/\tau_{cur}}, & if\ f_{S_{cand}} > f_{S_{best}} \\ 1, & otherwise \end{cases} \quad (6.39)$$

We also define the function $RAND(0,1)$ which returns a random number in the interval [0,1]. We denote the iteration limit with the generalized parameter κ, which is equal to κ_{init} if the mutation is applied in the initial solution construction phase and κ_{impr} if it is applied in the improvement phase. The mutation process is presented in Algorithm 6.4.

Algorithm 6.4. Mutation Process

1: *Initialization:* $\tau_{cur} := \tau_{start}$, $S_{act} := S_{init}$, $S_{best} := S_{init}$, $\rho_{m_d} = \omega_{m_d} / \sum_{i \in DO} \omega_i$ *for* $m_d \in DO$ *and* $\rho_{m_r} = \omega_{m_r} / \sum_{i \in RO} \omega_i$ *for* $m_r \in RO$
2: *For* $i = 1$ *through* κ*:*
3: *Select destroy operator* $m_d \in DO$ *by roulette-wheel selection with probability* ρ_{m_d}
4: *Apply destroy operator* m_d *to* S_{act} *and obtain* S_{dest}
5: *Select repair operator* $m_r \in RO$ *by roulette-wheel selection with probability* ρ_{m_r}
6: *Apply repair operator* m_r *to* S_{dest} *and obtain* S_{cand}
7: *If* $AP\left(f_{S_{cand}}, f_{S_{best}}, \tau_{cur}\right) \geq RAND(0,1)$*:*
8: $S_{act} \leftarrow S_{cand}$
9: *If* $f_{S_{cand}} < f_{S_{best}}$*:*
10: $S_{best} := S_{cand}$
11: $\tau_{cur} := \tau_{cur} * \theta$
12: *Return* S_{best}

6.2.6 Crossover

Crossover process is employed to create child chromosomes from parent solutions during the improvement phase of the hybrid evolutionary algorithm. The first phase of crossover is the selection of two parents out of the population. The parents are selected through a modified binary tournament. We randomly select two candidate solutions for each parent, and the probability of being selected is inversely proportional to the fitness level of each solution. Assume that S_A and S_B are two randomly selected solutions to be selected as one of the parents for the crossover operation. The probability of selecting S_A is $\left(1/f_{S_A}\right)/\left(1/f_{S_A}+1/f_{S_B}\right)$ whereas the probability of selecting S_B is $\left(1/f_{S_B}\right)/\left(1/f_{S_A}+1/f_{S_B}\right)$.

Recall that a chromosome is represented as a permutation of nodes and vehicles. After the parents are selected, the solutions are converted into chromosomes using the ordering of nodes and vehicles in the solution. Consider the following example solution:

$$S=\left(\left(\left(B,\left((5,2,6),(9,4)\right)\right),\left(C,\left((3,7,10)\right)\right),\left(A,(\,)\right)\right),\{1,8\}\right) \qquad (6.40)$$

After the conversion, we obtain chromosome $C=\left((5,2,6,9,4,3,7,10,1,8),(B,C,A)\right)$ for solution S. We apply ordered crossover to selected two parents' node permutations (C_{nodes}). Firstly, we randomly select two indices i and j between 1 and $|C_{nodes}|$ where $i<j$. The subsequence with indices $(i,\dots,j)$ of the first parent is copied to the first child whereas that of the second parent is copied to the second child. For the first child, empty positions $(1,\dots,i-1)$ and $(j+1,\dots,|C_{nodes}|)$ are filled in the order of second parents' nodes, skipping the ones already included between indices i and j. The same procedure is applied to second child using the node permutation of the first parent.

The permutation of vehicles ($C_{vehicles}$) is either directly transferred from the first parent to the first child and from the second parent to the second child, or it is shuffled and a random sequence is used in the child chromosomes, which is useful for increasing diversification. Both options have 50% selection probability for each crossover operation.

The newly created child chromosomes undergo split and mutation processes, and the one having the lower fitness value is added to the population.

6.2.7 Operator Weight Adjustment

Weights of the operators are updated at the end of each iteration of the improvement phase of the hybrid evolutionary algorithm based on their past success. During the mutation process, each ALNS operator gains a score of π_1, if it finds a new best solution, π_2, if it improves the active solution but fails to find a new best solution and π_3, if it fails to improve the active solution but the new solution is accepted. The cumulative score obtained since the beginning of the improvement phase is denoted by Π_m for ALNS operator m. We also keep track of the number of times each operator is applied, which is denoted by Γ_m. The weight adjustment coefficient λ determines the degree to which the scores gathered will affect the weights of the operators. The weight adjustment procedure works as follows for each ALNS operator: $\omega_m \leftarrow (1-\lambda)\omega_m + \lambda(\Pi_m/\Gamma_m)$.

6.2.8 Survivor Selection

At the end of each iteration of the improvement phase, we need to determine the n_{pop} individuals that will be transferred to the next generation. In order to keep a certain portion of the best solutions, we sort the solutions in ascending order by their fitness values. Any duplicate solutions are removed from the list and only distinct solutions are taken into consideration. Top $\alpha * n_{pop}$ individuals are selected for the next generation where α is a number between 0 and 1. The rest of the individuals are selected randomly from the rest of the list.

6.2.9 Boosting Algorithm

In order to improve promising solutions further, top n_{boost} distinct solutions obtained in the improvement phase undergo an ALNS algorithm in parallel. Boosting algorithm is similar to mutation process and the difference is that operator weights are updated at every κ_{upd} iterations. The number of iterations of boosting algorithm (κ_{boost}) is higher than that of the mutation process.

The pseudocode of the boosting algorithm is presented in Algorithm 6.5. Note that S_{init} is the solution found in the improvement phase and the weights of the operators are reset to their initial values before the boosting procedure.

Algorithm 6.5. Boosting Procedure

1: *Initialization:* $\tau_{cur} := \tau_{start}$, $S_{act} := S_{init}$, $S_{best} := S_{init}$, $\rho_{m_d} = \omega_{m_d} / \sum_{i \in DO} \omega_i$, $\Pi_{m_d} = 0$ *and* $\Gamma_{m_d} = 0$ *for* $m_d \in DO$ *and* $\rho_{m_r} = \omega_{m_r} / \sum_{i \in RO} \omega_i$, $\Pi_{m_r} = 0$ *and* $\Gamma_{m_r} = 0$ *for* $m_r \in RO$

2: *For* $i = 1$ *through* κ_{boost}*:*

3: *Select destroy operator* $m_d \in DO$ *by roulette-wheel selection with probability* ρ_{m_d}

4: *Apply destroy operator* m_d *to* S_{act} *and obtain* S_{dest}

5: *Select repair operator* $m_r \in RO$ *by roulette-wheel selection with probability* ρ_{m_r}

6: *Apply repair operator* m_r *to* S_{dest} *and obtain* S_{cand}

7: $\Gamma_{m_d} := \Gamma_{m_d} + 1$, $\Gamma_{m_r} := \Gamma_{m_r} + 1$

8: *If* $AP\left(f_{S_{cand}}, f_{S_{best}}, \tau_{cur}\right) \geq RAND(0,1)$*:*

9: $S_{act} := S_{cand}$

10: *If* $f_{S_{cand}} < f_{S_{best}}$*:*

11: $S_{best} := S_{cand}$, $\Pi_{m_d} := \Pi_{m_d} + \pi_1$, $\Pi_{m_r} := \Pi_{m_r} + \pi_1$

12: *Else if* $f_{S_{cand}} < f_{S_{act}}$*:*

13: $\Pi_{m_d} := \Pi_{m_d} + \pi_2$, $\Pi_{m_r} := \Pi_{m_r} + \pi_2$

14: *Else:*

15: $\Pi_{m_d} := \Pi_{m_d} + \pi_3$, $\Pi_{m_r} := \Pi_{m_r} + \pi_3$

16: $\tau_{cur} \leftarrow \tau_{cur} * \theta$

17: *If* $\left(i \bmod \kappa_{upd}\right) = 0$*:*

18: $\omega_m := (1 - \lambda)\omega_m + \lambda(\Pi_m / \Gamma_m)$ *for* $m \in DO \cup RO$

$\rho_{m_d} = \omega_{m_d} / \sum_{i \in DO} \omega_i$, $\rho_{m_r} = \omega_{m_r} / \sum_{i \in RO} \omega_i$

19: *Return* S_{best}

At the end of this phase, among the n_{boost} boosted solutions, the one with the lowest fitness value is reported as the final solution of the HEA.

6.3 *Computational Experiments for the Last-mile Delivery Planning Problem*

We conduct three sets of experiments to test the HEA for the LMDP. In the first experiments we compare the results of our algorithm against CPLEX results for small instances. These instances contain all the characteristics of the LMDP. Afterwards, we test algorithm for a special case of LMDP on benchmark instances extracted from the literature. The coded our algorithms in Python and use CPLEX 12.8 to solve the

mathematical models. Experiments were run on a 24-core workstation with 2.40 GHz processors and 128 GB RAM.

6.3.1 Experiments on Small Instances

Firstly, we test our algorithm against the exact MILP solution for instances generated based on the sampling from actual customer locations and real vehicle data of the 3PL company. Each instance has 20 customer nodes. We define five different instance types, each having a different set of customer nodes. For each instance type, we create two instances with low and high customer demands. In low-demand instances, $q_i \sim U(0,3)$ whereas in high demand instances, $q_i \sim U(3,10)$ for $i \in N$.

We assume that the same set of vehicles is available for each instance. Let parameter c_k^{km} denote the fuel cost per kilometer for vehicle k. The information regarding the vehicles used in the computational experiments are provided in Table 6.1. Note that each vehicle has a different final location to return after completing their routes.

Table 6.1. Values of Vehicle Parameters.

Vehicle (k)	c_k^f	c_k^o	c_k^{km}	Q_k
1	119	1000	0.976	18.56
2	233	1000	1.098	17.34
3	233	1000	1.098	15.52
4	233	1000	1.586	35.69
5	233	1000	1.586	37.92

For all 5 instance types, there are 100 customer nodes in total. Figure 6.2 shows the number of customers that can be served by each vehicle due to accessibility restrictions.

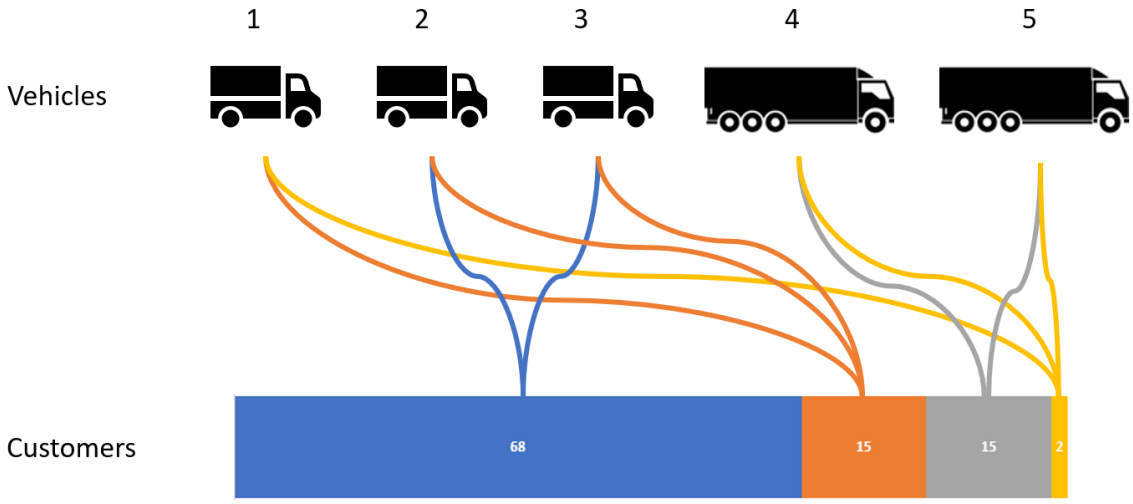


Figure 6.2. Number of Customers That Can Be Served by Each Vehicle.

The values of other problem parameters used in the experiments are given as follows:

Table 6.2. Other Parameter Values.

Parameter	***Value***
$\boldsymbol{c_i^l}$	$500, \quad i \in N$
$\boldsymbol{c_i^u}$	$5000q_i, \quad i \in N$
$\boldsymbol{\sigma_k}$	$1, \quad k \in V$
$\boldsymbol{d_k}$	$12.5, \quad k \in V$

Realistic travel times and driving distances are obtained from Google Maps API. Service times (s_i) and time windows ($[a_i, b_i]$) are provided in customer data. Figures 6.3 and 6.4 illustrate the distribution of service times and time windows among customers.

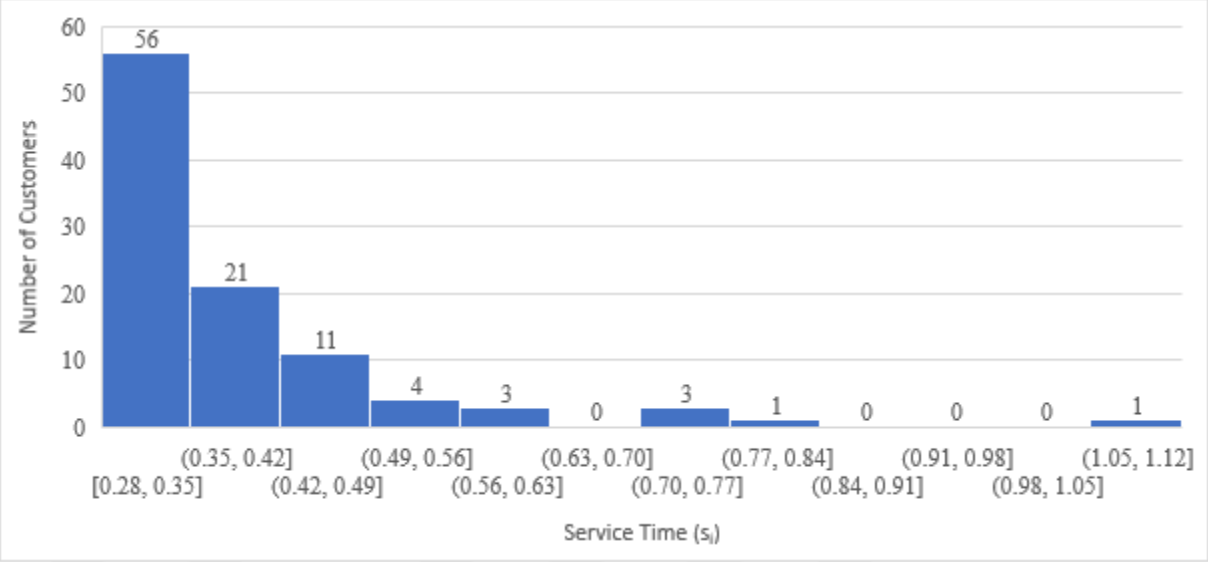


Figure 6.3. Distribution of Service Times.

