TOBB UNIVERSITY OF ECONOMICS AND TECHNOLOGY

GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OPTIMIZATION MODELS AND HEURISTIC SOLUTION METHODS FOR THE INTEGRATED FLEET SIZING AND REPLENISHMENT PLANNING PROBLEM WITH CANDIDATE DELIVERY PATTERNS

PHD THESIS

Duygu AGHAZADEH

Industrial Engineering Department

Supervisor: Assoc. Prof. Dr. Kadir ERTOGRAL

SEPTEMBER 2023

THESIS STATEMENT

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

.

Duygu AGHAZADEH

SIGNATURE

ÖZET

Doktora Tezi

"ADAY TESLİMAT PATERNLERİ İLE ENTEGRASYONLU FİLO BÜYÜKLÜĞÜ VE İKMAL PLANLAMA PROBLEMİ İÇİN OPTİMİZASYON MODELLER VE SEZGİSEL ÇÖZÜM YÖNTEMLERİ"

Duygu AGHAZADEH

TOBB Ekonomi ve Teknoloji Üniveritesi
Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Anabilim Dalı

Doç. Dr. Kadir ERTOĞRAL

Tarih: Eylül 2023

Bu tez çalışmasında entegre filo büyüklüğü belirleme ve ikmal planlaması problemi için matematiksel modeller üretilip, farklı çözüm yöntemleri önerilmiştir. Tezin ilk bölümünde, önceden belirlenmiş ikmal frekansları ve satıcı yönetimi politikası altında ikmal planlaması ve filo büyüklüğünün belirlenmesi amacıyla bir model oluşturulmuştur. Bahsedilen modelin çözümü için iki farklı meta sezgisel çözüm tekniği, Yaklaşık Dinamik Programlama ve Problem Alanı Arama, önerilip, gerçek hayat verilerinden esinlenerek üretilen veri setleri üzerine test edilmiştir. İkinci bölümde ise, ilk bölümdeki modelin daha genelleştirilmiş hali ele alınarak yeni bir bakış açısıyla matematiksel bir model geliştirilip, sezgisel çözüm yöntemleri önerilmiştir. Bu bölümde önceden belirlenmiş tekrarlanan ikmal frekansları yerine ikmal paternleri ele alınmıştır. Ek olarak, talep sezonsallığı ve araç kiralama opsiyonları göz önünde bulundurulmuştur. Problemi çözmek için Sabitle ve optimize et sezgiselinin üç farklı versyonu tasarlanıp, üretilen veriler üzerine test edilmiştir. Sonuçların kalitesi ve çözüm süreleri önerilen çözüm tekniklerinin etkili olduğunu göstermiştir.

Anahtar Kelimeler: İkmal planlama, Filo büyüklüğü belirleme, Sezgisel metot, Meta sezgisel metot

ABSTRACT

Doctor of Philosophy

OPTIMIZATION MODELS AND HEURISTIC SOLUTION METHODS FOR THE INTEGRATED FLEET SIZING AND REPLENISHMENT PLANNING PROBLEM WITH CANDIDATE DELIVERY PATTERNS

Duygu AGHAZADEH

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Industrial Engineering Science Programme

Supervisor: Assoc. Prof. Dr. Kadir ERTOGRAL

Date: September 2023

In this thesis work, two mathematical models were developed to formulate two basic and extended versions the integrated fleet sizing and replenishment planning problem, and various solution methods were proposed. In the first section of the thesis, a model was created for replenishment planning and fleet size determination under pre-defined replenishment frequencies and vendor management policy. For solving the mentioned model, two different metaheuristic solution techniques, namely Approximate Dynamic Programming, and Problem Space Search, were proposed and tested on datasets inspired by real-life data. In the second section, a more generalized version of the model in the first section is considered, and a mathematical model is developed from a new perspective, with intuitive solution methods proposed. Replenishment patterns were considered instead of pre-defined recurring replenishment frequencies. Additionally, demand seasonality and vehicle renting options were taken into account. To solve the problem, three different versions of the Fix and Optimize heuristic were designed and tested on generated data. The quality of the results and solution times demonstrated the effectiveness of the proposed solution techniques.

Keywords: Replenishment planning, Fleet sizing, Heuristic methods, Meta-heuristic methods

ACKNOWLEDGMENTS

To begin, my profound gratitude goes to my supervisor, Assoc. Prof. Dr. Kadir ERTOGRAL, for his invaluable mentorship, direction, constructive feedback, and, most notably, his unwavering support, which extended beyond the scope of this study to encompass my entire graduate journey. My appreciation extends to the examining committee members, including Prof. Dr. Fulya ALTIPARMAK, Assoc. Prof. Dr. Aysegul ALTIN KAYHAN, Asst. Prof. Dr. Salih TEKIN, and Prof. Dr. Diyar AKAY, for their invaluable insights and substantial contributions.

I want to express my gratitude to my wonderful family for their enduring support, affection, and motivation throughout my lifetime.

I extend my heartfelt appreciation to my husband, Reza AGHAZADEH, whose unwavering understanding and support were instrumental in helping me navigate through this challenging phase of my life.

I wish to express my sincere appreciation to my daughter, Ela AGHAZADEH, for her consistent motivation and exceptional understanding throughout my doctoral studies.

CONTENTS

Pages

TABLE OF FIGURES

Pages

TABLE OF TABLES

1 INTRODUCTION

This thesis addresses two notable challenges within the realm of fleet sizing and replenishment planning. The first and second sections suggest simple and extended versions of the problem with mathematical models and metaheuristic solution techniques for the solution of integrated fleet sizing and replenishment problem when a set of predetermined delivery frequencies is available. Within this context, our focus lies on the VMI framework, wherein the distributor or supplier has the freedom to decide delivery timing and quantities. The implementation of a VMI program is a common practice and it offers numerous advantages for all participants in the supply chain.

Logistics costs are significantly affected by three main factors: the ownership cost of the fleet, the routing cost of vehicles, and inventory related costs. Fleet ownership cost is contingent upon fleet size and composition which are strategic decisions. On the other hand, routing costs are the outcome of daily routing plans based on assignments of vehicles to customers. Hence, the composition of the fleet plays a pivotal role in optimizing logistics costs. Thus, finding optimal or highly efficient solutions for the strategic problems of fleet sizing and composition becomes a critical decision in the realm of logistics. This strategic problem must consider routing and inventory replenishments in an aggregate manner, which are operational problems. This framework is what we considered in the models suggested in this thesis.

In Dastjerd & Ertogral (2019) [14] we stated that optimizing a distribution system is dependent on optimizing its cost components as a whole. Hence, the objective function for the suggested problem includes fixed replenishment, inventory holding, routing, and vehicle ownership costs. We aim at determining the fleet size and composition along with assigning customers to a specific vehicle-frequency, or vehicle-delivery pattern, combination. In the first section, we tackle the same problem as the one in Dastjerd & Ertogral (2019) [14], and we both suggest a mathematical model and the following two different heuristic solution methods;

1. Approximate Dynamic Programming (ADP) with fix and optimize heuristic

2. Problem Space Search (PSS) with fix and optimize heuristic

In the second part of the thesis, we tackle a more extensive variation of the issues discussed in the first section. Initially, we constructed a mixed integer programming model to manage challenges encompassing the integration of fleet sizing and replenishment planning, all while taking into account predetermined delivery frequencies. The assumptions were made that demands from customers dispersed geographically were constant and not influenced by seasons. Also assumed is that the candidate delivery patterns are regular and repetitive. The replenishment procedures were executed utilizing a diverse array of owned vehicles, with the cost of routing determined by the number of customers replenished via specific vehicle types, delivery frequencies, and the per-kilometer cost of each assigned vehicle type.

Transitioning to the issue explored in the thesis's subsequent section, it centers around an assemblage of stores spread across a clearly define ed geographic region. In this context, the demand for these stores is consistent but demonstrates seasonal patterns. The fleet employed consists of both owned and rented heterogeneous trucks. The approximation of routing costs is accomplished through clustering, a technique that involves allocating stores to seed points. The replenishment processes are executed utilizing appropriate delivery patterns designed not only to fulfill store demands but also to minimize the comprehensive expenses associated with transportation, delivery pattern allocation (inventory costs), vehicle ownership/rental, and routing.

As solution method, three versions of fix and optimize heuristic are suggested:

1. FO with store-based subproblems
2. FO with cluster-based subproblems
3. FO with delivery day-based subproblems

As it is obvious from the name of the FO versions, they differ in the subproblem generation criteria.

We give motivation, related literature, problem definition, steps for solution techniques, and numerical analysis for each section in the subsequent parts of the thesis.

2 BASIC PROBLEM WITH SIMPLE DELIVERY PATTERNS

2.1 Introduction

Optimizing the supply chain is of utmost importance as it leads to cost reduction for both businesses and customers, fostering win-win situations. In today's ever-changing and uncertain business landscape, companies require robust systems that can adapt dynamically. As part of effective supply chain management practices, companies also prioritize reducing logistics-related costs. Therefore, the design of the distribution system must ensure that logistics operations costs are minimized while maintaining an acceptable service level.

According to Taleizadeh et al. (2020) [52], it is essential to acknowledge that the best course of action for one side, either the supplier or the consumer, may not always align with the other. Consequently, a systematic approach is needed to address this issue. Vendor Managed Inventory (VMI) is one such systematized approach that offers benefits to both suppliers and customers in the supply chain. The concept of VMI emerged from a study by John (1958) [29], which debated who should be responsible for maintaining inventory. VMI is a well-known system where the supplier determines the timing and quantity of deliveries, while ensuring that the customer's inventory remains within specified minimum and maximum limits. This provides suppliers with better demand information and enhances operational efficiency in distribution while significantly reducing inventory control costs for customers.

In the problems and models, we tackle in the thesis, we focus on the VMI setting, where the distributor or supplier has the freedom to choose the timing and quantity of deliveries. Implementing a VMI program yields numerous advantages for the supply chain and all its participants.

Two significant components of logistics costs are the fleet ownership cost and routing cost of vehicles. Fleet ownership cost depends on the fleet size and composition, making it a strategic decision. On the other hand, routing costs result from daily routing plans based on customer-vehicle assignments. Therefore, an effective fleet composition greatly influences the optimization process to minimize logistics cost significantly. Hence, arriving at optimal or near-optimal solutions for the strategic fleet sizing and composition problem becomes a crucial logistical decision.

Strategic fleet sizing primarily focuses on choosing the appropriate fleet composition and size to increase the firm's profitability. According to Koç et al. (2016) [30], fleet sizing decisions take various factors into account, including the types and quantity of vehicles to be owned, which are influenced by market considerations such as shipping costs, rates, and expected demand.

This section of the thesis introduces two different solution algorithms for the integrated fleet sizing and replenishment planning problem, namely approximate dynamic programming (ADP) and Problem Space Search (PSS) metaheuristic. ADP algorithm employs a Fix and Optimize (FO) method to approximate the objective function value at each iteration of the dynamic programming for the partial problem. We adopt the model suggested in Dastjerd & Ertogral (2019) [14] where the authors proposed a fix and optimize heuristic for the same problem. However, in this thesis, we present an enhanced solution approach, framing them as an ADP and a problem space search heuristic.

The method employed in this research involves a fix and optimize-based approximate dynamic programming approach, supplemented by a final improvement step. ADP is a well-established technique used in various problems, as discussed in detail in the literature. One of the key advantages of ADP is its ability to significantly reduce computational time by solving partial problems in a forward recursion formulation using problem-specific or general heuristics instead of seeking optimal solutions. In our approach, we use fix and optimize as the heuristic for the partial problems in the ADP, and we further enhance efficiency with a look-ahead strategy. Additionally, an improvement stage is executed at the end of the process.

The other proposed solution method, called Problem Space Search (PSS), introduces a novel metaheuristic technique that differs from the conventional approach of exploring the "solution space." Instead of solely searching for solutions within a designated solution space, PSS adopts a problem space search approach. In each iteration of a problem-specific heuristic, multiple solutions are evaluated by executing specific algorithms multiple times. Each execution uses temporarily perturbed data or perturbed heuristic parameters as input. This novel strategy aims to create a distinct search space within the problem space, leading to more efficient and effective solutions.

The core concept behind PSS is to create artificial "neighbor" problems by temporarily modifying the input data of the original problem or adjusting parameters in the embedded heuristic. The main idea is that by finding heuristic solutions to similar problems, there is a higher chance of obtaining solutions that are close to the optimal solution for the original problem. PSS aims to broaden the exploration scope by considering variations of the original problem through data perturbation, leading to the discovery of potentially high-quality solutions within a wider problem space. Our version of PSS incorporates a fixed and optimize heuristic. In PSSI, we perturb the cost data, while in PSSII, we randomize the order of subproblems tackled in the fix and optimize process. We generate multiple instances with distinct characteristics and evaluate the performance of the proposed metaheuristic through a numerical study using the generated dataset.

The structure of this part is as follows: the subsequent section reviews the relevant literature, while Section 2.3 provides an explanation of the mathematical model and the problem. The steps for the Approximate Dynamic Programming and Problem Space Search methods are outlined in Sections 2.4 and 2.5. Section 2.6 delves into the dataset's structure and presents the performance analysis of the heuristic. Finally, in Section 2.7, we offer our concluding remarks on this section.

2.1.1 Motivation

As discussed in introduction section, inventory and transportation costs play a key role in minimizing the total cost of a distribution system. On the other hand, having an efficient distribution system in terms of costs and performance, needs caring for all of the components of the system as a whole. Hence, this part of the thesis focuses on a problem which integrates fleet sizing and inventory decisions in a single problem and aims at bringing quality solution to this novel problem.

2.2 Literature

In this section, we present several pertinent research works. We provide an overview of the literature, which can be categorized into five main focus groups: fleet sizing, delivery problems with specified frequencies, studies related to approximate dynamic programming, Problem Space Search and applications of the fix and optimize method.

2.2.1 Fleet sizing problem

In this section, we discuss various studies directly related to fleet sizing. Desrochers & Verhoog (1991) [15] address the challenge of replenishing customers with known demands from a central depot. They tackle both fleet composition and routing decisions, utilizing a version of the saving type heuristic based on consecutive route fusions. Sayarshad & Ghoseiri (2009) [42] explore a novel mathematical formulation for fleet dimension optimization and freight car assignment. Their proposed solution approach relies on simulated annealing, which employs three techniques to escape local optima: inferior solutions, solution space neighborhood search, and acceptance probability.

Liu et al. (2009) [34] focus on the fleet sizing and vehicle routing problem involving a set of nonhomogeneous vehicles. They present a heuristic based on a genetic algorithm, demonstrating its effectiveness on benchmark instances compared to existing literature solutions in terms of solution quality and computation times. Żak et al. (2011) [57] tackle the fleet sizing problem in a road freight transportation firm with a set of heterogeneous vehicles. Their two-phased heuristic employs innovative software to produce a collection of Pareto-optimal results, which the decision maker then evaluates using their preference model.

Aziez et al. (2022) [4] propose methods to enhance transportation operations' efficiency in hospitals by utilizing automated guided vehicles to save labor and improve efficiency. They develop a mathematical formulation and a powerful metaheuristic for this purpose. Sun et al. (2021) [51] address an operational fleet dimensioning problem that considers the cost of fuel. Their problem is a modified version of fleet dimensioning and mix vehicle routing, and they introduce the economy traveling distance (ETD) method to determine the optimal travel distances for each type of vehicle, taking into account their fuel consumption rates.

2.2.2 Delivery with given frequencies

Speranza & Ukovich (1994) [47] explore the distribution problem involving candidate frequencies, accommodating both indivisible and divisible demand scenarios through their integer and mixed integer formulations. They propose a two-step heuristic using modified dominance principles as the solution strategy. In a separate work, Speranza & Ukovich (1996) [46] utilize a branch and bound approach to tackle the distribution of various commodities from a single origin to multiple consumers based on preset replenishment schedules.

Bertazzi et al. (1997) [8] present a strategy for resolving a problem with predefined frequencies, where items are delivered from a single origin to multiple destinations, considering transportation and inventory costs, similar to our study. However, they use continuous frequencies in their replenishment decision-making approach, distinguishing it from our case. Additionally, they do not account for fixed costs per delivery. Their sequential heuristic method involves building a mixed integer programming model for single link problems and solving a shortest path problem to make decisions on frequency-truck assignment and cargo transportation percentage.

In another study, Bertazzi & Speranza (1999) [7] analyze a different situation involving several products, one origin, a few intermediary nodes, and a destination, with predetermined candidate delivery frequencies. They propose four distinct heuristic techniques, including dividing sequences into links and handling each link independently, applying the same shipping percentages to all links, and two heuristics based on discrete versions of the EOQ formulations. They also explore a dynamic programming-based technique, considering links as stages and the collection of delivery frequencies on preceding links as states.

Examining the problem of transporting multiple items from an origin to a single destination at a predefined frequency, Bertazzi et al. (2000) [9] employ both heuristic and exact methods. The heuristic methods are developed based on the EOQ formula, while the exact method involves a modified version of the branch and bound algorithm. In a more complex production-distribution network under a Vendor Managed Inventory (VMI) setting, Bertazzi et al. (2005) [6] aim to regularly produce and distribute products using a fleet of trucks. They propose hierarchic heuristics, solving the production and distribution subproblems consecutively, with the production results influencing the distribution phase.