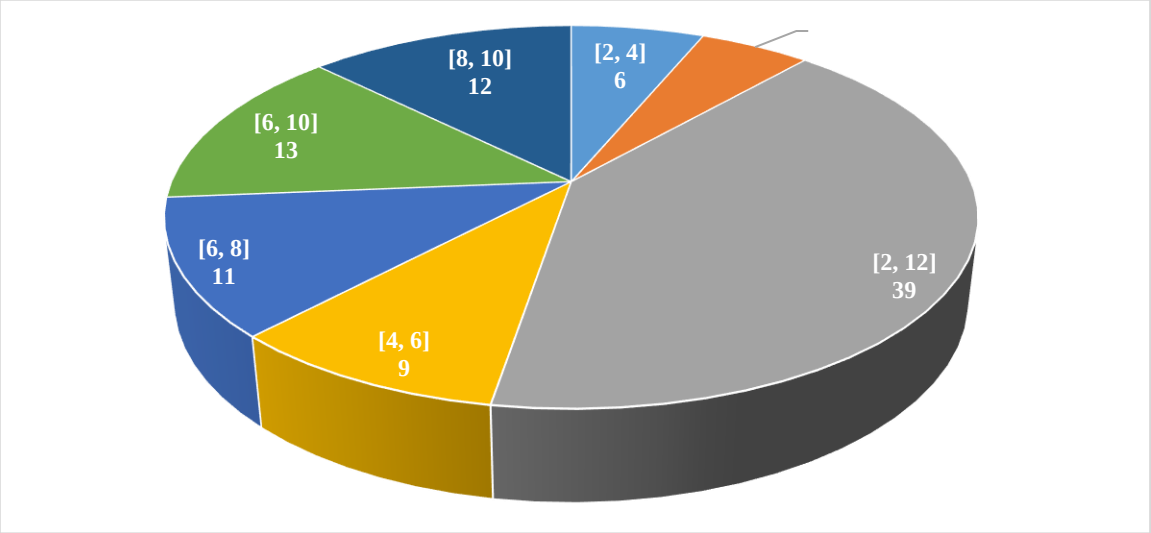


Figure 6.4. Distribution of Time Windows.

The list of parameters used within the hybrid evolutionary algorithm, their descriptions and values are given in Table 6.3.

Table 6.3. HEA Parameters and their Values.

Parameter	Description	Value
n_{pop}	Number of individuals in a population	28
$n_{process}$	Number of parallel processes used for computation	28
n_{gen}	Number of generations in the improvement phase	10
n_{cross}	Number of crossover operations to be applied at each iteration	28

$\boldsymbol{n_{boost}}$	Number of solutions to be boosted	28
$\boldsymbol{n_{top}}$	Number of candidate nodes for selection in initial chromosome construction	10
$\boldsymbol{\mu_1}$	Coefficient for allowed vehicle difference. Used in modified distance calculation	3
$\boldsymbol{\mu_2}$	Coefficient for time window difference. Used in modified distance calculation	1.5
$\boldsymbol{\xi}$	Minimum capacity utilization per trip in split algorithm	80%
$\boldsymbol{c_i^v}$	Penalty coefficient for violation of accessibility restrictions	100
D	Percentage of nodes to be removed from their trips	D ~ U(0.05, 0.25)
$\boldsymbol{\eta}$	Noise coefficient	0.1
$\boldsymbol{\kappa_{init}}$	Number of destroy-repair operations in initial solution phase	250
$\boldsymbol{\kappa_{impr}}$	Number of destroy-repair operations in improvement phase	1000
$\boldsymbol{\kappa_{boost}}$	Number of destroy-repair operations in boosting phase	5000
$\boldsymbol{\kappa_{upd}}$	Number of iterations required for updating operator weights	500
$\boldsymbol{\tau_{start}}$	Starting temperature for simulated annealing	$\frac{-0.5 f_{S_{init}}}{\ln(0.5)}$
$\boldsymbol{\theta}$	Cooling rate	$\sqrt[\kappa]{\frac{0.1}{\tau_{start}}}$
$\boldsymbol{\pi_1}$	Increase in operator score if new best solution is found	5
$\boldsymbol{\pi_2}$	Increase in operator score if the active solution is improved	3
$\boldsymbol{\pi_3}$	Increase in operator score if the new solution is accepted	1
$\boldsymbol{\alpha}$	Portion of best solutions to be transferred to boosting	0.3

Note that the initial weight of all ALNS operators are equal to 1. The maximum number of trips that can be performed by a vehicle is equal to 2. The MILP models are solved by CPLEX 12.8 with a time limit of 3 hours (10,800 seconds). HEA is executed in a 28-core machine with parallel threading. The results of experiments for 10 instances are reported in Table 6.4. f^*_{MILP} and f^*_{HEA} denote the best solution values reported by the solver and the hybrid evolutionary algorithm, respectively. CPU times (in seconds) are shown by T_{MILP} and T_{HEA}. $\%\Delta_f$ represents the percentage difference between the best solution values.

The MILP model was solved optimally in 7 out of 10 instances. Our algorithm found the optimal solution in 4 of these instances while the percentage difference between solution costs is below 0.1% for the remaining 3 instances. The solver was not able to find the optimal solution in 3 instances. Better solutions are reported by HEA for 2 of these instances whereas the same solution is found by both methods in 1 instance.

In terms of CPU time, HEA performs significantly better than the exact solution approach in low-demand instances. However, HEA yields higher solution times in high-demand instances compared to MILP solutions. The main reason behind this the high number of iterations employed within the algorithm. Further experiments have shown that the same results could have been achieved in much quicker with less number of iterations.

Table 6.4. MILP vs. HEA Results.

Instance Type	**Demand Type**	f^*_{MILP}	**MILP GAP**	T_{MILP}	f^*_{HEA}	T_{HEA}	$\%\Delta_f$
1	Low	1140.9	1.5%	10814.3	**1140.9**	42.0	0.0%
2	Low	1256.7	1.9%	10811.6	**1251.6**	43.8	-0.4%
3	Low	1060.4	3.5%	10808.7	**1058.9**	42.0	-0.1%
4	Low	977.4	0.0%	141.8	977.5	41.3	0.0%
5	Low	843.5	0.0%	4753.3	**843.5**	44.1	0.0%
1	High	40920.8	0.0%	3.7	**40920.8**	24.1	0.0%
2	High	189291.8	0.0%	5.3	189328.7	21.2	0.0%
3	High	115524.0	0.0%	2.8	115525.2	22.8	0.0%
4	High	87512.0	0.0%	9.6	**87512.0**	24.1	0.0%
5	High	46433.6	0.0%	3.9	**46433.6**	25.2	0.0%

Overall, our algorithm was able to detect the same or better solutions than the solver in 7 out of 10 instances and the differences are significantly small in the remaining 3 instances. Our algorithm is also competitive in terms of solution time, especially in instances which are difficult to solve.

6.3.2 Experiments on Literature Benchmark Instances

The variant of the VRP we address has not been studied in the literature before with all the characteristics involved. Therefore, we test our algorithm on benchmark instances for *Heterogeneous Fleet Vehicle Routing Problem with Time Windows* (HVRPTW), which is a special case of MTRVRP with single trips and no accessibility restrictions. Although

the time window constraints are hard, overtimes are not allowed and all customers need to be served for the provided instances, we run our algorithm without any changes by giving high penalties for violation of these constraints.

We use the benchmark instances provided by Koç et al. [92], who also applied a hybrid evolutionary approach to address HVRPTW. The instances are the extended versions of well-known Solomon [165] instances with multiple vehicle types. We compare or results against so-called HD instances, which represent HVRPTW problems where a distance-based objective is employed and the fleet size is assumed to be fixed. We use the same mix of vehicle types for each instance as the ones used by Koç et al. [92]. There are 25 instances in total each having 100 customer nodes.

The comparative results are tabulated in Table 6.5. Solution cost and time of Koç et al. [92] are denoted as f^*_{KOC} and T_{KOC}, respectively. Solution times are reported in minutes. The percentage difference ($\%\Delta_f$) is calculated by $(f^*_{HEA} - f^*_{KOC})/f^*_{HEA}$.

Table 6.5. Literature Benchmark Comparison.

Instance	f^*_{KOC}	T_{KOC}	f^*_{HEA}	T_{HEA}	$\%\Delta_f$
R101	4355.41	5.19	4455.56	11.10	2.25%
R102	4356.44	6.24	4466.73	11.07	2.47%
R103	4080.16	6.57	4324.57	7.41	5.65%
R104	3954.72	5.89	4416.65	6.00	10.46%
C101	**8828.94**	**4.25**	**8828.94**	**20.00**	**0.00%**
C102	7080.17	3.97	7157.37	11.93	1.08%
C103	7079.21	3.99	7228.45	11.58	2.06%
C104	7075.06	2.98	7240.45	10.26	2.28%
RC101	5162.28	6.41	5385.28	14.13	4.14%
RC102	5018.05	5.24	5275.04	11.89	4.87%
RC103	4926.55	4.39	5329.37	8.39	7.56%
RC104	4995.91	4.88	5354.62	6.98	6.70%
R201	3448.76	6.74	3484.70	32.54	1.03%
R202	3308.16	8.13	3370.49	31.00	1.85%
R203	3382.39	7.49	3401.92	33.22	0.57%
R204	3018.14	5.47	3068.34	26.56	1.64%
C201	**6082.38**	**4.21**	**6082.38**	**33.79**	**0.00%**
C202	**7618.62**	**3.69**	**7618.62**	**39.02**	**0.00%**
C203	**7303.37**	**3.67**	**7303.37**	**37.20**	**0.00%**
C204	**5677.66**	**5.11**	**5678.45**	**29.65**	**0.01%**
RC201	**5344.47**	**6.72**	**5296.61**	**34.23**	**-0.90%**
RC202	4856.02	6.48	4895.23	34.41	0.80%

RC203	**4246.25**	**6.93**	**4242.63**	**24.77**	**-0.09%**
RC204	4195.32	6.17	4441.58	10.73	5.54%
Average	5224.77	5.45	5347.81	20.74	2.30%

In 5 out of 25 instances, we report the same solutions as Koç et al. [92] whereas we improve their solutions in 2 instances. The average percentage difference is 2.3%, which shows that our method is promising. Since the computations are carried out different machines, it is difficult to make a fair comparison in terms of solution times. But since the difference is significant, one can say that there is room for improvement for our method in terms of solution speed.

In addition to the presented computational experiments, we also tested HEA using the real daily order data of the 3PL company by running it in parallel to the regular manual planning approach over a 6-day period. The problem instances involved 225 orders per day on average with 12 available vehicles, each having slightly different capacities. Average run-time for HEA was around 60 seconds, while manual planning took approximately one hour. Our algorithm was able to perform over 30% better on average in terms of total kilometers traveled, using 2 vehicles less on average compared to the manual planning approach, with a 100% compliance to the problem constraints. However, we observed that the planners tend to behave prudent in preparing the route plans in order to reduce the risk of late deliveries due to unexpected incidents. Still, the gap of 30% in total travel distance seems promising terms of potential improvement that can be achieved by using HEA in daily planning of last-mile delivery orders.

This chapter finalizes the discussion on the three main problems addressed related to the road freight operations of the 3PL company. In the next chapter, we present our concluding remarks along with the statement of further research opportunities in the relevant areas.

Chapter 7

CONCLUSION

In this thesis, we analyzed the road freight operations of a 3PL service provider and developed an overall solution strategy to centralize and automate their planning processes by addressing three novel optimization problems that can be extended to other logistics carriers.

In this conclusive chapter, we give a summary of the work presented throughout the thesis and recommend future research directions in terms of enhancing the problems we dealt with and identifying new research areas.

7.1 Summary of Presented Work

In the introductory chapter, we presented our objectives and motivations for this study, which is followed by a summary of the main contributions of our work in terms of academic and business perspectives. The first part of literature review chapter was allocated to an overview of the VRP literature, in which we addressed the common extensions and solution methods proposed for this highly popular combinatorial optimization problem. The second part of the chapter presented problem variants that are relevant to the problems we deal with in this study.

Chapter 3 analyzed the system in detail by characterizing its main components, underlying network and route structures. We described the existing planning approaches employed in the 3PL company by identifying different types of planner profiles and demonstrating the factors that are taken into consideration by them while planning daily transport operations. The last section of this chapter proposed a general planning approach for the road freight operations by integrating the solution techniques developed for independent problems.