2.2.3 Approximate dynamic programming

Approximate Dynamic Programming (ADP) is introduced as an algorithmic technique to address the issue of the curse of dimensionality, commonly used in stochastic problems. ADP employs a heuristic approach to approximate the objective function value. The literature covers various problems effectively solved by ADP, including scheduling problems, fleet management problems, knapsack problem variations, vehicle routing problems, replenishment planning problems, and allocation problems.

Bertsimas & Demir (2002) [10] propose an ADP approach for the multidimensional knapsack problem, estimating the objective function value using non-parametric and parametric techniques. They claim that their heuristic approach provides high-quality solutions, outperforming ready-to-use software like CPLEX in terms of solution quality and computational time. Hua et al. (2006) [25] consider the convex Quadratic Knapsack Problem (QKP) and offer two approaches to estimate the objective function value, achieving high-quality solutions for large-scale QKPs. Perry & Hartman (2009) [37] use ADP to solve a dynamic, stochastic knapsack problem, combining simulation with deterministic dynamic programming to solve longer-term problems effectively.

Topaloglu (2005) [54] employs an ADP-based technique to optimize distribution activities for a business, involving production across multiple facilities and distribution to various locations. They utilize concave approximations to estimate the objective function value, demonstrating the effectiveness of their ADP version. Simao et al. (2009) [45] create a model for a large truckload motor carrier company in the US, using Monte Carlo simulation and machine learning to approximate the value function and accurately assess the marginal value of different types of drivers.

ADP is also applied in other areas, such as the surgical scheduling problem, patient admission problem, and machine scheduling problem (Astaraky & Patrick (2015) [3], Silva & de Souza (2020) [44], Hulshof et al. (2016) [27], and Ronconi & Powell (2010) [39], respectively).

2.2.4 Problem space search metaheuristic

Storer et al. (1992) [50] introduced the Problem Space Search (PSS) metaheuristic, which involves applying a problem-specific heuristic repeatedly in the problem space using local search. PSS has been widely adopted in various studies. For example, Naphade et al. (1997) [35] solved the resource-constrained scheduling problem using PSS, achieving results comparable to the branch and bound technique. Leon & Ramamoorthy (1997) [33]applied PSS to the flexible flow line scheduling problem and demonstrated its effectiveness in this domain. Albrecht et al. (2013) [2] utilized PSS to address the rail network rescheduling problem, generating numerous different train timetables. Storer et al. (1996) [49] used PSS for the number partitioning problem, noting that while it may require longer solution durations, the quality of solutions improves. PSS can be combined with various heuristics, such as the Lagrange relaxation based heuristic used by Jeet & Kutanoglu (2007) [28] to solve the generalized assignment problem. Despite the wide applications of PSS in various problem domains, there hasn't been any prior study suggesting the use of PSS metaheuristic for the fleet sizing problem and employing the fix and optimize (FO) technique as the problem-specific heuristic. This research introduces an innovative approach to address the integrated replenishment planning and fleet sizing problem with predetermined replenishment frequencies, considering vehicle ownership, inventory, and routing costs. We apply the FO heuristic in two versions of PSS to solve the problem presented in Dastjerd & Ertogral (2019) [14]. Our main objective is to build upon and advance the existing solutions proposed in their previous research. By incorporating the PSS metaheuristic with FO, we aim to enhance the effectiveness, efficiency, and overall quality of the solutions, refine existing methodologies, and potentially introduce novel approaches to achieve more robust and improved solutions in the field of fleet sizing and replenishment planning.

2.2.5 Fix and optimize heuristic

Pochet & Wolsey (2006) [38] introduced the multi-level capacitated lot size problem and developed the "exchange heuristic," which may have been the origin of the Fix and Optimize (FO) method. When existing solvers proved inadequate in delivering high-quality or optimal solutions within reasonable computing timeframes, researchers turned to mixed integer programming-based heuristics (Tanksale & Jha (2020) [53]). FO emerged as an effective solution method, producing excellent solutions within reasonable timeframes, and has been applied to various lot-sizing problem variations, such as capacitated lot sizing with setup carryover (Gören & Tunalı (2015) [22], Chen (2015) [13]), cooperative lot-sizing (Drechsel & Kimms (2011) [17]), and stochastic capacitated lot sizing (Helber & Sahling (2010) [23]).

The versatility of the FO method extends to other problem types as well. Gintner et al. (2005) [21] used FO to solve a bus scheduling problem, while Dorneles et al. (2014) [16] applied a combination of FO and variable neighborhood search to address high school timetabling. Federgruen et al. (2007) [18] utilized FO for solving a multi-product capacitated lot size challenge. Helber & Sahling (2010) [23] employed the FO method for the multi-level capacitated lot sizing problem, and Neves-Moreira et al. (2018) [36] used it for assigning time intervals and creating delivery schedules, asserting its superiority over commercial solvers.

As of our knowledge, no previous work has suggested an ADP-based solution for the fleet sizing problem proposed by Dastjerd & Ertogral (2019) [14]. In this study, we present a novel solution strategy based on ADP for their problem. Our approach involves an enhanced version of classical ADP, incorporating the FO method to estimate the objective function value.

2.3 Problem Definition

In this section, we outline our problem and present its mathematical formulation. The problem involves a group of geographically dispersed customers who make deterministic product restocking requests. The main objective is to minimize the total annual expenses, which encompass vehicle ownership, routing costs, fixed replenishment costs, and inventory holding costs.

The solution to the problem involves determining the appropriate fleet size and composition of vehicles, as well as the optimal replenishment size and schedule for each customer. The fleet consists of a set of heterogeneous vehicles with different cost parameters and carrying capacities. Replenishments are conducted based on a predefined set of candidate frequencies, which are determined by the number of weeks between deliveries and the specific delivery day of the week. Customers' inventories can be restocked on a weekly, biweekly, thrice-weekly, or quarto-weekly basis, on any of the five weekdays. This results in 20 weekly base frequencies (5 weekdays $\times$ 4 possible delivery intervals). Additionally, we consider a daily frequency, resulting in a total of 21 distinct possible frequencies, which are listed in Table 2.1 along with their respective labels.

Table 2.1: Frequency labels.

Label	**Frequency**
1	Daily
2-6	Weekly
7-11	Biweekly
12-16	Thrice-weekly
17-21	Quarto-weekly

Some of the frequencies frequently overlap, therefore it is important to record them for avoiding both double counting of deadheading costs and accurately depicting capacity limits on coinciding days. For example, considering frequencies that cover Thursdays, labels of the coinciding frequencies are $5, 10, 15$, and 20, plus the daily frequency.

We assume the following operational challenges as they pertain to our problem:

1. Each customer's inventory must be restocked using a single truck based on a single delivery frequency,
2. Only a certain maximum number of customers may be visited on a given day, and,
3. We took into account the customer's location when allocating consumers to a route. That is if the problem has clustered customer structure, two customers on different clusters cannot be delivered with the same frequency and vehicle.

One of the distinctive features of the mathematical model is that it adopts an approximation method in determining the routing costs. The routing cost is estimated by the product of the average cost of travelling from one consumer to another and the number of consumers visited on a specific route. Since the main focus of the problem is fleet sizing and routing cost is an operational cost that may vary on a daily basis, taking routing cost as an approximate value was appropriate. Besides, this simplification saves significant computational time in terms of solution durations and makes it possible to build the problem without the need for exact routing data.

We provide an overview of the mixed integer programming model for the problem below. The notations used are shown below:

Sets:

I: Customer set

V: Vehicle set

F: Frequencies set, F= {1, 2, ..., 21}

F_j: Coinciding frequencies set, $\forall j = 1, \dots, n$

D: Set of days of the week, D = {1, 2, …, 5}

H: Set of weeks per year, H = {1, 2, …, 52}

Parameters:

N: Number of coinciding frequency sets

m: Number of customers

r_v: Approximate routing cost between two customers for vehicle v

g_v: Dead heading cost for vehicle v

a_v : Annual ownership cost of vehicle v

λ_{if}: Demand of customer i at frequency f

h: Annual inventory holding cost per unit of a product

k_{if}: Fixed cost of replenishing customer i using frequency f

s_{max}: Maximum number of customers that can be visited during the day

c_v: Capacity of vehicle v

M: A big number

p_f: Total number of annual replenishments for frequency f

t_{ik}: Incidence matrix of customers i and k (customers i and k can be in the same route if t_{ik}=2, and cannot be in the same route when t_{ik}=1)

Decision variables:

x_{ivf}:	1 if customer i is replenished by vehicle v and frequency f, 0 otherwise.
V_v:	1 if vehicle v is used, 0 otherwise.
L_{vf}:	1 if any customer is assigned to vehicle v and frequency f, 0 otherwise.
R_{dvh}:	1 if any customer assigned is to vehicle v and frequency f on the day d in week h, 0 otherwise
C_{vf}:	Number of customers assigned to vehicle v and frequency f.

Mathematical model

$$Min.\ \sum_{v\in V} g_v p_1 L_{v1} + \sum_{d\in D}\sum_{h\in H}\sum_{v\in V} g R_{dvh} + \sum_{v\in V}\sum_{f\in F} r_v \,.\, p_f \,.\, C_{vf} + \sum_{v\in V} a_v V_v + \sum_{i\in I}\sum_{v\in V}\sum_{f\in F} h\, X_{ivf}\frac{\lambda_{if}}{2} + \sum_{i\in I}\sum_{v\in V}\sum_{f\in F} k_{if} X_{ivf} \tag{2.1}$$

Subject to

$$\sum_{v\in V}\sum_{f\in F} X_{ivf} = 1 \qquad \forall\ i \in I \tag{2.2}$$

$$\sum_{i\in I} X_{ivf} = C_{vf} \qquad \forall\ v \in V, \forall f \in F \tag{2.3}$$

$$M L_{vf} \geq C_{vf} \qquad \forall v \in V, f \in F \tag{2.4}$$

$$L_{vf} \leq C_{vf} \qquad \forall v \in V, f \in F \tag{2.5}$$

$$M\ V_v \geq \sum_{i\in I}\sum_{f\in F} X_{ivf} \qquad \forall\ v \in V \tag{2.6}$$

$$\sum_{i\in I} \lambda_{if} X_{ivf} \leq c_v \qquad \forall v \in V, \forall f \in F \quad (2.7)$$

$$\sum_{i\in I} \sum_{f\in F_j} \lambda_{if} X_{ivf} \leq c_v \qquad \forall\, v \in V, \forall j = 1, \dots, n \quad (2.8)$$

$$\sum_{f\in F_j} C_{vf} \leq s_{max} \qquad \forall\, v \in V, \forall j = 1, \dots, n \quad (2.9)$$

$$\sum_{f\in F_j} X_{ivf} + \sum_{f\in F_j} X_{kvf} \leq t_{ik} \qquad \forall v \in V, i \in I, k \in I, \forall j = 1, \dots, n \quad (2.10)$$

$$R_{dvh} \geq L_{vf} - L_{v1} \qquad \forall v \in V, d \in D, f \in Fj, h \in H \quad (2.11)$$

$$X_{ivf} \in \{0,1\} \qquad \forall\, i \in I, \forall v \in V, \forall f \in F \quad (2.12)$$

$$L_{vf} \in \{0,1\} \qquad \forall v \in V, \forall f \in F \quad (2.13)$$

$$C_{vf} \in Z_{\geq 0} \qquad \forall v \in V, \forall f \in F \quad (2.14)$$

$$V_v \in \{0,1\} \qquad \forall v \in V \quad (2.15)$$

$$R_{dvh} \in \{0,1\} \qquad d \in D, v \in V, h \in H \quad (2.16)$$

The constraint (2.3) assures that all the demands are satisfied. The number of consumers being served by a specific frequency and vehicle are calculated using (2.2). The value of L_{vf} is calculated using constraints (2.4) and (2.5). Constraint (2.6) decides whether a vehicle of type v is used. Vehicle capacity limitations are imposed to the model by constraints (2.7) and (2.8). The first one assures that the trucks are loaded with the products occupying a space which is less than or equal to the carrying capacity of each vehicle. The latter assures the same on coinciding frequencies. The constraint (2.9) sets s_{max} as the maximum number of clients that may be visited on a route each day. Some customers are not eligible to be on the same route simultaneously due to their geographical location. This restriction is reflected by constraint (2.10). The redundant deadheading calculations are omitted by the constraint (2.11). The remaining,2.12 to 2.16 declare the domain of the decision variables.

For further details about the mathematical formulation one can see Dastjerd & Ertogral (2019) [14].

2.4 Approximate Dynamic Programming Heuristic

In this section, we propose a heuristic solution for the problem based on Approximate Dynamic Programming (ADP). ADP encompasses a wide range of computational and modeling techniques used to address complex decision problems with large dimensions. ADP is particularly useful in overcoming the well-known issue of the dimensionality curse, which hinders the application of Bellman's equation. The use of an approximate value function is fundamental in ADP to facilitate decision-making.

To estimate the objective function's value at each ADP iteration, we employ the Fix and Optimize (FO) heuristic. In a prior work by Dastjerd & Ertogral (2019) [14], we developed the original FO heuristic and applied it to the current problem. In the subsequent sections, we provide the recursive equation for our proposed ADP approach, explain the steps for implementing the FO heuristic, and then outline the steps of our suggested ADP methodology.

2.4.1 Suggested approximate dynamic programming

2.4.1.1 Formulation of the recursive equation

ADP, which stands for Approximate Dynamic Programming, is a powerful method used to handle large-scale decision-making processes involving discrete time multistage optimization. In these problems, there exists a state space S, and at each stage, the system is in a specific state $S_t \in S$, from which a decision x_t can be made. After making a decision x_t, the system receives rewards or incurs costs, denoted as $C_t(S_t;\ x_t)$, and transitions to a new state $S_t + 1$. Consequently, decisions at each state are conditionally dependent on all previous states and decisions. As a result, the decision made not only affects immediate costs but also influences the environment in which future decisions are made, thereby impacting upcoming costs.

Dynamic programming tackles complex decision-making problems by breaking them down into smaller subproblems. The optimal solution for the overall problem is achieved by obtaining optimal solutions for each of the subproblems, as outlined in Bellman & Kalaba (1957) [5].

In our specific scenario, the stages of the dynamic programming formulation correspond to the customers (indexed with i). At each iteration, we determine whether a frequency-vehicle-customer assignment is appropriate for that particular customer or not. The decision variable at each stage is denoted as x_{ivf}, and we must decide whether to set $x_{ivf}$to 1 or to 0 for each customer i. Setting $x_{ivf}$to 1 for a specific (v, f) pair implies that x_{ivf} is set to 0 for all other (v, f) options for customer i since each customer can only be assigned to a single vehicle and frequency pair. When making decisions about the x_{ivf} values, we consider all available vehicle capacity options for each frequency for the current customer i, which constitutes the state in dynamic programming terminology. At each stage, when we are deciding the x_{ivf}value for a customer, the state of the system consists of the remaining capacity for each vehicle-frequency pair in terms of both remaining volume (Cap_i) and the remaining number of daily customers (C_{vf}) that can be assigned to it. This state, also referred to as capacity state, Cap_i, is naturally determined by the decisions already made before the current stage. In our implementation, we utilize a mathematical model to determine the state of the system at each stage.

At each stage, we must calculate the incremental cost of allocating or not allocating the current customer to a vehicle-frequency pair, given that the capacity state is Cap_i. The most crucial factor is the estimated solution value of the objective function when there are i remaining consumers, and Cap_i represents the capacity state for customer i at stage i. Taking into account these dependencies and definitions, we present the forward recursion equation below:

Notations:

Cap_i: Available capacity state at each stage for customer i, $i \in I$.

$K(i, Cap_i)$: Value of optimal objective function of the partial problem for customers i, $i+1, i+2, \dots, |I|$ customers the capacity state Cap_i.

$I(x_{ivf}, C_i)$: Incremental cost of x_{ivf} (equating to 1 or 0), for customer $i \in I$, if we have capacity state Cap_i.

f_{i+1} $(Cap_i, x_{ivf}$): The function that returns the capacity state at customer i+1, if we have capacity state Cap_i before customer i and we take the decision x_{ivf} (Setting it to 1 or 0).

Forward recursive equation of the problem:

$$K(i, Cap_i) = min_{\left\{\substack{x_{ivf}=0 \text{ or } 1, v=1,\dots,V \\ f=1,\dots,F}\right\}} \left\{ I\left(Cap_i, x_{ivf}\right) + K\left(i+1, f\left(Cap_i, x_{ivf}\right)\right) \right\}$$

$$K(N+1, Any\ capacity\ state) = 0.$$

Approximate dynamic programming we suggest is founded on an algorithmic approach that progresses forward in number of customers. To solve this problem using traditional dynamic programming, we would need to identify the exact value function $K\left(i+1, f\left(Cap_i, x_{ivf}\right)\right)$ for each value of Cap_i which becomes so complex when the data size increases. Hence, instead of calculating the exact value for the objective function of the proceeding steps, we try to approximate it through implementation of a problem specific heuristic method. In this paper, we employ a Fix and Optimize heuristic for solving the partial problem approximately in each iteration of the approximate dynamic programming. The results obtained from the Fix and Optimize method (FO) is used as the approximate optimal value in place of $K\left(i+1, f\left(Cap_i, x_{ivf}\right)\right)$ in the recursive formula. The problem specific heuristic FO relies on breaking the main problem down into smaller and easier sub problems. The details of the implemented FO are given below.