In Chapter 4, we presented two versions of the Less-Than Truckload Planning Problem (LTLP). The first version (LTLP-C) was defined on a continuous-time planning

horizon with synchronization requirements at cross-dock terminals. LTLP-C instances with up to 500 orders were solved through a decomposition-based matheuristic algorithm. Although the results were promising for small-size logistic organizations, the practical implementation of the LTLP-C was not possible due to the size of the real problems we needed to tackle. After a second round of analysis, we changed the scope of the problem and defined the *LTLP with Discrete Planning Horizon and Multi-period Considerations* (LTLP-DM), which was not only a better representation of the real LTL planning problem but also less complex compared to LTLP-C thanks to a few of simplifying assumptions. We devised a novel and more advanced matheuristic which utilized a column generation-based method to solve sub-problems of LTLP-DM, which performed very well on real problem instances by creating a potential saving of 7.8% in the long-term freight costs.

Chapter 5 represented the *Dedicated Vehicle Assignment Problem* (DVAP), which had the purpose of assigning a limited number of dedicated vehicles to LTL and FTL routes in order to reduce the overall freight costs through utilizing the own fleet of the 3PL company effectively. We proposed a constructive heuristic and exact MILP solution approach for solving the problem and we were able to show the improvement potential of our methods via computational experiments.

In Chapter 6, we addressed the *Last-mile Delivery Problem* (LMDP) which was configured as a *Multi-trip Rich Vehicle Routing Problem* (MTRVRP). In order to tackle this problem, we proposed a novel *Hybrid Evolutionary Algorithm* (HEA) which combined a *Genetic Algorithm* (GA) with the *Adaptive Large Neighborhood Search* (ALNS). HEA was implemented in a parallel fashion allowing multi-thread computation, which created a significant advantage in terms of exploring of the solution space in an efficient manner. We compared the solutions of HEA against MILP solutions for small instances and literature benchmarks for a specialized version of the MTRVRP and obtained promising results.

Overall, all the proposed problems and solution methodologies demonstrated significant potential in reduction of freight costs while the practical rules and constraints are taken into careful consideration. The objectives set at the beginning of this research were fulfilled through important contributions in terms of academic and business perspectives.

7.2 Further Research Opportunities

Although a significant effort has been devoted for identifying the problems and developing solution methodologies throughout this study, there are still numerous research possibilities in terms of building upon the work we presented and discovering new research problems in the relevant fields.

One of the most promising research directions would be implementing an integrated solution approach, similar to the one we presented in Chapter 3, to plan the overall operations in a completely centralized fashion. Due to disjoint structure of current operations and information systems of the 3PL company, it has not been possible for us to implement this throughout this study. Investigating the benefits of employing such an approach would be a proper research question.

A possible extension to LTLP would be adapting the proposed centralized heuristic approach, which has a demonstrated success in our specific case, to different types of logistics networks that incorporate distinct route types. This would bring about new algorithmic challenges researchers in terms of redesigning the solution methods and introducing new constraints to the problem.

An important rule we take into consideration when solving LTLP is that pickups always take place before deliveries. This obligation eases the truck loading and unloading operations at the cross-docking facilities and inside the trailer. Lifting this requirement would also increase the diversity of the route types and potentially lead to better solutions. However, this should be discussed in detail with logistics managers and operations personnel in order to prevent unpredicted practical implications.

Another promising further research direction for LTLP is developing a new policy for finding appropriate values of artificial cost coefficients. In this study, we used a constant value for these coefficients for all sub-problems solved on each day throughout the simulation. However, setting these values dynamically for each order depending on certain conditions or sub-problem characteristics might result in lower freight costs and better on-time delivery performance in the long run.

The most promising and immediate future research opportunity for the DVAP is developing a new method for solving larger instances of the problem in a computationally efficient manner. Column generation-based methods and metaheuristics are two of the

approaches that are likely to perform well for this problem. Another possible way to extend DVAP is to adapt it to the discrete-time horizon case in order that LTLP-DM outputs, similar to the LTLP-C outputs, can be directly converted to DVAP inputs.

Although it already creates a significant cost advantage when compared to manual methods, there is still a room for improvement for the HEA which solves the LMDP. In order to increase solution quality, new destroy and/or repair operators can be added to the mutation and boosting procedures. The level of diversification can also be expanded through an improved survivor selection mechanism.

In all the problems we addressed, we assumed that the parameters are deterministic and known. However, some of the parameters are dynamic in reality and can change from day to day. One example would be the prices for spot-hired vehicles, which can vary depending on the supply and demand in the spot market. One interesting research direction would be integrating our solution methods with predictive models to derive forecasts for the unknown parameters.

Similar to the certain problem parameters, future transportation orders can also be predicted based on the past data. A promising research direction would be re-designing LTLP and DVAP problems and respective solution methodologies in such a way that future order forecasts are taken into consideration in addition to the realized transportation requests.

BIBLIOGRAPHY

[1] CSCMP, "27th Annual State of Logistics Report," 2016.

[2] T. Nemoto and K. Tezuka, "Advantage of Third Party Logistics in Supply Chain," Hitotsubashi University, 2002.

[3] I. Group, "A guide to Turkey's transport & logistics industry," 2016.

[4] A. Nettsträter, T. Geißen, M. Witthaut, D. Ebel and J. Schoneboom, "Logistics Software Systems and Functions: An Overview of ERP, WMS, TMS and SCM Systems," in *Cloud Computing for Logistics*, Berlin, Springer, 2015, pp. 1-11.

[5] P. Helo and B. Szekely, "Logistics information systems: An analysis of software solutions for supply chain co-ordination," *Industrial Management & Data Systems*, vol. 105, no. 1, pp. 5-18, 2005.

[6] B. Golden, S. Raghavan and E. Wasil, The Vehicle Routing Problem: Latest Advances and New Challenges, New York: Springer, 2008.

[7] G. B. Dantzig and K. H. Ramser, "The Truck Dispatching Problem," *Management Science*, vol. 6, pp. 80-91, 1959.

[8] G. Laporte, "Fifty Years of Vehicle Routing," *Transportation Science*, vol. 43, no. 4, pp. 408-416, 2009.

[9] K. Braekers, K. Ramaekers and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Computers & Industrial Engineering*, vol. 99, pp. 300-313, 2016.

[10] B. Eksioglu, A. V. Vural and A. Reisman, "The vehicle routing problem: A taxonomic review," *Computers & Industrial Engineering*, vol. 57, pp. 1472-1483, 2009.

[11] J. Desrosiers, F. Soumis and M. Desrochers, "Routing with time windows by column generation," *Networks*, vol. 14, pp. 545-565, 1984.

[12] M. Solomon, "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints," *Operations Research*, vol. 35, pp. 254-265, 1987.

[13] G. Clarke and J. W. Wright, "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," *Operations Research*, vol. 12, pp. 568-581, 1964.

[14] B. Golden, A. Assad, L. Levy and F. Gheysens, "The fleet size and mix vehicle routing problem," *Computers & Operations Research*, vol. 11, no. 1, pp. 49-66, 1984.

[15] P. Toth and D. Vigo, "A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls," *European Journal of Operational Research,* vol. 113, no. 3, pp. 528-543, 1999.

[16] A. Wren and A. Holliday, "Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points," *Operational Research Quarterly,* vol. 23, no. 3, pp. 333-344, 1972.

[17] M. W. P. Savelsbergh and M. Sol, "The General Pickup and Delivery Problem," *Transportation Science,* vol. 29, no. 1, pp. 17-29, 1995.

[18] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: models and algorithms," *Annals of Operations Research,* vol. 153, p. 29–46, 2007.

[19] P. M. Francis, K. R. Smilowitz and M. Tzur, "The Period Vehicle Routing Problem and its Extensions," in *The Vehicle Routing Problem: Latest Advances and New Challenges*, Boston, MA, Springer, 2008, pp. 73-102.

[20] J. C. S. Brandão and A. Mercer, "The multi-trip vehicle routing problem," *Journal of the Operational Research Society,* vol. 49, no. 8, pp. 799-805, 1998.

[21] É. D. Taillard, G. Laporte and M. Gendreau, "Vehicle Routeing with Multiple Use of Vehicles," *The Journal of the Operational Research Society,* vol. 47, no. 8, pp. 1065-1070, 1996.

[22] S. Erdoğan and E. Miller-Hooks, "A Green Vehicle Routing Problem," *Transportation Research Part E: Logistics and Transportation Review,* vol. 48, no. 1, pp. 100-114, 2012.

[23] C. Lin, K. L. Choy, G. T. S. Ho, S. H. Chung and H. Y. Lam, "Survey of Green Vehicle Routing Problem: Past and future trends," *Expert Systems with Applications,* vol. 41, no. 4, pp. 1118-1138, 2014.

[24] T. Bektaş and G. Laporte, "The Pollution-Routing Problem," *Transportation Research Part B: Methodological,* vol. 45, no. 8, pp. 1232-1250, 2011.

[25] O. C. Saka, S. Gürel and T. Van Woensel, "Using cost change estimates in a local search heuristic for the pollution routing problem," *OR Spectrum,* vol. 39, no. 2, pp. 557-587, 2017.

[26] S. Pelletier, O. Jabali and G. Laporte, "50th Anniversary Invited Article—Goods Distribution with Electric Vehicles: Review and Research Perspectives," *Transportation Science,* vol. 50, no. 1, pp. 3-22, 2016.

[27] M. Schneider, A. Stenger and D. Goeke, "The Electric Vehicle-Routing Problem with Time Windows and Recharging Stations," *Transportation Science,* vol. 48, no. 4, pp. 500-520, 2014.

[28] C. Malandraki and M. S. Daskin, "Time Dependent Vehicle Routing Problems: Formulations, Properties and Heuristic Algorithms," *Transportation Science,* vol. 26, no. 3, pp. 161-260, 1992.

[29] M. Dror, G. Laporte and P. Trudeau, "Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks," *Transportation Science,* vol. 23, no. 3, pp. 151-229, 1989.

[30] M. Dror and P. Trudeau, "Split delivery routing," *Naval Research Logistics,* vol. 37, no. 3, pp. 383-402, 1990.

[31] A. Larsen, "The dynamic vehicle routing problem," Technical University of Denmark, Kongens Lyngby, 2000.

[32] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera and A. A.Juan, "Rich vehicle routing problem: Survey," *ACM Computing Surveys,* vol. 47, no. 2, 2014.

[33] T. L. Magnanti, "Combinatorial optimization and vehicle fleet planning: Perspectives and prospects," *Networks,* vol. 11, no. 2, pp. 179-213, 1981.

[34] P. Toth and D. Vigo, The Vehicle Routing Problem, Philadelphia, PA: Society for Industrial and Applied Mathematics, 2001.

[35] R. V. Kulkarni and P. R. Bhave, "Integer programming formulations of vehicle routing problems," *European Journal of Operations Research,* vol. 20, no. 1, pp. 58-67, 1985.

[36] G. Laporte, Y. Nobert and M. Desrochers, "Optimal routing under capacity and distance restrictions," *Operations Research,* vol. 33, pp. 1050-1073, 1985.

[37] B. L. Golden, T. L. Magnanti and H. Q. Nguyan, "Implementing vehicle routing algorithms," *Networks,* vol. 7, no. 2, pp. 113-148, 1977.

[38] M. Fisher and R. Jaikumar, "A generalized assignment heuristic for vehicle routing," *Networks,* vol. 11, pp. 109-124, 1981.

[39] B. Gavish and S. C. Graves, "The travelling salesman problem and related problems," Massachusetts Institute of Technology, Operations Research Center, 1978.

[40] R. Baldacci, E. Hadjiconstantinou and A. Mingozzi, "An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation," *Operations Research,* vol. 52, pp. 723-738, 2004.

[41] M. L. Balinski and R. E. Quandt, "On an Integer Program for a Delivery Problem," *Operations Research,* vol. 12, no. 2, pp. 300-304, 1964.

[42] G. Laporte, "The Vehicle Routing Problem: An overview of exact and approximate algorithms," *European Journal of Operational Research,* vol. 59, pp. 345-358, 1992.

[43] P. Toth and D. Vigo, "Models, relaxations and exact approaches for the capacitated vehicle routing problem," *Discrete Applied Mathematics,* vol. 123, pp. 487-512, 2002.

[44] G. Laporte and Y. Nobert, "Exact Algorithms for the Vehicle Routing Problem," *Surveys in Combinatorial Optimization,* vol. 132, pp. 147-184, 1987.

[45] G. Laporte, H. Mercure and Y. Nobert, "An exact algorithm for the asymmetrical capacitated vehicle routing problem," *Networks,* vol. 16, no. 1, pp. 33-46, 1986.

[46] N. Christofides, A. Mingozzi and P. Toth, "Exact Algorithms for the Vehicle Routing Problem, Based on Spanning Tree and Shortest Path Relaxations," *Mathematical Programming,* vol. 20, pp. 255-282, 1981a.

[47] E. Hadjiconstantinou, N. Christofides and A. Mingozzi, "A new exact algorithm for the vehicle routing problem based onq-paths andk-shortest paths relaxations," *Annals of Operations Research,* vol. 61, no. 1, pp. 21-43, 1995.

[48] S. Eilon, C. D. T. Watson-Gandy and N. Christofides, Distribution management: mathematical modelling and practical analysis, Wisconsin: Griffin, 1971.

[49] N. Christofides, A. Mingozzi and P. Toth, "Space state relaxation procedures for the computation of bounds to routing problems," *Networks,* vol. 11, pp. 145-164, 1981b.

[50] W. Ou and B.-G. Sun, "A Dynamic Programming Algorithm for Vehicle Routing Problems," in *2010 International Conference on Computational and Information Sciences (ICCIS)*, Chengdu, 2013.

[51] C. Novoa and R. Storer, "An approximate dynamic programming approach for the vehicle routing problem with stochastic demands," *European Journal of Operational Research,* vol. 196, no. 2, pp. 509-515, 2009.

[52] N. Secomandi, "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands," *Computers & Operations Research,* vol. 27, no. 11-12, pp. 1201-1225, 2000.

[53] M. Dell'Amico, G. Righini and M. Salani, "A Branch-and-Price Approach to the Vehicle Routing Problem with Simultaneous Distribution and Collection," *Transportation Science,* vol. 40, no. 2, pp. 133-258, 2006.

[54] M. Salani and I. Vacca, "Branch and price for the vehicle routing problem with discrete split deliveries and time windows," *European Journal of Operational Research,* vol. 213, no. 3, pp. 470-477, 2011.

[55] H. B. Ticha, N. Absi, D. Feillet, A. Quilliot and T. Van Woensel, "A branch-and-price algorithm for the vehicle routing problem with time windows on a road network," *Networks,* vol. 73, no. 4, pp. 401-417, 2019.

[56] Y. Yu, S. Wang, J. Wang and M. Huang, "A branch-and-price algorithm for the heterogeneous fleet green vehicle routing problem with time windows," *Transportation Research Part B: Methodological,* vol. 122, pp. 511-527, 2019.

[57] D. Naddef and G. Rinaldi, "Branch-And-Cut Algorithms for the Capacitated VRP," in *The Vehicle Routing Problem,* Society for Industrial and Applied Mathematics, 2002, pp. 53-84.

[58] J. Lysgaard, A. N. Letchford and R. W. Eglese, "A new branch-and-cut algorithm for the capacitated vehicle routing problem," *Mathematical Programming,* vol. 100, p. 423–445, 2004.

[59] J. F. Bard, G. Kontoravdis and G. Yu, "A Branch-and-Cut Procedure for the Vehicle Routing Problem with Time Windows," *Transportation Science,* vol. 36, no. 2, pp. 250-269, 2002.

[60] R. Fukasawa, H. Longo, J. Lysgraad, M. Poggi de Aragão, M. Reis, E. Uchoa and R. F. Werneck, "Robust branch-and-cut-and-price for the capacitated vehicle routing problem," *Mathematical Programming Ser. A,* vol. 106, pp. 491-511, 2006.

[61] G. Desaulniers, "Branch-and-Price-and-Cut for the Split-Delivery Vehicle Routing Problem with Time Windows," *Operations Research,* vol. 58, no. 1, pp. 179-192, 2010.

[62] S. Ropke and J.-F. Cordeau, "Branch and Cut and Price for the Pickup and Delivery Problem with Time Windows," *Transportation Science,* vol. 43, no. 3, pp. 267-286, 2009.

[63] C. Archetti, M. Bouchard and G. Desaulniers, "Enhanced Branch and Price and Cut for Vehicle Routing with Split Deliveries and Time Windows," *Transportation Science,* vol. 45, no. 3, pp. 285-298, 2011.

[64] D. Lu and F. Gzara, "The robust vehicle routing problem with time windows: Solution by branch and price and cut," *European Journal of Operational Research,* vol. 275, no. 3, pp. 925-938, 2019.

[65] G. Laporte, S. Ropke and T. Vidal, "Heuristics for the Vehicle Routing Problem," in *Vehicle Routing: Problems, Methods, and Applications*, Society for Industrial and Applied Mathematics, 2014, pp. 87-116.

[66] A. Landeghem, "A bi-criteria heuristic for the vehicle routing problem with time windows," *European Journal of Operational Research,* vol. 36, pp. 217-226, 1988.

[67] H. Paessens, "The savings algorithm for the vehicle routing problem," *European Journal of Operational Research,* vol. 34, pp. 336-344, 1988.

[68] B. E. Gillett and L. R. Miller, "A heuristic algorithm for the vehicle dispatch problem," *Operations Research,* vol. 22, pp. 340-349, 1974.

[69] B. A. Foster and D. M. Ryan, "An integer programming approach to the vehicle routing problem," *Operations Research,* vol. 27, pp. 367-384, 1976.

[70] J. Renaud and F. F. Boctor, "A sweep-based algorithm for the fleet size and mix vehicle routing problem," *European Journal of Operational Research,* vol. 140, pp. 618-628, 2002.

[71] J. E. Beasley, "Route first—Cluster second methods for vehicle routing," *Omega,* vol. 11, no. 4, pp. 403-408, 1983.

[72] N. A. El-Sherbeny, "Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods," *Journal of King Saud University (Science)*, vol. 22, pp. 123-131, 2010.

[73] I. H. Osman, "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," *Annals of Operations Research*, vol. 41, pp. 421-451, 1993.

[74] M. Gendrau, A. Hertz and G. Laporte, "A tabu search heuristic for the vehicle routing problem," *Management Science*, vol. 40, pp. 1276-1290, 1994.

[75] G. Barbarosoglu and D. Ozgur, "A tabu search algorithm for the vehicle routing problem," *Computers & Operations Research*, vol. 26, pp. 255-270, 1999.

[76] J. F. Cordeau, G. Laporte and A. Mercier, "A unified tabu search heuristic for vehicle routing problems with time windows," *Journal of the Operational Research Society*, vol. 52, pp. 928-936, 2001.

[77] P. Toth and D. Vigo, "The granular tabu search and its application to the vehicle routing problem," *INFORMS Journal on Computing*, vol. 15, pp. 333-348, 2003.

[78] J. Brandão, "A tabu search algorithm for the open vehicle routing problem," *European Journal of Operational Research*, vol. 157, pp. 552-564, 2004.

[79] P. Chen, H.-k. Huang and X.-Y. Dong, "Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem," *Expert Systems with Applications*, vol. 37, no. 2, p. 1620–1627, 2010.

[80] J. Kytöjoki, T. Nuortio, O. Bräysy and M. Gendreau, "An efficient variable neighborhood search heuristic for very large scale vehicle routing problems," *Computers & Operations Research*, vol. 34, no. 9, pp. 2743-2757, 2007.

[81] P. Shaw, "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems," in *International conference on principles and practice of constraint programming*, Berlin, Heidelberg, Springer, 1998, pp. 417-431.

[82] D. Pisinger and S. Ropke, "Large neighborhood search," in *Handbook of Metaheuristics*, Springer, 2010, pp. 99-127.

[83] S. Ropke and D. Pisinger, "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows," *Transportation Science*, pp. 455-472, 2006.

[84] B. M. Baker and M. A. Ayechew, "A genetic algorithm for the vehicle routing problem," *Computers & Operations Research*, vol. 30, pp. 787-800, 2003.

[85] J. Berger and M. Barkaoui, "A new hybrid genetic algorithm for the capacitated vehicle routing problem," *Journal of the Operational Research Society*, vol. 54, pp. 1254-1262, 2003.

[86] C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem," *Computers & Operations Research*, vol. 31, pp. 1985-2002, 2004.

[87] D. Mester and O. Bräysy, "Active guided evolution strategies for large-scale capacitated vehicle routing problems," *Computers & Operations Research,* vol. 34, pp. 2964-2975, 2007.

[88] Y. Nagata, "Edge assembly crossover for the capacitated vehicle routing problem," *Lecture Notes in Computer Science,* vol. 4446, pp. 142-153, 2007.

[89] M. Reimann, K. Doerner and R. F. Hartl, "D-Ants: Savings Based Ants divide and conquer the vehicle routing problem," *Computers & Operations Research,* vol. 31, no. 4, pp. 563-591, 2004.

[90] T. J. Ai and V. Kachitvichyanukul, "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery," *Computers & Operations Research,* vol. 36, no. 5, pp. 1693-1702, 2009.

[91] P. Moscato and C. Cotta, "A Modern Introduction to Memetic Algorithms," in *Handbook of Metaheuristics*, New York, Springer, 2010, pp. 141-183.

[92] Ç. Koç, T. Bektaş, O. Jabali and G. Laporte, "A hybrid evolutionary algorithm for heterogeneous fleet vehicle routing problems with time windows," *Computers & Operations Research,* vol. 64, pp. 11-27, 2015.

[93] C. Archetti and M. G. Speranza, "A survey on matheuristics for routing problems," *EURO Journal on Computational Optimization,* vol. 2, no. 4, p. 223–246, 2014.

[94] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan and R. Qu, "Hyper-heuristics: a survey of the state of the art," *Journal of the Operational Research Society,* vol. 64, p. 1695–1724, 2013.

[95] T. G. Crainic, "Service Network Design in Freight Transportation," *European Journal of Operational Research,* vol. 122, no. 2, pp. 272-288, 2000.

[96] M. Hewitt, G. L. Nemhauser and M. W. P. Savelsbergh, "Combining Exact and Heuristic Approaches for the Capacitated Fixed-Charge Network Flow Problem," *Informs Journal on Computing,* vol. 22, no. 2, pp. 314 - 325, 21 September 2009.

[97] T. G. Crainic, M. Gendreau and J. M. Farvolden, "A Simplex-Based Tabu Search Method for Capacitated Network Design," *Informs Journal on Computing,* vol. 12, no. 3, pp. 223 - 236, 1 August 2000.

[98] I. Ghamlouche, T. G. Crainic and M. Gendreau, "Cycle-Based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design," *Operations Research,* vol. 51, no. 4, pp. 55 - 667, 1 August 2003.

[99] T. G. Crainic, B. Gendron and G. Hernu, "A Slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design," *Journal of Heuristics,* p. 525–545, 2004.

[100] M. Hewitt, G. Nemhauser and M. W. P. Savelsbergh, "Branch-and-Price Guided Search for Integer Programs with an Application to the Multicommodity Fixed-Charge Network Flow Problem," *Informs Journal on Computing,* vol. 25, no. 2, pp. 302 - 316, 2012.

[101] L. Barcos, V. Rodriguez, M. J. Alvarez and F. Robuste, "Routing design for less-than-truckload motor carriers using Ant Colony Optimization," *Transportation Research Part E,* p. 367–383, 2010.

[102] A. I. Jarrah, E. Johnson and L. C. Neubert, "Large-Scale, Less-than-Truckload Service Network Design," *Operations Research,* pp. 609-625, 2009.

[103] W. B. Powell and Y. Sheffi, "The Load Planning Problem of Motor Carriers: Problem Description and a Proposed Solution Approach," *Transportation Research Part A: General,* vol. 17, no. 6, p. 471–480, 1983.

[104] A. Erera, M. Hewitt, M. Savelsbergh and Y. Zhang, "Improved Load Plan Design Through Integer Programming Based Local Search," *Transportation Science,* vol. 47, no. 3, pp. 412-427, 2013.

[105] K. Lindsey, A. Erera and M. Savelsbergh, "Improved Integer Programming-Based Neighborhood Search for Less-Than-Truckload Load Plan Design," *Transportation Science,* vol. 50, no. 4, pp. 1360-1379, 2016.

[106] T. Yamada and Z. Febri, "Freight Transport Network Design Using Particle Swarm Optimisation in Supply Chain–transport Supernetwork Equilibrium," *Transportation Research Part E: Logistics and Transportation,* vol. 75, p. 164–187, 2015.

[107] Y. H. Lee, J. W. Jung and K. M. Lee, "Vehicle routing scheduling for cross-docking in the supply chain," *Computers & Industrial Engineering,* vol. 51, no. 2, p. 247–256, 2006.

[108] C.-J. Liao, Y. Lin and S. C. Shih, "Vehicle routing with cross-docking in the supply chain," *Expert Systems with Applications,* vol. 37, no. 10, p. 6868–6873, 2010.

[109] C. D. Tarantilis, "Adaptive multi-restart Tabu Search algorithm for the vehicle routing problem with cross-docking," *Optimization Letters,* vol. 7, no. 7, pp. 1583-1596, 2013.

[110] F. A. Santos, G. R. Mateus and A. Salles da Cunha, "The Pickup and Delivery Problem with Cross-Docking," *Computers & Operations Research,* vol. 40, no. 4, pp. 1085-1093, 2013.

[111] V. W. C. Morais, G. R. Mateus and T. F. Noronha, "Iterated local search heuristics for the Vehicle Routing Problem with Cross-Docking," *Expert Systems with Applications,* vol. 41, no. 16, p. 7495–7506, 2014.

[112] P. Grangier, M. Gendreau, F. Lehuédé and L.-M. Rousseau, "A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking," *Computers and Operations Research,* vol. 84, pp. 116-126, 2017.

[113] V. Schmid, K. F. Doerner and G. Laporte, "Rich routing problems arising in supply chain management," *European Journal of Operational Research,* vol. 224, no. 3, p. 435–448, 2013.

[114] J. Gonzalez-Feliu, "Vehicle Routing in Multi-Echelon Distribution Systems with Cross-Docking: A Systematic Lexical-Metanarrative Analysis," *Computer and Information Science,* vol. 6, no. 3, pp. 28-47, 2013.

[115] R. Cuda, G. Guastaroba and M. Speranza, "A survey on two-echelon routing problems," *Computers & Operations Research,* vol. 55, p. 185–199, 2015.

[116] R. Baldacci, A. Mingozzi, R. Roberti and R. W. Calvo, "An Exact Algorithm for the Two-Echelon Capacitated Vehicle Routing Problem," *Operations Research,* vol. 61, no. 2, pp. 298-314, 2013.

[117] M. Jepsen, S. Spoorendonk and S. Ropke, "A Branch-and-Cut Algorithm for the Symmetric Two-Echelon Capacitated Vehicle Routing Problem," *Transportation Science,* vol. 47, no. 1, pp. 23-37, 2013.

[118] G. Perboli, R. Tadei and D. Vigo, "The Two-Echelon Capacitated Vehicle Routing Problem: Models and Math-Based Heuristics," *Transportation Science,* vol. 45, no. 3, pp. 364-380, 2011.

[119] T. G. Crainic, S. Mancini, G. Perboli and R. Tadei, "Multi-start Heuristics for the Two-Echelon Vehicle Routing Problem," in *Evolutionary Computation in Combinatorial Optimization 11th European Conference, EvoCOP 2011, Torino, Italy, April 27-29, 2011, Proceedings*, Springer Berlin Heidelberg, 2011, pp. 179-190.

[120] V. C. Hemmelmayr, J.-F. Cordeau and T. G. Crainic, "An adaptive large neighborhood search heuristic for Two-Echelon Vehicle Routing Problems arisingin city logistics," *Computers &OperationsResearch,* vol. 39, no. 12, p. 3215–3228, 2012.