2.4.2 Fix and optimize heuristic

Fix and optimize is a two-phase heuristic comprising a first phase that generates a feasible solution of good quality, followed by a second phase focused on improving the solution obtained in the initial phase. The key concept behind this heuristic is to solve the main problem by dividing it into sequential partial integer problems. This approach reduces solution times while still producing high-quality solutions. In this method, the problem is divided into subsets of customers. We will outline the phases of the fix and optimize heuristic and provide detailed steps in the following sections.

During the first phase of the Fix and Optimize (FO) heuristic, the main problem is partitioned into smaller subproblems based on a predefined criterion, which, in this case, is the customer index. At each iteration, only the variables of one subproblem are expressed as integers or binaries, while the variables in the remaining subproblems are defined as linear variables or set to fixed values from previous iterations. This process continues until all variables become integers, and the final solution is saved for further improvement in the next phase. The problem addressed in Phase I at iteration i is illustrated in Figure 2.1.

Integer variables, fixed to the values in iteration $i-1$				Integer variables, optimized in iteration i	Relaxed variables as continuous, optimized in iteration i			
A_1	A_2	….	A_{i-1}	A_i	A_{i+1}	A_{i+2}	….	A_p

Figure 2.1:The problem solved at iteration $\boldsymbol{i}$ in Phase I.

The details of phase I of the FO version for our problem are given below;

Phase I

Step 1. Divide the customers into P sub groups with equal number of customers, or as equal as possible. Divide the decision variables in to P sub groups corresponding to P customer groups. Let A_i be the set of all binary variables in the i^{th} sub group.

Step 2. Set the iteration counter i to 1.

Step 3. If $i > 1$, Fix the binary variables in A_j for $j = 1..i-1$, to the values of variables found in iteration $i-1$.

Step 4. Set the variables in A_i as binary variables.

Step 5. Relax the binary variables in A_j for $j = i + 1, i+2, .., P$ as linear variables.

Step 6. Solve the complete model and set $i = i + 1$. If $i < P$ go to step 3. If $i = P$, STOP.

In the second phase, the main problem is also divided into the same subproblems as in the first phase. The variables in this phase are defined as either integers, or they are set to the fixed values obtained in the previous phase. At iteration i, we solve the entire model with all binary variables fixed to the values from the previous iteration, except for A_i, which is considered as an integer and re-optimized. If any improvement is observed in the objective function, the algorithm starts the process again. This procedure continues until no further improvements are observed. The problem addressed in Phase II at iteration i is depicted in Figure 2.2:

Integer variables, fixed to the values in iteration $i-1$				Integer variables, optimized in iteration i	Integer variables, fixed to the values in iteration $i-1$			
A_1	A_2	….	$A_{i\text{-}1}$	A_i	A_{i+1}	A_{i+2}	….	A_p

Figure 2.2: The problem solved at iteration $\boldsymbol{i}$ in Phase II.

The steps of the second phase of the FO are illustrated below:

Phase II

Step 1. Set the counter i to 1.

Step 2. Clear the values of all variables in A_i and re-define them as binaries.

Step 3. if $i > 1$ then for $j = 1, 2, .., P$ and $j \neq i$ set the variable values in A_j to values found in iteration $i-1$. If $i = 1$ then for $j = 1, 2, .., P$ and $j \neq i$ set the variable values in A_j to values found in Phase I.

Step 4. Solve the complete model, set $i = i + 1$.

Step 5. Check the objective value, if the objective value is better than best solution so far and $i < P$, update the best solution as the current solution and go to step 1. If no improvement occurred in the current iteration and $i < P$ go to step 2. If no improvement occurred in the current iteration and $i > P$ STOP.

2.4.3 Look ahead strategy for ADP

During each ADP step, we need to determine the value of x_{ivf} for a specific customer i, which can either be set to 1 or 0 for each possible vehicle-frequency combination. However, as the data size increases, the number of combinations to be checked for determining the value of x_{ivf} grows rapidly, leading to lengthy solution times for moderate or large-sized problems. To address this issue, we implemented a "look ahead" fixation technique, which helped reduce the number of vehicle-frequency pairs that need to be checked. The concept behind the fixation technique is to predict the x_{ivf} values for later customers in the dynamic programming stages that are likely to be set to 1 in the final solution. As we conduct FO iterations during dynamic programming, we monitor the x_{ivf} values for later customers, and if they frequently assume the value of 1 or a value close to 1, we fix them to 1. By doing so, we save significant computational effort since fixing a x_{ivf} to 1 for a specific customer i and (v, f) pair results in setting the remaining x_{ivf} values to 0 for the remaining (v, f) options for that customer. Our algorithm keeps track of the x_{ivf} variables in the last α% of the FO iterations and if they reach a value of 0.9 or higher in this period, we fix them to 1. This optimization helps expedite the computation process and improve the efficiency of the overall algorithm.

2.4.4 Improvement algorithm for ADP

We propose a two-phase improvement algorithm to further enhance the results obtained from ADP with the look ahead strategy. After analyzing the characteristics of the solutions generated by ADP, we observed that the significant deviations from the optimal or best-bound values are primarily attributed to the excessive utilization of vehicles. Since the vehicle ownership charges represent the most substantial cost component in our context, assigning an extra vehicle in comparison to the optimal solution has a substantial impact on the objective function value.

The improvement algorithm is designed to focus on optimizing the fleet composition and size. In the first phase, we specifically examine solutions where each type of vehicle in the ADP solution is incrementally increased and decreased by one, while keeping the number of other vehicles fixed.

Let k_v represent the number of vehicles in the solution found by ADP or the best solution during the improvement process. The steps of phase I in the improvement can be illustrated as follows;

Improvement phase I:

Step 1. Let i=1, and improvement flag = 0, best_solution = The solution from ADP.

Step 2. Fix the $k_i = k_i+1$ and fix the other vehicle numbers to their value in the best solution so far. Apply ADP. If the solution found is better than the best solution so far, let best_solution = the current solution, improvement flag = 1. Else go to next step

Step 3. Fix the $k_i = k_i-1$ and fix the other vehicle numbers to their value in the best solution so far. Apply ADP. If the solution found is better than the best solution so far, let best_solution = the current solution, improvement flag = 1. Else go to next step.

Step 4. Let $i = i + 1$. If $i < |V|$ then go to Step 2, else go to next step

Step 5. If improvement flag = 1 go to Step 1, else STOP and report the best_solution.

Improvement phase II:

In second phase of the improvement algorithm, we check if there is any improvement when we replace a vehicle with two smaller size vehicles or with a smaller size vehicle. Two neighborhoods are defined as explained below:

1. Removing two small vehicle and replacing them with a large one
2. Replacing one large vehicle in the fleet with a small one

We use the defined neighborhoods for changing the fixed V_v set on the solution obtained after the first improvement phase. The resulting vehicle numbers are used as bounds on the number of vehicles that can be used. If the changes in vehicle sets of the second phase yields improved solutions compared to the solutions from the first phase, the first phase is restarted with the new vehicle composition and the solution is updated. Otherwise, the algorithm is terminated and the current best solution is reported as the final solution.

2.4.5 Pseudo codes of the ADP and improvement algorithm

The following section shows the ADP pseudocode:

Notation:

C: Set of Customers

V: Set of Vehicles

F: Set of given frequencies

P_0: Set of (i, v, f) indices for which x_{ivf} are set to 0 in the previous stages in dynamic programming

P_1: Set of (i, v, f) indices for which x_{ivf} are set to 1 in the previous stages in dynamic programming

P : Set of (i, v, f) indices for which x_{ivf} are set to 1 or 0 in the previous stages using look ahead strategy in dynamic programming

Cap_i: Remaining capacity in terms of volume and assignable customer number in stage i

L_1^{α}: Set of x_{ivf}'s for later customers in dynamic programming that have a value near to 1 (greater than 0.9) in the last α percent of the iterations of FO executed in the dynamic programming.