[121] M. Drexl, "Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints," *Transportation Science,* vol. 46, no. 3, p. 297–316, 2012.

[122] F. Neves-Moreira, P. Amorim, L. Guimarães and B. Almada-Lobo, "A long-haul freight transportation problem: Synchronizing resources to deliver requests passing through multiple transshipment locations," *European Journal of Operational Research,* vol. 248, no. 2, p. 487–506, 2016.

[123] T. Gschwind, "A comparison of column-generation approaches to the Synchronized Pickup and Delivery Problem," *European Journal of Operational Research,* vol. 247, no. 1, pp. 60-71, 2015.

[124] A. Grimault, N. Bostel and F. Lehuédé, "An adaptive large neighborhood search for the full truckload pickup and delivery problem with resource synchronization," *Computers and Operations Research,* vol. 88, pp. 1-14, 2017.

[125] M. W. P. Savelsbergh and M. Sol, "The General Pickup and Delivery Problem," *Transportation Science,* vol. 29, no. 1, pp. 17-29, 1995.

[126] Y. Dumas, J. Desrosiers and F. Soumis, "The pickup and delivery problem with time windows," *European Journal of Operational Research,* vol. 54, no. 1, p. 7–22, 1991.

[127] S. Ropke, J.-F. Cordeau and G. Laporte, "Models and Branch-and-Cut Algorithms for Pickup and Delivery Problems with Time Windows," *Networks,* vol. 49, no. 4, pp. 258-272, 2007.

[128] S. Ropke and J.-F. Cordeau, "Branch and Cut and Price for the Pickup and Delivery Problem with Time Windows," *Transportation Science,* vol. 43, no. 3, pp. 267-286, 2009.

[129] S. Ropke and D. Pisinger, "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows," *Transportation Science,* vol. 40, no. 4, pp. 455-472, 2006.

[130] W. P. Nanry and J. W. Barnes, "Solving the pickup and delivery problem with time windows using reactive tabu search," *Transportation Research Part B: Methodological,* vol. 34, no. 2, pp. 107-121, 2000.

[131] H. Li and A. Lim, "A Metaheuristic for the Pickup and Delivery Problem with Time Windows," *International Journal on Artificial Intelligence Tools,* vol. 12, no. 2, pp. 173-186, 2003.

[132] R. Bent and P. V. Hentenryck, "A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows," *Computers & Operations Research,* vol. 33, no. 4, p. 875–893, 2006.

[133] A. I. Nikolopoulou, P. P. Repoussis, C. D. Tarantilis and E. E. Zachariadis, "Moving products between location pairs: Cross-docking versus direct-shipping," *European Journal of Operational Research,* vol. 256, no. 3, pp. 803-819, 2017.

[134] S. Mitrović-Minić and G. Laporte, "The pickup and delivery problem with time windows and transshipment," *INFOR: Information Systems and Operational Research,* vol. 44, no. 3, pp. 217-227, 2006.

[135] C. E. Cortés, M. Matamala and C. Contardo, "The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method," *European Journal of Operational Research,* vol. 200, no. 3, pp. 711-724, 2010.

[136] A. Rais, F. Alvelos and M. S. Carvalho, "New mixed integer-programming model for the pickup-and-delivery problem with transshipment," *European Journal of Operational Research,* vol. 235, no. 3, p. 530–539, 2014.

[137] H. L. Petersen and S. Ropke, "The Pickup and Delivery Problem with Cross-Docking Opportunity," in *ICCL 2011: Computational Logistics*, Hamburg, 2011.

[138] R. Masson, F. Lehuédé and O. Péton, "An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Transfers," *Transportation Science,* vol. 47, no. 3, pp. 344-355, 2013.

[139] M. Wen, J.-F. L. G. Cordeau and J. Larsen, "The dynamic multi-period vehicle routing problem," *Computers & Operations Research,* vol. 37, no. 9, pp. 1615-1623, 2010.

[140] M. Albareda-Sambola, E. Fernández and G. Laporte, "The dynamic multiperiod vehicle routing problem with probabilistic information," *Computers&OperationsResearch,* vol. 48, p. 31–39, 2014.

[141] C. Archetti, O. Jabali and M. G. Speranza, "Multi-period Vehicle Routing Problem with Due dates," *Computers & Operations Research,* vol. 61, pp. 122-134, 2015.

[142] S. Mancini, "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, 2016.

[143] I. Dayarian, T. G. Crainic, M. Gendreau and W. Rei, "An adaptive large-neighborhood search heuristic for a multi-period vehicle routing problem," *Transportation Research Part E: Logistics and Transportation Review,* vol. 95, pp. 95-123, 2016.

[144] V. F. Yu, P. Jewpanya and V. Kachitvichyanukul, "Particle swarm optimization for the multi-period cross-docking distribution problem with time windows," *International Journal of Production Research,* vol. 54, no. 2, pp. 509-525, 2016.

[145] K. Doerner, R. F. Hartl and M. Reimann, "A hybrid ACO algorithm for the Full Truckload Transportation Problem," *Adaptive Information Systems and Modelling in Economics and Management Science,* pp. 1-11, 2001.

[146] S. Arunapuram, K. Mathur and D. Solow, "Vehicle Routing and Scheduling with Full Truckloads," *Transportation Science,* pp. 170-182, 2003.

[147] M. Gronalt, R. F. Hartl and M. Reimann, "New savings based algorithms for time constrained pickup and delivery of full truckloads," *European Journal of Operational Research,* p. 520–535, 2003.

[148] R. Liu, Z. Jianga, R. Y. K. Fung, F. Chen and X. Liu, "Two-phase heuristic algorithms for full truckloads multi-depot capacitated vehicle routing problem in carrier collaboration," *Computers & Operations Research,* pp. 950--959, 2010.

[149] J. Li and W. Lu, "Full truckload vehicle routing problem with profits," *Journal of Traffic and Transportation Engineering (English Edition),* vol. 1, no. 2, pp. 146-152, 2014.

[150] B. Fleischmann, "The vehicle routing problem with multiple use of vehicles," Fachbereich Wirtschaftswissenschaften, Hamburg, 1990.

[151] D. Cattaruzza, N. Absi and D. Feillet, "Vehicle routing problems with multiple trips," *4OR: A Quarterly Journal of Operations Research,* vol. 14, no. 3, pp. 223-259, 2016.

[152] N. Azi, M. Gendreau and J.-Y. Potvin, "An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles," *European Journal of Operational Research,* vol. 202, no. 3, pp. 756-763, 2010.

[153] F. Hernandez, D. Feillet, R. Giroudeau and O. Naud, "A new exact algorithm to solve the multi-trip vehicle routing problem with time windows and limited duration," *4OR,* vol. 12, no. 3, p. 235–259, 2014.

[154] R. Petch and S. Salhi, "A multi-phase constructive heuristic for the vehicle routing problem with multiple trips," *Discrete Applied Mathematics,* vol. 133, no. 1-3, pp. 69-92, 2003.

[155] F. Despaux and S. Basterrech, "Multi-Trip Vehicle Routing Problem with Time Windows and Heterogeneous Fleet," *International Journal of Computer Information Systems and Industrial Management Applications,* vol. 8, pp. 355-363, 2016.

[156] M. P. Seixas and A. B. Mendes, "Column generation for a multitrip vehicle routing problem with time windows, driver work hours, and heterogeneous fleet," *Mathematical Problems in Engineering,* vol. 2013, pp. 1-13, 2013.

[157] Y. P. Anggodo, A. K. Ariyani, M. K. Ardi and W. F. Mahmudy, "Optimization of multi-trip vehicle routing problem with the time windows using genetic algorithm," *Journal of Environmental Engineering & Sustainable Technology,* vol. 3, no. 2, pp. 92-97, 2016.

[158] R. Ayadi and Y. Benadada, "Memetic algorithm for a multi-objective vehicle routing problem with multiple trips," *International Journal of Computer Science and Applications,* vol. 10, no. 2, pp. 72-91, 2013.

[159] D. Cattaruzza, N. Absi, D. Feillet and T. Vidal, "A memetic algorithm for the Multi Trip Vehicle Routing Problem," *European Journal of Operational Research,* vol. 236, no. 3, pp. 833-848, 2014.

[160] F. Alonso, M. Alvarez and J. Beasley, "A tabu search algorithm for the periodic vehicle routing problem with multiple vehicle trips and accessibility restrictions," *Journal of the Operational Research Society,* vol. 59, no. 7, pp. 963-976, 2008.

[161] P. K. Nguyen, T. G. Crainic and M. Toulouse, "A tabu search for time-dependent multi-zone multi-trip vehicle routing problem with time windows," *European Journal of Operational Research,* vol. 231, no. 1, pp. 43-56, 2013.

[162] T. G. Crainic, Y. Gajpal and M. Gendreau, "Multi-Zone Multi-Trip Vehicle Routing Problem with Time Windows," *INFOR: Information Systems and Operational Research,* vol. 53, no. 2, pp. 49-67, 2015.

[163] L. C. Coelho, Laporte and Gilbert, "Classification, Models and Exact Algorithms for Multi-compartment Delivery Problems," *European Journal of Operational Research,* vol. 242, no. 3, pp. 854-864, 2015.

[164] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth and A. H. Teller, "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics,* vol. 21, no. 6, pp. 1087-1092, 1953.

[165] Solomon and M. M, "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints," *Operations Research,* vol. 35, no. 2, pp. 254-266, 1987.

APPENDIX A

Pseudocode of the Matheuristic Construction Algorithm:

Step 0. Initialization

- Activate the largest vehicle type
- *active_orders* ← list of all orders
- *collection_orders* ← { }, *last_mile_delivery_orders* ← { }, *unfulfilled_orders* ← { }
- *activated_route_vehicle_pairs* ← { } Empty

Step 1. Plan Pickup and Delivery Routes (Type 2)

- Partition *active_orders* based on the pickup and delivery zone
- For each order segment, solve the LTLP-C-Sub with Type 2 routes
- Remove the orders assigned to a route-vehicle pair from *active_orders*
- Append the activated route-vehicle pairs to *activated_route_vehicle_pairs*

Step 2. Decompose Orders

- Decompose each order in *active_orders* into two such that:
  - *order1:* From pickup location of the parent order to cross-dock at the delivery zone
  - *order2:* From cross-dock at the delivery zone to delivery location of the parent order
  - Set the values of latest delivery time of *order1* and earliest pickup time of *order2* such that the timespan allocated to each order is sufficient and synchronization of orders is preserved
- Keep *order1* in *active_orders*, append *order2* to *last_mile_delivery_orders*

Step 3. Plan Inter-zone Collection Routes (Type 1)

- Partition *active_orders* based on the pickup and delivery zone
- For each order segment, solve the LTLP-C-Sub with inter-zone Type 1 routes
- Remove the orders assigned to a route-vehicle pair from *active_orders*
- Update the earliest pickup of associated last-mile delivery orders
- Append the activated route-vehicle pairs to *activated_route_vehicle_pairs*

Step 4. Add Extra Delivery Stops (Type 1 to Type 4 Conversion)

- For each route-vehicle pair with route Type 1 in *activated_route_vehicle_pairs*:
  - While the maximum delivery location limit defined by Type 4 are not violated:
    - ✓ Sort the orders assigned to the route-vehicle pair having a size bigger than a certain threshold limit according to their closeness to the last location of the route.
    - ✓ Append the delivery location of the first order to the end of the route if the maximum deviation limit from the shortest path is not violated. Remove the associated last-mile delivery order from *last_mile_delivery_orders*. Change the type of route-vehicle pair to Type 4.

Step 5. Reunite and Decompose Orders

- Reunite orders in *active_orders* with their associated orders in *last_mile_delivery_orders*
- Decompose each order in *active_orders* into two such that:
  - *order1:* From pickup location of the parent order to cross-dock at the pickup zone
  - *order2:* From cross-dock at the pickup zone to delivery location of the parent order
  - Set the values of latest delivery time of *order1* and earliest pickup time of *order2* such that the timespan allocated to each order is sufficient and synchronization of orders is preserved.
- Keep *order2* in *active_orders*, append *order2* to *collection_orders*

Step 6. Plan Inter-zone Distribution Routes (Type 2)

- Partition *active_orders* based on the pickup and delivery zone
- For each order segment, solve the LTLP-C-Sub with route Type 2
- Remove the orders assigned to a route-vehicle pair from *active_orders*
- Update the latest delivery of associated collection orders
- Append the activated route-vehicle pairs to *activated_route_vehicle_pairs*

Step 7. Add Extra Pickup Stops (Type 2 to Type 4 Conversion)

- For each route-vehicle pair with route type 2 in *activated_route_vehicle_pairs*:
  - While the maximum pickup location limit defined by RT4 are not violated:
    - ✓ Sort the orders assigned to the route-vehicle pair having a size bigger than a certain threshold limit according to their closeness to the first location of the route.

- ✓ Append the pickup location of the first order to the beginning of the route if the maximum deviation limit from the shortest path is not violated. Remove the associated collection order from *collection_orders*. Change the type of route-vehicle pair to Type 4.

At the end of this process, if the active vehicle type is the one with the smallest capacity, then activate all vehicle types and continue to Step 8. Otherwise, reunite the orders in *active_orders, collection_orders* and *last_mile_delivery_orders*. Activate the next largest vehicle and go back to Step 1.

Step 8. Reunite and Decompose Orders

- Reunite orders in *active_orders* with their associated orders in *collection_orders*
- Decompose each order in *active_orders* into three such that:
  - *order1:* From pickup location of the parent order to cross-dock at the pickup zone
  - *order2:* From cross-dock at the pickup zone to cross-dock at the delivery zone
  - *order3:* From cross-dock at the delivery zone to delivery location of the parent order
  - Set the values of latest delivery time of *order1* and *order2*, and earliest pickup time of *order2* and *order3* such that the timespan allocated to each order is sufficient and synchronization of orders is preserved
- Keep *order2* in *active_orders*, append *order1* to *collection_orders*, append *order3* to *last_mile_delivery_orders*

Step 9. Plan Trunk Routes (Type 5)

- Partition *active_orders* based on the pickup and delivery zone
- For each order segment, solve the LTLP-C-Sub with route Type 5
- Remove the orders assigned to a route-vehicle pair from *active_orders*
- Update the latest delivery of associated collection orders and the earliest pickup of associated last-mile delivery orders
- Remove the unassigned orders from *active_orders* and append them to *unfulfilled_orders*
- Append the activated route-vehicle pairs to *activated_route_vehicle_pairs*

Step 10. Add Extra Delivery Stops (Type 5 to Type 7 conversion)

- For each route-vehicle pair with route type 5 in *activated_route_vehicle_pairs*:
  - While the maximum delivery location limit defined by Type 7 are not violated:
    - ✓ Sort the orders assigned to the route-vehicle pair having a size bigger than a certain threshold limit according to their closeness to the last location of the route.

- ✓ Append the delivery location of the first order to the end of the route if the maximum deviation limit from the shortest path is not violated. Remove the associated last-mile delivery order from *last_mile_delivery_orders*. Change the type of route-vehicle pair to Type 7.

Step 11. Add Extra Pickup Stops (Type 5 to Type 6 and Type 7 to Type 8 conversions)

- For each route-vehicle pair with route type 5 (6) in *activated_route_vehicle_pairs*:
  - While the maximum pickup location limit defined by Type 6 (Type 8) are not violated:
    - ✓ Sort the orders assigned to the route-vehicle pair having a size bigger than a certain threshold limit according to their closeness to the first location of the route.
    - ✓ Append the pickup location of the first order to the beginning of the route if the maximum deviation limit from the shortest path is not violated. Remove the associated collection order from *collection_orders*. Change the type of route-vehicle pair to Type 6 (Type 8).

Step 12. Plan Intra-zone Collection Orders (Type 1)

- Set *active_orders* ← *collection_orders*
- Partition *active_orders* based on the pickup and delivery zone
- For each order segment, solve LTLP-C-Sub with route Type 1
- Remove the orders assigned to a route-vehicle pair from *active_orders*
- Append the activated route-vehicle pairs to *activated_route_vehicle_pairs*

Step 13. Plan Last-Mile Delivery Orders (Type 2)

- Set *active_orders* ← *last_mile_delivery_orders*
- Partition *active_orders* based on the pickup and delivery zone
- For each order segment, solve LTLP-C-Sub with route Type 2
- Remove the orders assigned to a route-vehicle pair from *active_orders*
- Append the activated route-vehicle pairs to *activated_route_vehicle_pairs*

APPENDIX B

Table B.1. Vehicle Type Parameters.

Physical Type	Ownership Type	Capacity	Max Stops	Max Extra Stops	Fixed Cost	Per Km Cost	Extra Stop Cost	Non-Use Cost	Cross-docking Cost
T1	Owned	16	10	0	0	2	0	100	150
T2	Owned	32	8	0	0	2.3	0	150	200
T1	Contract-hired	16	5	2	100	2	100	75	150
T2	Contract-hired	32	5	2	150	2.3	100	125	200
T1	Spot-hired	16	5	2	200	2.5	200	0	150
T2	Spot-hired	32	5	2	250	3	200	0	200

Table B.2. Loading and Unloading Times.

Parameter	Value
Loading Time (Pickup Locations)	0.5 hrs
Unloading Time (Delivery Locations)	0.5 hrs
Loading Time (Cross-docks)	0.75 hrs
Unloading Time (Cross-docks)	0.75 hrs

APPENDIX C

Table C.1. Matheuristic Parameters.

Parameter	Description	Value or Formula
T_{start}	Starting temperature	$\frac{-0.5C_{S_{best}}}{\ln(0.5)}$
CR	Cooling rate	$\sqrt[N]{\frac{0.1}{T_{start}}}$
N	Maximum number of improvement matheuristic iterations	50
K	Maximum number of iterations for a route-vehicle pair to stay in the tabu list	10
α	Coefficient for dependent route-vehicle pairs	2
β	Coefficient for strongly related route-vehicle pairs	1
γ	Coefficient for weakly related route-vehicle pairs	0.5
O_{min}	Minimum number of orders to be removed from solution	10
O_{max}	Maximum size of order set that can be solved by the exact solution algorithm	20
O_{max2}	Maximum size of order set that can be solved by the exact solution algorithm with a restricted set of route types	30

APPENDIX D

Table D.1. Daily Simulation Results for DH.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{CGH}	T_{tot}
1	260	370534.4	19	157	150	1167612.0	1167107.7	1163572.9	1629.2	583.2	2212.4
2	327	502412.8	42	284	276	1603069.1	1602958.4	1599953.4	1709.1	576.0	2285.1
3	255	396158.5	63	274	271	1234314.7	1234314.7	1233505.4	760.1	34.7	794.8
4	125	179018.0	62	180	180	541518.2	541518.2	541518.2	59.7	0.0	59.7
5	312	400682.7	56	251	245	1460981.7	1460961.5	1458493.0	1274.7	705.9	1980.5
6	382	578860.0	63	308	299	1787623.6	1787592.5	1783246.7	1912.3	778.6	2690.9
7	435	633780.8	81	337	326	1866508.0	1865701.7	1858041.6	2365.7	1572.8	3938.5
8	388	583340.5	67	338	329	1813980.5	1813980.5	1809686.3	1784.0	583.7	2367.7
9	416	654788.8	58	308	297	1808479.1	1808098.9	1801371.7	2170.5	885.9	3056.4
10	261	386543.7	75	271	267	1220764.4	1220588.7	1219371.8	803.2	227.8	1031.0
11	108	155331.5	62	178	178	500745.5	500745.5	500745.5	5.2	0.0	5.2
12	293	416798.6	50	238	231	1295136.0	1295119.1	1291929.0	1441.0	674.7	2115.7
13	402	621243.6	55	282	269	1656717.5	1656153.3	1647896.1	2593.7	945.7	3539.4
14	409	602598.4	72	288	277	1707353.9	1707267.1	1701820.6	2072.7	704.3	2777.1
15	431	658658.2	70	287	276	1782188.9	1782136.4	1777010.6	2305.2	1072.1	3377.3
16	432	623926.5	67	287	276	1815618.3	1814836.6	1806153.8	2365.0	1203.6	3568.5
17	344	546606.8	56	263	255	1392089.2	1391525.5	1387734.5	1626.3	372.3	1998.6
18	117	149702.8	34	150	150	473124.2	473124.2	473124.2	38.5	0.0	38.5
19	282	399093.0	50	210	202	1380410.6	1380306.7	1377483.1	1928.7	570.0	2498.7
20	410	573894.2	61	304	291	1902467.5	1901998.9	1895858.9	2500.5	1344.3	3844.7
21	473	713794.9	67	319	309	1981125.4	1980177.9	1972222.8	2101.5	950.2	3051.7
22	417	611245.2	67	338	330	2006961.6	2006546.7	2002948.0	1691.6	897.4	2588.9
23	442	710633.6	72	329	316	1903518.9	1903048.1	1895808.7	2580.1	1302.0	3882.1
24	305	485101.0	84	268	263	1338563.5	1338536.4	1336574.5	1016.8	340.2	1357.0
25	136	185780.2	54	195	195	619284.5	619284.5	619284.5	15.2	0.0	15.2
26	288	389192.7	47	236	229	1358868.3	1358751.1	1355945.0	1476.5	1097.8	2574.3
27	386	555239.9	54	292	286	1690964.0	1690964.0	1687240.4	1402.2	732.5	2134.6
28	327	449966.5	63	279	274	1594157.1	1593498.2	1590170.7	1048.4	607.5	1656.0
29	261	355323.8	56	246	241	1207059.3	1207044.6	1205689.0	979.3	307.4	1286.7
30	204	267867.3	44	218	217	980046.2	980046.2	979996.6	254.1	13.2	267.3
31	115	147779.4	43	157	157	479484.6	479484.6	479484.6	12.0	0.0	12.0
32	64	67434.9	29	87	87	188133.9	188133.9	188133.9	3.1	0.0	3.1
33	42	38419.7	13	40	40	81255.1	81255.1	81255.1	1.9	0.0	1.9
34	20	17386.6	7	19	19	32607.4	32607.4	32607.4	0.5	0.0	0.5
35	16	11304.4	0	11	11	11304.4	11304.4	11304.4	0.1	0.0	0.1
36	1	416.1	0	1	1	416.1	416.1	416.1	0.0	0.0	0.0
Overall	**9886**	**14440859.7**	**0**	**8230**	**8020**	**43884453.2**	**43877135.0**	**43767598.9**	**43928.5**	**19083.7**	**63012.2**

Table D.2. Daily Simulation Results for CH.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	255	410467.7	12	88	73	598679.5	568743.1	552227.6	3010.3	0.0	1464.5	4474.8
2	263	451746.4	45	110	97	666242.0	654988.6	610577.9	2447.3	0.0	1443.6	3890.9
3	199	309341.9	70	117	113	535520.1	530147.9	526463.2	970.8	0.0	583.9	1554.7
4	95	118053.4	52	95	95	283281.5	283281.5	283281.5	44.4	0.0	0.0	44.4
5	297	454915.4	33	129	116	754616.1	708731.8	636842.3	2730.0	0.0	1750.4	4480.4
6	315	511054.5	52	134	119	824653.3	801746.2	696473.8	3179.2	0.0	2031.7	5210.9
7	372	581623.1	55	146	128	959615.8	883332.2	752773.3	3769.5	0.0	2440.2	6209.7
8	336	555040.1	54	140	126	911249.2	833952.3	724737.6	2888.6	0.0	1722.6	4611.2
9	353	578068.6	50	139	123	901943.5	833974.7	757233.4	3473.8	0.0	2360.9	5834.7
10	209	311742.6	66	126	121	593090.8	571163.5	560952.2	1102.2	0.0	703.9	1806.1
11	93	102367.8	42	103	103	292180.9	292180.9	292180.9	56.2	0.0	0.0	56.2
12	292	449547.3	31	128	115	712587.0	712018.9	670116.6	2821.6	0.0	1784.8	4606.4
13	365	582559.0	37	130	113	1048461.5	852289.1	751511.7	3799.8	68.6	1893.3	5761.7
14	360	591896.3	50	130	114	921704.6	841175.7	677098.5	3413.2	0.0	2099.5	5512.7
15	368	589383.3	57	134	118	1055589.3	854989.9	632284.1	3624.7	65.9	2132.7	5823.3
16	379	592984.8	54	130	113	993990.6	833626.1	659648.3	3739.4	0.0	2287.9	6027.3
17	284	439970.3	50	124	114	722556.3	716018.3	705636.8	2146.5	0.0	925.0	3071.5
18	95	110193.0	42	91	90	295437.4	295437.4	295334.8	209.5	0.0	63.8	273.3
19	296	469057.5	26	122	110	743942.4	736329.7	719431.6	2579.7	0.0	1162.9	3742.6
20	375	564776.9	65	143	127	1136897.3	915040.2	736822.8	3456.6	0.0	2269.9	5726.5
21	415	678890.1	55	141	123	1058591.9	951886.2	824797.7	3876.4	0.0	2299.5	6176.0
22	350	534376.7	63	148	134	1080149.9	882970.0	780978.7	3032.2	59.8	2155.4	5247.4
23	391	648494.3	47	145	127	1045616.3	979889.5	827151.6	3492.0	0.0	2214.5	5706.5
24	261	431213.2	69	128	119	742693.5	737032.2	727794.1	1730.4	0.0	992.7	2723.0
25	125	149325.9	44	109	107	371680.8	371680.8	371256.5	400.7	0.0	187.7	588.4
26	292	434407.6	39	124	112	825820.6	747097.0	668441.8	2418.0	0.0	1583.8	4001.9
27	343	519685.9	51	121	109	787100.4	786345.7	724646.5	2606.8	0.0	1850.1	4456.9
28	275	412354.2	47	128	119	709961.6	661482.6	618315.8	1976.4	0.0	1317.3	3293.7
29	192	264346.9	43	97	92	489339.2	487401.7	483026.5	1028.4	0.0	630.6	1659.0
30	171	220779.9	34	96	94	468535.8	468535.8	467302.8	544.0	0.0	22.0	566.1
31	90	101532.3	45	91	91	273238.5	273238.5	273238.5	12.2	0.0	0.0	12.2
32	59	57465.6	23	63	63	167800.4	167800.4	167800.4	7.6	0.0	0.0	7.6
33	51	47965.0	10	52	52	137018.0	137018.0	137018.0	2.7	0.0	0.0	2.7
34	30	25289.2	6	32	32	68701.3	68701.3	68701.3	1.4	0.0	0.0	1.4
35	16	9311.2	1	22	22	25912.5	25912.5	25912.5	0.2	0.0	0.0	0.2
36	6	3505.7	0	10	10	9014.6	9014.6	9014.6	0.1	0.0	0.0	0.1
Overall	**8668**	**13313733.8**	**0**	**3966**	**3634**	**23213414.4**	**21475174.9**	**19417026.2**	**70592.9**	**194.2**	**42375.2**	**113162.3**