$Cost_0$: Approximate cost of setting x_{ivf} = 0 in each stage (approximate value for $\{I(Cap_i, 0) + K(i + 1, f(Cap_i, 0))\}$)

$Cost_1$: Approximate cost setting x_{ivf} = 1 in each stage (approximate value for $\{I(Cap_i, 1) + K(i + 1, f(Cap_i, 1))\}$)

ADP pseudocode

```
Input $P_0, P_1, Cap_i$ , $\forall i \in C, v \in V, f \in F$
1.   for each $i = 1..|C|$
2.     if $\notin P$ and $Cap_i = true$ do
3.       Determine $Cost_0$ by calling FO $(i, v, f, 0, P_0, P_1, P_1', Cap_i)$
4.       Determine $Cost_1$ by calling FO $(i, v, f, 1, P_0, P_1, P_1', Cap_i)$
5.       if $Cost_0 \leq Cost_1$ then
6.         $P_0 = P_0 \cup (i, v, f)$
7.       else
8          if $(i, v, f) \in L_1^{\alpha}$
9            $P_1' = P_1' \cup (i, v, f)$
```

```
10            Else
11.             $P_1 = P_1 \cup (i, v, f)$
12.             $P_0 = P_0 \cup \{(i, v^{'}, f^{'}) | v^{'} \in V\backslash\{v\}, f^{'} \in F\backslash\{f\}\}$
13            end if
14.         end if
15.       end if
16.     end for
17.   end for
18. return Cost (null, null, null, $P_0, P_1, P_1', L_1^{\alpha}$)
```

Steps of the improvement phase and the notations are defined below:

<u>Notations:</u>

N: Set of neighborhoods generated in the first phase, $\{n_1, n_2, n_3, n_4\}$.

N': Set of neighborhoods generated in the second phase, $\{n'_1, n'_2\}$.

N_v: Set of bounds generated by applying moves in N to V.

N_v': Set of bounds generated by applying moves in N' to V.

C: Set of customers

V: Set of vehicles

F: Set of frequencies

J: Neighbors

P_1: Set of the (i, v, f) triples that are fixed to 1

$Cost$: Value of objective function from ADP

$Cost'_{n_v}$: Value of objective function for the first phase

$Cost'_{n'_v}$: Value of objective function for the second phase

n_1: Increasing large vehicles number by 1

n_2: Decreasing large vehicles number by 1

n_3: Increasing small vehicles number by 1

n_4: Decreasing small vehicles number by 1

n': Addition of one small vehicle and exclusion of one large vehicle

n'_2: Exclusion of two small vehicles and addition of one large vehicle

Improvement Stage Pseudocode

```
Input $P_1, Cost, N_v, N'_v, \forall i \in C, v \in V, f \in F, j \in J$
1.   **initialize** $j = 0$ and $v \in P_1$ **do**
2.     Calculate $Cost'_{n_v}$ by calling $F\&O\ (i, v, f, N_v)$
3.     **if** $Cost'_{n_v} \leq Cost$ **then**
4.       $Cost \leftarrow Cost'_{n_v}, V \leftarrow N_v$ do
5.       Go to step 1
6.     **else if**
7.       $j \neq |j|\ set\ j \leftarrow j + 1$ and go to step 2
8.     **else**
9.       Set $k = 0$ **do**
10.      Calculate $Cost'_{n'}$by calling $F\&O\ (i, v, f, N'_v)$
11.    **if** $Cost'_{n'_v} \leq Cost$ **then**
12.      $k \neq |k|\ set\ k \leftarrow k + 1$ and go to step 10
13.    **else if**
14.      $k = |k|\ and\ Cost'_{n'_v} \leq Cost$ **then**
15.      Go to step 1.
16.    **Else**
17.     **end if**
18.    **end if**
19   **return** $Cost\ (i, v, f)$
```

2.5 Problem Space Search Metaheuristics

In this section, we will provide a brief explanation of the proposed metaheuristic and then outline the algorithm steps. The Problem Space Search (PSS) was first introduced by Storer et al. (1992) [50] as a novel metaheuristic algorithm. According to Naphade et al. (1997) [35], PSS is a solution technique that uses another heuristic to perform a local search. The heuristic employed in PSS must be fast and capable of producing acceptable solutions. The PSS metaheuristic requires an initial feasible solution, a problem-specific fast heuristic, and a method for generating neighborhoods. Unlike traditional search methods, PSS generates neighborhoods by perturbing the problem data or adjusting heuristic parameters. The perturbed data is then used as input for the problem-based heuristic, while the original data is used to evaluate the quality of solutions within each neighborhood. A pair (h, p) represents a neighborhood in which the search is conducted. Here, h represents the heuristic method, and p represents the problem data. Applying the heuristic to the problem yields the solution, which can be represented as $s = h(p)$. The essence of PSS lies in the repeated application of the heuristic with altered data, leading to changes in decision sequences within the heuristic. The quality of solutions produced by PSS depends on factors such as the suitability and efficiency of the base heuristic. It is crucial to choose a problem-specific heuristic that can produce quality solutions within reasonable time frames. Additionally, the magnitude of perturbations must be appropriate—not too large or too small—to ensure variation in instances without compromising solution quality.

Here, we introduce two variations of the PSS metaheuristic, both of which incorporate the Fix and Optimize (FO) technique proposed in our previous paper, Dastjerd & Ertogral (2019) [14]. First, we present PSSI, which slightly modifies the cost data of the problem. Second, we introduce PSSII, which involves changes in heuristic parameters for neighborhood generation. The steps for PSS are described in following sections.

2.5.1 PSSI description and steps

The problem involves various cost parameters, including holding and replenishment costs, deadheading and per kilometer costs, and vehicle ownership costs. To create different instances for the problem, we use a perturbation parameter, β, which helps generate upper and lower limits for each cost component, C. The cost ranges are then utilized to generate new random values for each cost parameter, ensuring they fall within the high and low levels specified by the cost range. Each newly generated random instance is then fed into the fix and optimize algorithm.

During the process, the objective function value for each unique random instance is calculated using the original cost data. After each call to the heuristic, the best current solution value is updated, and the values of the variables corresponding to the best solution are saved. The solution obtained in the initial phase is then used in the second phase for further improvement. To guide through the PSSI application, we first provide a list of the notations used to define the steps, followed by instructions on applying the PSSI method.

Notations:

m: The number of data instances generated randomly.

D_i: i^{th} random instance.

R_c: Range vector for cost component C.

C: Vector of original cost component.

C_i^r: Vector of random cost component for instance i.

S_i^{FO}: Solution from FO for instance i.

S_c^*: Latest best solution

Obj_c^*: Latest best objective value.

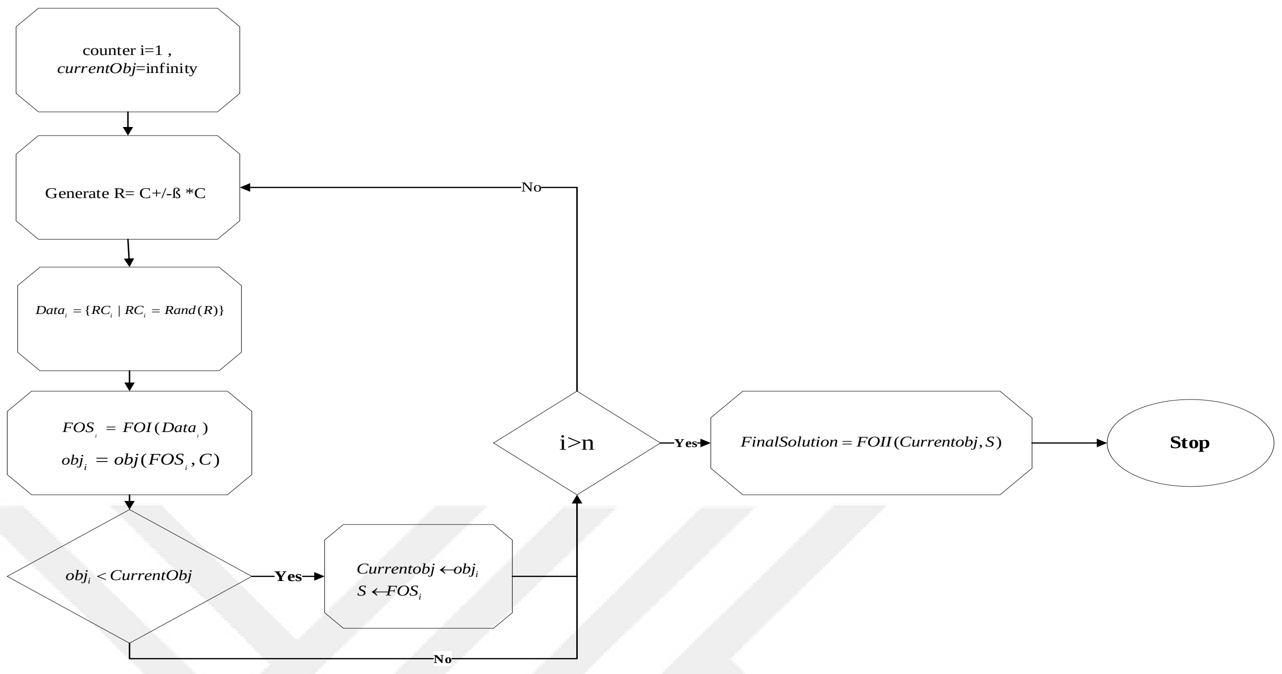
Obj_i: Objective function value for i^{th} random instance.