APPENDIX E

Table E.1. Daily Simulation Results for Scenario 2.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	184	306883.0	165	89	78	488892.2	487013.8	456827.0	2199.2	0.0	1300.1	3499.3
2	226	362445.9	220	111	99	682658.0	642018.7	604055.2	2461.9	0.0	1465.8	3927.7
3	173	299021.3	243	119	113	623363.7	595961.2	532692.3	1384.1	0.0	856.1	2240.2
4	96	162842.7	150	110	110	386250.0	386250.0	386250.0	73.9	0.0	0.0	73.9
5	238	359514.4	208	142	133	705617.8	677864.3	609981.4	2050.1	0.0	1413.2	3463.3
6	264	441091.2	285	151	138	897040.7	844212.0	731598.6	3045.3	0.0	2195.8	5241.1
7	327	513699.2	352	160	144	1030824.1	953248.8	861629.6	3548.0	0.0	2477.2	6025.2
8	291	500468.2	313	164	150	947933.2	934701.7	842032.7	3099.6	0.0	2215.7	5315.3
9	306	514103.9	334	163	148	1002256.0	951428.3	864006.5	3153.4	0.0	2372.6	5526.0
10	216	333401.8	287	159	152	764062.9	752711.9	715180.3	1482.5	0.0	825.7	2308.2
11	98	142056.1	207	145	145	490034.8	490034.8	490034.8	59.6	0.0	0.0	59.6
12	232	363391.3	260	167	155	779698.9	771551.1	733575.1	2397.9	0.0	1681.6	4079.5
13	285	472364.5	363	169	155	1018514.1	963316.9	818380.7	3484.5	116.3	1951.0	5551.8
14	326	530318.0	379	170	155	1125313.2	1043406.3	821264.8	3430.0	162.4	2447.7	6040.1
15	315	532510.8	395	174	157	1164206.8	1068659.6	886514.6	3941.7	119.7	2241.9	6303.4
16	358	564014.1	387	171	158	1212666.2	1114668.9	857665.3	3386.6	252.1	2265.3	5904.0
17	260	441468.9	367	167	155	963505.2	943463.4	932799.2	2553.8	0.0	1285.0	3838.8
18	106	149975.6	263	154	154	568783.9	568783.9	568783.9	157.5	0.0	0.0	157.5
19	245	390927.0	300	170	161	873468.9	868381.0	857374.5	2049.4	0.0	1114.1	3163.5
20	303	457599.5	410	177	160	1067966.9	1040772.1	960856.2	3563.2	0.0	2421.2	5984.4
21	367	591040.9	446	182	165	1215635.0	1183107.2	1097752.3	3856.7	0.0	2298.4	6155.0
22	335	531676.6	397	190	177	1279436.8	1143442.4	986277.7	3069.8	147.9	2223.4	5441.2
23	327	544407.0	421	196	182	1265927.7	1204129.1	1057265.9	3510.4	0.0	2348.2	5858.6
24	223	399782.5	414	189	180	1060843.9	1032968.6	990933.8	1943.2	0.0	1139.9	3083.1
25	134	216729.3	289	174	174	733156.6	733156.6	733156.6	356.5	0.0	0.0	356.5
26	234	373996.5	348	178	171	996683.0	942239.5	872057.3	1833.0	0.0	1282.5	3115.5
27	282	424868.5	416	177	167	1065078.3	1040743.2	957771.9	2512.4	0.0	1537.8	4050.2
28	242	389609.1	415	177	169	993562.7	981376.9	932310.3	2060.1	0.0	1296.3	3356.4
29	182	280650.4	363	167	162	849243.8	838907.9	800487.3	1255.3	0.0	665.1	1920.4
30	133	218045.2	325	163	163	740683.7	740683.7	740683.7	269.1	0.0	0.0	269.1
31	73	78563.0	314	159	158	589847.0	589847.0	589651.0	206.6	0.0	182.4	389.1
32	53	65445.5	279	156	156	533574.8	533574.8	533574.8	13.9	0.0	0.0	13.9
33	42	45239.7	250	144	144	494176.5	494176.5	494176.5	8.6	0.0	0.0	8.6
34	16	13371.0	249	133	133	430352.9	430352.9	430352.9	8.1	0.0	0.0	8.1
35	13	11978.1	245	131	131	419890.7	419890.7	419890.7	8.8	0.0	0.0	8.8
36	3	3057.6	243	129	129	402099.8	402099.8	402099.8	6.9	0.0	0.0	6.9
37	0	0.0	243	129	129	394536.9	394536.9	394536.9	7.0	0.0	0.0	7.0
38	0	0.0	244	129	129	395347.0	395347.0	395347.0	6.7	0.0	0.0	6.7
Overall	**7508**	**12026558.2**	**244**	**5935**	**5639**	**30653134.8**	**29599029.6**	**27359828.9**	**68455.3**	**798.4**	**43504.1**	**112757.8**

Table E.2. Daily Simulation Results for Scenario 3.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	277	446162.3	0	88	73	698624.7	683810.0	658943.8	3226.9	0.0	1456.2	4683.1
2	296	489541.6	9	110	95	770367.4	761499.6	695644.2	2979.8	0.0	1601.3	4581.2
3	239	366527.5	8	111	107	628206.8	621575.7	617734.2	951.5	0.0	592.5	1544.0
4	94	102038.0	3	74	74	256898.7	256898.7	256898.7	15.4	0.0	0.0	15.4
5	305	453352.8	3	121	106	891954.3	853346.1	747430.9	3102.7	0.0	2049.2	5151.9
6	351	564179.9	6	129	112	1075114.1	974716.9	844442.8	3581.3	0.0	2074.1	5655.5
7	391	611041.6	9	131	111	1079324.1	995018.8	810846.2	4157.2	0.0	2484.0	6641.2
8	357	563790.3	11	139	122	1014330.6	928678.9	815927.0	3302.6	0.0	1790.5	5093.1
9	386	617931.5	11	125	107	1109562.4	971483.9	894934.0	3690.4	0.0	2620.0	6310.4
10	241	336161.4	9	114	107	729267.7	685068.1	671655.1	1537.5	0.0	677.5	2215.0
11	91	101914.9	1	69	69	304357.2	304357.2	304357.2	7.2	0.0	0.0	7.2
12	308	441955.5	0	120	104	935226.2	916420.2	821582.1	3123.7	0.0	1885.4	5009.0
13	394	641822.6	6	126	109	1046336.4	998456.6	869682.2	3739.2	0.0	2077.1	5816.3
14	389	632709.0	8	124	105	1172699.3	920934.9	775317.4	3982.2	0.0	2074.3	6056.5
15	384	604943.6	5	117	99	1043771.2	923034.7	770105.0	3836.4	0.0	2060.6	5896.9
16	406	606218.4	6	120	103	1290073.7	937833.0	737228.2	3883.1	100.9	2268.5	6252.5
17	299	467378.1	5	111	101	869083.8	858859.0	839476.5	2342.4	0.0	1030.8	3373.2
18	104	118675.1	2	74	74	316058.1	316058.1	316058.1	44.9	0.0	0.0	44.9
19	299	459557.2	2	110	98	950901.1	907859.2	879926.2	2793.5	0.0	958.5	3752.0
20	427	639922.0	7	134	115	1159932.6	1117228.1	928825.0	4189.9	0.0	2094.2	6284.1
21	433	706555.7	7	138	119	1202613.5	1023854.1	882955.4	4025.8	0.0	2425.6	6451.4
22	385	581644.8	7	132	114	1400225.0	1025390.4	900590.9	3698.2	52.1	2540.4	6290.7
23	407	667510.2	6	138	121	1189757.1	1103227.4	893955.3	3508.6	0.0	2343.2	5851.8
24	293	437810.8	7	119	110	860021.6	856513.5	843693.6	1837.6	0.0	827.0	2664.5
25	135	154037.8	6	87	86	421108.2	421108.2	420898.9	387.1	0.0	23.9	411.0
26	296	435794.4	4	113	100	1114796.9	911158.8	815987.2	2898.3	0.0	1246.9	4145.3
27	363	529716.3	5	113	99	959096.3	910464.9	813551.1	3239.9	0.0	1874.3	5114.2
28	300	444870.9	6	121	109	735051.8	733956.2	687627.5	2483.4	0.0	1131.4	3614.8
29	206	267254.1	6	99	94	559744.5	559744.5	551911.2	1276.8	0.0	607.7	1884.5
30	175	228001.7	1	92	91	546944.5	546944.5	544404.6	385.8	0.0	19.1	404.8
31	107	108021.6	8	77	77	380782.1	380782.1	380782.1	25.9	0.0	0.0	25.9
32	52	47637.8	0	46	46	157985.3	157985.3	157985.3	3.0	0.0	0.0	3.0
33	35	25730.4	1	33	33	135351.7	135351.7	135351.7	3.0	0.0	0.0	3.0
34	17	13389.5	1	20	20	66828.6	66828.6	66828.6	0.6	0.0	0.0	0.6
35	5	3262.3	0	6	6	12829.1	12829.1	12829.1	0.0	0.0	0.0	0.0
36	1	769.8	0	2	2	4088.2	4088.2	4088.2	0.0	0.0	0.0	0.0
Overall	**9248**	**13917831.3**	**0**	**3583**	**3218**	**27089314.7**	**24783365.0**	**22370455.5**	**78261.8**	**153.0**	**42834.2**	**121248.9**

Table E.3. Daily Simulation Results for Scenario 4.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	253	404287.6	14	88	72	567036.0	552004.2	526113.0	3196.6	0.0	2097.5	5294.2
2	271	445000.1	53	116	101	705107.2	652795.8	602796.8	2997.1	0.0	1695.3	4692.4
3	218	322537.5	97	124	117	583404.1	580882.0	576137.9	1471.2	0.0	820.3	2291.5
4	124	140465.6	70	102	100	363679.2	363502.4	362313.0	417.0	0.0	255.2	672.1
5	304	451940.4	57	132	117	734024.1	721170.8	665110.9	3240.3	0.0	1908.7	5149.0
6	324	507305.4	68	136	119	838852.9	812752.2	707475.0	3821.1	0.0	2243.2	6064.2
7	395	591668.8	73	147	128	942754.6	916529.0	825083.1	3883.8	0.0	2598.6	6482.4
8	355	564953.2	84	142	126	909694.9	861211.9	766803.7	3336.8	0.0	2097.4	5434.2
9	369	576564.6	79	141	126	946206.7	893665.6	849462.8	3623.5	0.0	2501.1	6124.6
10	244	331722.2	72	131	123	672626.1	647718.5	638341.7	1741.7	0.0	927.8	2669.4
11	112	117692.8	62	102	100	368465.6	368465.6	368122.0	425.9	0.0	281.9	707.8
12	300	442407.0	48	126	111	757496.9	753868.0	715383.7	2958.2	0.0	2170.8	5129.0
13	370	580357.1	46	134	117	1009678.3	870405.3	765670.6	3819.0	0.0	2202.5	6021.5
14	382	601174.0	74	135	118	959991.7	885443.8	725548.5	3615.9	0.0	2247.1	5863.0
15	370	570183.5	71	139	120	1111236.3	842457.1	676708.5	4047.1	127.7	1960.6	6135.4
16	404	604004.3	78	135	115	1092746.0	913725.7	746025.2	4380.0	0.0	2848.2	7228.2
17	317	468725.7	66	129	114	871124.8	826596.0	815174.1	3011.0	0.0	1446.9	4457.9
18	126	143855.6	62	93	91	387798.3	386578.4	384274.0	467.8	0.0	213.1	680.8
19	314	481224.7	49	124	110	784141.9	783173.1	765965.7	2908.3	0.0	1693.0	4601.3
20	381	560207.6	71	145	127	1104157.3	913710.2	743005.4	3849.6	0.0	2713.9	6563.5
21	417	657178.6	80	144	125	1031689.3	972007.6	788180.1	4167.4	0.0	3069.6	7237.1
22	387	574659.2	98	150	134	1155587.9	968886.4	864847.0	3407.8	56.3	2397.9	5862.0
23	428	662444.3	68	149	129	1109183.1	1053496.4	904743.2	4078.7	0.0	3199.8	7278.5
24	279	438505.1	88	129	119	767871.9	760938.0	750398.7	1915.3	0.0	1215.4	3130.8
25	144	166847.2	72	107	106	455795.9	451039.1	448555.7	297.4	0.0	183.0	480.4
26	306	428424.8	52	124	112	878579.2	781725.7	703086.7	2862.4	0.0	1649.9	4512.3
27	351	518471.5	75	127	112	841043.6	836525.3	751032.4	3239.9	0.0	1867.3	5107.3
28	302	430684.1	86	129	116	784870.2	714698.0	669404.4	2804.8	0.0	1703.5	4508.3
29	218	264158.7	72	107	100	555283.8	555283.8	550808.3	1467.9	0.0	956.4	2424.2
30	187	232360.5	68	101	98	497758.1	497758.1	495768.0	756.7	0.0	44.2	801.0
31	107	115461.3	65	90	90	320941.4	320941.4	320941.4	24.8	0.0	0.0	24.8
32	68	66114.1	37	66	66	198044.2	198044.2	198044.2	30.3	0.0	0.0	30.3
33	45	46923.7	17	49	49	136190.6	136190.6	136190.6	3.8	0.0	0.0	3.8
34	24	22559.8	7	27	27	64902.6	64902.6	64902.6	1.3	0.0	0.0	1.3
35	16	9655.5	1	20	20	25670.2	25670.2	25670.2	0.2	0.0	0.0	0.2
36	5	3003.9	0	8	8	7382.6	7382.6	7382.6	0.1	0.0	0.0	0.1
Overall	**9217**	**13543730.2**	**0**	**4048**	**3663**	**24541017.5**	**22892145.5**	**20905471.6**	**82270.8**	**184.0**	**51210.2**	**133665.0**

Table E.4. Daily Simulation Results for Scenario 5.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	252	406953.3	15	88	72	560743.0	559195.4	537861.2	3296.2	0.0	1218.8	4515.0
2	267	446683.4	54	115	102	753995.1	655270.1	606686.0	2737.0	0.0	1695.6	4432.5
3	209	323242.4	80	124	119	557952.4	553209.8	548645.7	1118.5	0.0	782.3	1900.7
4	104	122950.7	61	101	100	311914.5	311914.5	311836.8	222.6	0.0	24.5	247.1
5	300	455515.9	50	131	116	745351.9	724828.1	652186.3	2951.9	0.0	1963.1	4915.0
6	323	508417.0	55	136	118	864288.3	812541.9	708268.2	3728.1	0.0	2254.3	5982.4
7	375	584716.0	63	147	127	891757.6	874084.9	783939.9	4005.2	0.0	2505.5	6510.7
8	340	553040.2	59	139	125	878583.5	831192.9	731807.0	2887.0	0.0	1713.9	4600.9
9	350	571890.7	79	141	124	882189.4	845796.4	775259.4	3680.4	0.0	2614.1	6294.6
10	227	322379.6	83	128	122	627506.6	621208.0	612005.0	1287.7	0.0	716.8	2004.5
11	101	111173.4	59	105	104	329161.2	329153.7	328962.5	210.1	0.0	150.5	360.6
12	292	438958.3	41	130	114	732148.3	720483.5	678937.4	3150.4	0.0	1705.3	4855.7
13	369	590646.4	42	134	117	1027288.3	855712.6	757197.7	3688.0	70.8	1875.4	5634.2
14	363	591213.8	59	135	119	904500.6	835980.2	674502.7	3602.1	0.0	1941.8	5544.0
15	361	569223.8	66	137	118	1007431.0	840535.3	673741.7	3963.7	70.0	1881.5	5915.2
16	396	586762.3	56	135	116	1086746.3	885873.1	711707.9	4163.9	146.2	2327.0	6637.1
17	295	451388.6	51	128	117	784135.5	753298.4	738228.4	2337.4	0.0	1164.5	3501.9
18	114	127393.0	59	98	97	347636.3	347636.3	347322.0	383.9	0.0	181.5	565.4
19	298	469898.7	44	124	110	817343.7	771756.6	751830.5	3065.3	0.0	1432.0	4497.3
20	384	561181.0	59	146	128	941023.2	892607.7	797695.1	3864.6	0.0	2609.4	6474.1
21	410	655884.3	64	145	126	1070814.1	929965.7	803928.3	4176.1	0.0	2825.2	7001.3
22	363	551749.5	86	148	134	1166301.1	909891.4	806790.5	3198.9	102.1	1864.6	5165.7
23	403	638491.5	70	149	130	1065571.9	1024999.3	823788.7	3756.7	0.0	2625.2	6381.9
24	275	456190.5	87	127	117	779578.7	770718.4	760717.0	1909.8	0.0	1196.2	3106.1
25	127	141468.6	52	105	104	404823.0	404823.0	404238.4	239.1	0.0	181.9	421.0
26	304	441631.8	41	125	112	781904.9	761596.3	683084.4	2713.6	0.0	1694.3	4407.9
27	342	512632.3	58	125	112	848367.4	808892.1	729903.5	2955.9	0.0	2070.8	5026.7
28	282	415563.9	67	129	119	706558.0	680665.4	634396.8	2338.5	0.0	1427.5	3766.1
29	208	269198.0	60	105	100	531160.2	525677.8	521701.5	987.9	0.0	624.4	1612.3
30	174	215659.7	51	102	100	469863.4	469863.4	467771.9	453.4	0.0	33.9	487.4
31	99	106886.4	52	91	91	290313.3	290313.3	290313.3	29.1	0.0	0.0	29.1
32	60	60349.2	30	69	69	170626.8	170626.8	170626.8	3.0	0.0	0.0	3.0
33	47	52493.7	19	54	54	140984.6	140984.6	140984.6	5.1	0.0	0.0	5.1
34	26	23062.3	10	29	29	75632.0	75632.0	75632.0	1.0	0.0	0.0	1.0
35	19	12256.4	2	26	26	33497.0	33497.0	33497.0	0.3	0.0	0.0	0.3
36	8	6040.9	0	10	10	15551.9	15551.9	15551.9	0.1	0.0	0.0	0.1
Overall	**8867**	**13353187.6**	**0**	**4061**	**3698**	**23603245.1**	**22035977.9**	**20091548.0**	**77112.7**	**389.1**	**45301.7**	**122803.5**

Table E.5. Daily Simulation Results for Scenario 6.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	252	407759.5	17	88	71	606180.7	587546.4	566768.3	3233.3	0.0	1574.8	4808.1
2	268	456494.6	51	108	95	770266.8	679196.8	631902.1	2686.9	0.0	1484.4	4171.3
3	201	320199.6	69	114	108	553000.9	550350.3	545182.7	1296.6	0.0	785.1	2081.6
4	88	111949.6	39	92	92	273398.4	273398.4	273398.4	44.5	0.0	0.0	44.5
5	293	445222.8	34	130	118	785983.3	714650.8	639177.9	2365.9	0.0	1832.3	4198.3
6	319	518613.8	54	132	115	896234.2	836800.9	727824.0	3362.6	0.0	2120.0	5482.6
7	366	581129.4	68	142	125	977581.2	902923.5	744938.5	3444.5	0.0	2340.9	5785.4
8	335	549015.4	57	138	125	929768.9	867822.3	716925.0	2646.6	0.0	1732.2	4378.7
9	351	580104.4	52	136	119	933443.4	845128.9	750533.8	3532.5	0.0	2319.8	5852.4
10	213	319330.9	59	125	120	601407.2	596562.6	588345.7	1086.7	0.0	685.2	1772.0
11	92	99102.3	38	106	106	300579.1	300579.1	300579.1	67.7	0.0	0.0	67.7
12	284	427332.0	29	126	113	793694.0	716245.5	627482.4	2747.6	0.0	1804.2	4551.7
13	366	591081.5	48	132	115	1033416.0	866804.6	743406.5	3594.6	92.4	1879.9	5566.9
14	358	600124.4	45	129	114	951549.6	822881.9	694240.3	3272.0	0.0	2096.3	5368.3
15	351	558118.8	61	128	112	959796.6	855610.6	685397.7	3522.0	0.0	2095.1	5617.0
16	384	595058.0	55	129	112	1078401.5	884174.9	694612.0	3579.6	117.6	2220.5	5917.6
17	285	451445.2	52	121	110	749923.1	731308.7	721086.6	2175.6	0.0	961.0	3136.6
18	88	102462.3	44	90	90	276592.5	276592.5	276592.5	31.4	0.0	0.0	31.4
19	284	468507.7	33	114	102	771753.4	727213.2	707721.3	2610.3	0.0	980.5	3590.8
20	378	563216.6	62	143	128	1010224.2	895612.5	727906.2	3379.9	0.0	2133.0	5512.9
21	412	665605.8	44	138	120	1060853.2	952258.8	770908.1	3993.7	0.0	2365.8	6359.5
22	347	539609.8	60	142	129	1065384.2	915490.6	803632.6	2901.2	0.0	1891.1	4792.3
23	391	633730.0	51	143	127	1065918.3	1006295.7	876015.2	3276.0	0.0	2118.6	5394.6
24	256	430346.8	78	126	120	755382.7	745964.0	736357.3	1399.4	0.0	636.9	2036.2
25	111	146212.7	55	111	111	372852.4	372852.4	372852.4	118.5	0.0	0.0	118.5
26	287	431578.5	36	125	113	860930.4	736948.0	657939.6	2448.3	0.0	1664.8	4113.1
27	339	510976.8	54	121	110	848646.4	789647.5	707989.1	2434.2	0.0	1744.2	4178.4
28	284	422931.5	48	127	119	735566.2	693952.9	650497.1	1792.9	0.0	1345.6	3138.5
29	187	257698.6	52	99	95	509224.1	507179.9	502695.3	1120.0	0.0	577.1	1697.0
30	165	222552.4	33	91	90	454454.0	454454.0	453958.4	481.9	0.0	21.0	502.9
31	92	92294.0	47	92	92	289772.9	289772.9	289772.9	21.3	0.0	0.0	21.3
32	56	59768.0	27	69	69	171725.0	171725.0	171725.0	4.2	0.0	0.0	4.2
33	52	52879.9	5	58	58	134149.0	134149.0	134149.0	2.0	0.0	0.0	2.0
34	26	23795.1	9	33	33	66187.8	66187.8	66187.8	1.1	0.0	0.0	1.1
35	19	12212.9	3	27	27	38472.7	38472.7	38472.7	0.3	0.0	0.0	0.3
36	6	3119.6	1	11	11	9820.9	9820.9	9820.9	0.1	0.0	0.0	0.1
37	0	0.0	1	1	1	1684.6	1684.6	1684.6	0.0	0.0	0.0	0.0
38	0	0.0	1	1	1	1684.6	1684.6	1684.6	0.0	0.0	0.0	0.0
Overall	**8586**	**13251580.9**	**1**	**3938**	**3616**	**23695904.7**	**21819945.6**	**19610363.6**	**68675.7**	**210.0**	**41410.2**	**110295.9**

Table E.6. Daily Simulation Results for Scenario 7.

Day	N_{trip}	C^F	N_{uns}	N_{sub}	N_{sub}^{opt}	C_{DPM}^{tot}	C_{CGH}^{tot}	L^{tot}	T_{DPM}	T_{PIH}	T_{CGH}	T_{tot}
1	254	406341.5	16	88	73	612715.2	601559.1	582615.4	3174.2	0.0	1376.2	4550.4
2	269	448195.3	54	108	96	731021.5	705678.0	662048.5	2605.4	0.0	1494.9	4100.3
3	203	322797.8	76	110	104	589873.4	584316.5	579470.1	1346.4	0.0	674.5	2020.9
4	88	119847.3	47	91	91	305608.4	305608.4	305608.4	16.9	0.0	0.0	16.9
5	286	433175.3	38	130	119	801064.6	730162.5	656600.1	2162.6	0.0	1619.9	3782.6
6	312	509611.4	69	131	116	899721.3	860494.4	754688.4	2948.0	0.0	1860.4	4808.4
7	364	562888.2	88	143	127	1076780.4	944879.9	790552.7	3583.9	0.0	2290.7	5874.6
8	329	554567.4	79	139	128	1038260.9	912336.1	760991.2	2263.6	0.0	1582.0	3845.6
9	352	583498.0	69	138	123	984827.5	905975.4	828061.8	3186.9	0.0	2128.2	5315.1
10	213	313825.4	73	129	124	640039.1	633836.4	625872.2	1214.1	0.0	574.8	1789.0
11	75	79437.5	68	113	113	325849.5	325849.5	325849.5	4.8	0.0	0.0	4.8
12	270	421568.9	56	134	122	766286.7	749449.1	702292.5	2571.0	0.0	1639.8	4210.9
13	360	586699.3	53	138	121	1104260.4	917283.2	800215.2	3471.9	84.0	1674.9	5230.8
14	358	586572.8	72	139	123	1013818.1	892637.9	762302.3	3343.1	0.0	1916.3	5259.4
15	356	569777.5	73	137	122	1089130.9	912844.3	694533.2	3241.9	71.7	1778.0	5091.6
16	373	581165.7	85	134	118	1099164.7	914188.6	743974.8	3430.3	0.0	2188.6	5618.9
17	282	449165.5	75	128	118	805950.7	797982.6	788868.1	2061.4	0.0	966.8	3028.3
18	73	91522.5	74	106	106	333562.8	333562.8	333562.8	15.9	0.0	0.0	15.9
19	287	465984.2	47	125	112	780877.3	779971.8	764382.8	2638.5	0.0	944.6	3583.1
20	375	559614.7	85	150	132	1109651.5	955010.5	786699.2	3713.9	0.0	2150.8	5864.8
21	406	662435.6	75	145	129	1102459.7	1025917.9	850768.4	3715.6	0.0	2214.7	5930.4
22	347	535697.6	96	146	134	1261629.8	1000990.0	886850.9	2823.0	41.3	1654.4	4518.7
23	388	637452.8	78	149	133	1223680.2	1092843.6	937371.8	3298.2	0.0	1899.0	5197.2
24	236	407477.6	116	130	123	832446.4	802051.1	794303.8	1446.5	0.0	836.4	2282.9
25	98	118627.3	89	120	119	460076.9	460076.9	460034.7	201.8	0.0	1.6	203.4
26	279	438439.4	61	133	123	968456.1	844592.9	767567.5	2257.5	0.0	1543.2	3800.7
27	333	501956.6	86	132	123	979590.4	889575.2	799217.0	2253.2	0.0	1579.6	3832.8
28	272	411305.2	95	140	132	815046.1	811637.7	764509.1	1733.3	0.0	1282.3	3015.6
29	192	271044.1	93	120	117	690801.6	664582.4	660544.1	943.6	0.0	394.7	1338.4
30	166	225264.9	65	107	107	585322.6	585322.6	585322.6	164.5	0.0	0.0	164.5
31	71	72621.3	89	98	98	372182.4	372182.4	372182.4	13.0	0.0	0.0	13.0
32	46	51019.5	66	86	86	305753.7	305753.7	305753.7	2.5	0.0	0.0	2.5
33	48	43432.3	51	84	84	279808.0	279808.0	279808.0	2.4	0.0	0.0	2.4
34	27	22973.7	48	59	59	209143.9	209143.9	209143.9	0.7	0.0	0.0	0.7
35	16	9284.4	45	52	52	175638.5	175638.5	175638.5	0.6	0.0	0.0	0.6
36	7	4119.2	44	40	40	160760.7	160760.7	160760.7	0.5	0.0	0.0	0.5
37	0	0.0	44	30	30	150611.0	150611.0	150611.0	0.4	0.0	0.0	0.4
38	0	0.0	44	30	30	150611.0	150611.0	150611.0	0.4	0.0	0.0	0.4
Overall	**8411**	**13059407.3**	**44**	**4312**	**4007**	**26832483.5**	**24745726.3**	**22560188.3**	**65852.6**	**197.0**	**38267.7**	**104317.3**

APPENDIX F

Table F.1. Algorithm Parameters for DVAP.

Parameter	Description	Value
RT_r	Rest time between routes	1 hr
D_v	Deadline for dedicated vehicles to return to their home locations	60 hrs