β: Perturbation parameter

Steps for PSSI algorithm:

Step 1. Equate the instance counter i to 1, and the Obj_c^* to infinity.
Step 2. Calculate the range vector R for C using the equations (4.1) and (4.2).
Step 3. Generate D_i of instance I using the randomly generated C_i^r values.
Step 4. Call the first phase of FO and solve the problem for D_i values.
Step 5. Find the value of Obj_i by incorporating S_i^{FO} and C to the objective function formula.
Step 6. Compare the new objective value with current objective value:
    6.1. If $Obj_i < Obj_c^*$, $Obj_c^* \leftarrow Obj_i$ and $S_c^* \leftarrow S_i^{FO}$.
    6.2. If $Obj_i \geq Obj_c^*$, go to Step 3.
Step 7. Increase instance counter by 1 and go through steps 3 to 6 until $i > m$.
Step 8. Select the smallest value of objective function among all the m instances.
Step 9. Feed the best current solution S_c^* to the second phase of FO, improve and report the solution value.

For better clarity, we provide the flowchart in Figure 2.3 to illustrate the PSSI steps. follows:



counter i=1 , *currentObj*=infinity
Generate R= C+/-ß *C
$Data_i = \{RC_i \mid RC_i = Rand(R)\}$
$FOS_i = FOI(Data_i)$
$obj_i = obj(FOS_i, C)$
$obj_i < CurrentObj$
Yes
$Currentobj \leftarrow obj_i$
$S \leftarrow FOS_i$
No
i>n
No
Yes
$FinalSolution = FOII(Currentobj, S)$
Stop

Figure 2.3: Flowchart for PSSI.

2.5.2 PSSII description and steps

The quality and duration of the solution can be influenced by the order in which the subproblems are solved in FO. Thus, the subproblem solving order becomes crucial. In this study, we leverage this characteristic to generate random yet acceptable solutions. In PSSII, we introduce randomization in generating the subproblem orders. Specifically, for each FO run, the customer index set is randomly regenerated. As a result, in this version of PSS, the perturbation for neighborhood generation involves changing the parameters of the heuristic rather than the dataset. The only notation that differs from PSSI is denoted as I_i', which represents the new customer index set for dataset i. The algorithm steps are outlined below:

Steps for PSSII algorithm:

Step 1. Equate the instance counter i to 1, and the Obj_c^* to infinity.

Step 2. Produce a random consumer index set I_i'.

Step 3. Generate the dataset i using the I_i' to build customer subgroups.

Step 4. Feed the new dataset D_i to first phase of FO algorithm and solve the problem.

Step 5. Compare the objective function value:

5.1. If $Obj_i < Obj_c^*$, $Obj_c^* \leftarrow Obj_i$ and $S_c^* \leftarrow S_i^{FO}$.

5.2. If $Obj_i \geq Obj_c^*$, go to Step 2.

Step 6. Increase i by 1 and go through steps 2 to 5 until $i > m$.

Step 7. Select the smallest value of objective function among all the m instances.

Step 8. Feed the best current solution S_c^* to the second phase of FO, improve and report the solution value.

To enhance clarity, we provide a flowchart in Figure 2.4 which illustrates the steps of PSSII:

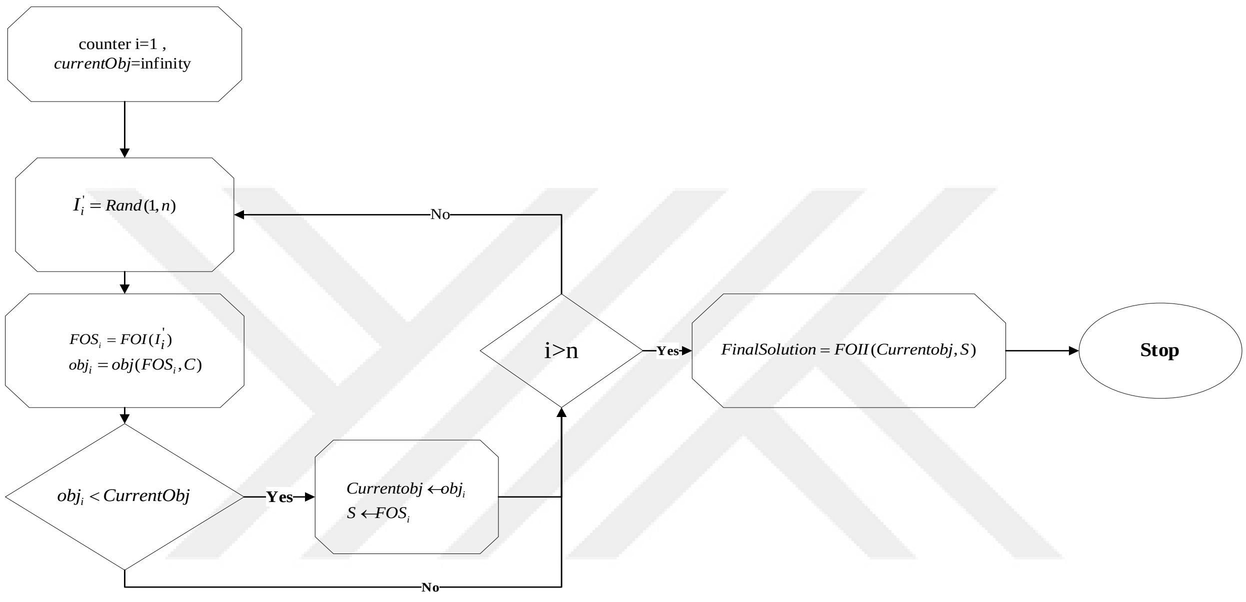


Figure 2.4: Flowchart for PSSII.

2.6 Computational Results

We conducted a numerical analysis of the proposed solution approach using the dataset generated as described in 2.4.12.6.1. The details of the analysis setup and the obtained results are presented in the following subsections.

2.6.1 Dataset

The proposed heuristic is assessed through four distinct scenarios, each characterized by varying demand levels and customer clusters. These scenarios correspond to the ones used in Dastjerd & Ertogral (2019) [14]. Scenario 1 and 2 represent cases with normal demand, with scenario 2 further grouping customers based on their geographical location. Scenarios 3 and 4 present high demand versions, with the demand increased by 50%, of scenario 1 and 2, respectively.

Regarding problem variations, we defined 24 different settings, taking into account vehicle capacities (cap), costs per kilometer for vehicles (R), vehicle ownership costs (A), inventory holding costs (h), and fixed setup cost (K). In Table **2.2**, 1.2A denotes cases with a 20% increase in ownership costs, while 1.4A indicates settings with a 40% increase in ownership costs for larger vehicles. Similar patterns apply to 1.2R and 1.4R in terms of cost per kilometer for vehicles. Across all parameter settings, 40 customers are served with 8 owned vehicles, and their yearly demand is generated based on a uniform distribution ranging from 80 to 120 units. The parameter settings are presented in Table 2.2.

Table 2.2: Problem characteristics.

Setup cost	Holding	Ownership	Routing	Capacity	Problem
k=50	h=300	1.2A	1.2R	7,10	1
				14,20	2
				21,27	3
			1.4R	7,10	4
				14,20	5
				21,27	6
		1.4A	1.2R	7,10	7
				14,20	8
				21,27	9
			1.4R	7,10	10
				14,20	11
				21,27	12
	h=600	1.2A	1.2R	7,10	13
				14,20	14
				21,27	15
			1.4R	7,10	16
				14,20	17
				21,27	18
		1.4A	1.2R	7,10	19
				14,20	20
				21,27	21
			1.4R	7,10	22
				14,20	23
				21,27	24

2.6.2 ADP results with improvement step

The proposed heuristic was implemented to evaluate its performance on the generated instances. The results obtained from the heuristic were then compared to those obtained from CPLEX. All of the problems were solved within the 3-hour time limit in CPLEX, and for some instances, the results are reported as best lower bounds. The gaps from the lower bounds are denoted with an asterisk (*). We conducted an analysis of the results, categorizing them based on the gaps from CPLEX results, fleet compositions, and solution durations. The tabulated results are presented below.

From the gaps presented in Table 2.3, we can conclude that the solutions obtained from the ADP heuristic demonstrate satisfactory performance. The gaps, ranging from 4.5 to 9.24 average deviations across the four scenarios, are mostly close to the lower bounds obtained from CPLEX within the 3-hour run time. As a result, the deviation from the actual optimal solution is expected to be even less than these values.

Observing the data in Table 2.3 , it becomes evident that clustering has a negative impact on the performance of the ADP heuristic when comparing scenario 2 to scenario 1 and scenarios 4 to scenario 2. This outcome is expected because clustering makes the problem more challenging for the fix and optimize process. Any incorrect early assignment of a vehicle-frequency pair to a customer becomes harder to rectify through later assignments due to the significantly reduced feasible solution space in the clustered cases compared to scenarios with no clusters.

The challenging nature of the clustered problems is evident from the fact that the number of problems solved to optimality within the 3-hour run time is much higher in scenarios 1 and 3 when compared to scenarios 2 and 4.

The performance of ADP is negatively affected by the demand. In high-demand scenarios, the solution typically requires a larger number of vehicles, leading to an increase in the number of alternatives that need to be considered. As a result, higher demand makes the problem more complex, thus slightly reducing the performance of the heuristic.

In Table 2.4, when considering the scenario-based average vehicle utilization, the total average number of vehicles assigned for distribution operations increases with the addition of clusters and by increasing the demand values by 50%. For instance, moving from scenario 2 to 3, the total average changes from 2.25 to 2.67, and in terms of fleet composition, the number of small trucks changes from 1.58 to 2. This increase is attributed to the higher demand in scenario 3, while the available truck capacity remains unchanged. Similarly, transitioning from scenario 2 to 4 results in changes in the average total number of utilized vehicles and fleet composition. The addition of clusters in scenarios with higher demands leads to alterations in the total average number of vehicles, whereas in scenarios with base demand, it remains the same.

Table 2.5, tabulates solution times for different scenarios. As it is anticipated, solving scenarios with clusters yield longer solution times resulting from the limitation put on the route generation. That is, there is a constraint on the customers to be served on the same route. Geographically distant ones cannot be visited on the same route in a day. This constraint leads to an increase in the number of choices to be assessed for assigning customers to vehicles and frequencies.

Figure 2.5 to Figure 2.8 depict the percentages of cost components relative to the total cost in each specific scenario. The considered cost components include setup cost, inventory holding cost, vehicle ownership cost, and approximate routing cost. Across all scenarios and problem settings, ownership cost consistently emerges as the highest cost item, accounting for 30% to 60% of the total cost. Following closely is the setup cost, which fluctuates between 20% and 50% of the total annual cost. Inventory holding cost constitutes approximately 5% to 30% of the total cost. As mentioned earlier, the routing cost has the lowest percentage, making up to 20% of the total cost. These cost percentages align with the expected distribution for logistics costs in practice, where inventory-related costs constitute about one third of the total logistics costs, while the remaining two thirds are attributed to transportation expenses.

The following set of charts, Figure 2.9 to Figure 2.12, illustrate the frequency distribution for each frequency set in each problem scenario. These charts provide insights into how frequently each set of frequencies (daily, weekly, bi-weekly, thrice-weekly, or quarto-weekly) is utilized in each parameter setting within each scenario.

In scenarios 1 and 2, bi-weekly replenishment plans are predominantly favored, while in scenarios 3 and 4, customers are mostly replenished with weekly delivery programs. The reason for this disparity between scenarios lies in the demand levels. In the first two scenarios, the demand is relatively low, allowing for a larger consolidation of products in a single truck, considering the carrying capacities. On the other hand, in scenarios 3 and 4, the demand values are increased by 50%, while the vehicle capacities remain the same. Consequently, there is less opportunity for consolidation in a single truck, resulting in a greater need for more frequent replenishments to meet customer demands.

Table 2.3: Percentage deviations from best bounds /optimal for improved ADP.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
1	1.67%*	3.76%*	13.38%	14.76%*
2	4.15%	7.46%*	1.61%*	6.94%*
3	1.23%*	10.26%*	1.23%	12.93%*
4	9.90%	4.77%*	12.44%	10.95%*
5	3.54%	2.18%*	1.19%	5.60%*
6	1.75%*	10.06%*	1.23%	7.67%*
7	12.49%	14.72%*	13.44%	11.50%*
8	1.62%*	8.38%*	1.01%	10.04%*
9	1.67%	10.19%*	1.32%*	6.83%*
10	12.34%	6.75%*	11.77%	7.52%*
11	1.32%	7.75%*	1.01%	10.78%*
12	1.43%*	10.00%*	1.34%*	7.10%*
13	9.47%	4.16%*	9.99%	11.08%*
14	1.30%	9.97%*	8.54%	7.11%*
15	0.00%	0.00%	1.02%	5.70%*
16	13.01%	5.01%*	16.00%	10.08%*
17	1.12%	9.62%*	0.60%	7.12%*
18	0.00%	0.00%	0.77%	21.33%*
19	12.47%	6.30%*	11.94%	8.08%*
20	1.30%	7.68%*	8.54%	11.75%*
21	0.13%	0.00%	0.82%	3.78%*
22	12.20%	7.41%*	11.28%	7.21%*
23	1.02%	7.29%*	8.54%	12.59%*
24	0.13%	0.00%	0.85%	3.20%*
Ave	4.50%	6.41%	5.75%	9.24%

Table 2.4: Scenario based Fleet utilization average for improved ADP.

	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	L	S	L	S	L	S	L	S
Ave	0.67	1.58	0.54	1.71	0.67	2.00	0.92	2.13

Table 2.5: CPU times (in seconds) for improved ADP.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
1	233.22	2686.43	2626.93	4567.63
2	384.86	2244.66	241.18	1947.72
3	409.29	4399.31	361.70	4685.58
4	425.80	3627.30	978.29	5127.14
5	320.33	2052.15	399.65	2710.26
6	579.04	3816.51	266.09	4487.38
7	684.45	2343.58	949.97	1886.79
8	655.26	5996.36	658.90	5802.12
9	267.41	4328.48	235.93	5010.19
10	431.78	3660.73	1230.30	3939.08
11	496.26	5868.22	727.76	6409.28
12	274.68	4751.66	502.45	4810.56
13	953.07	2006.66	1086.33	2775.83
14	245.61	2937.31	429.50	3433.01
15	208.42	1758.74	259.50	2512.47
16	437.50	1750.55	1100.78	3065.19
17	253.07	2959.58	437.88	4230.39
18	208.27	1613.82	345.55	2728.12
19	721.72	2138.13	873.62	3701.81
20	303.34	3813.74	449.71	3653.63
21	248.30	1335.90	312.33	2823.36
22	497.28	1619.63	1142.93	2809.44
23	548.66	3410.48	494.57	3602.08
24	192.53	1600.90	232.82	2804.65
Ave	415.84	3030.03	681.03	3730.15

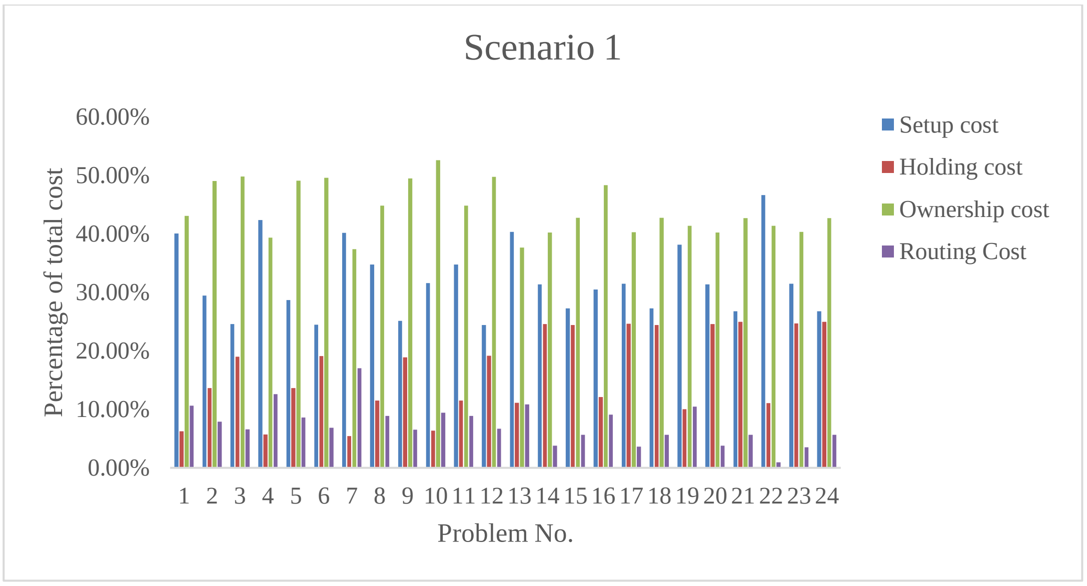


Figure 2.5: Cost element percentages for scenario 1.

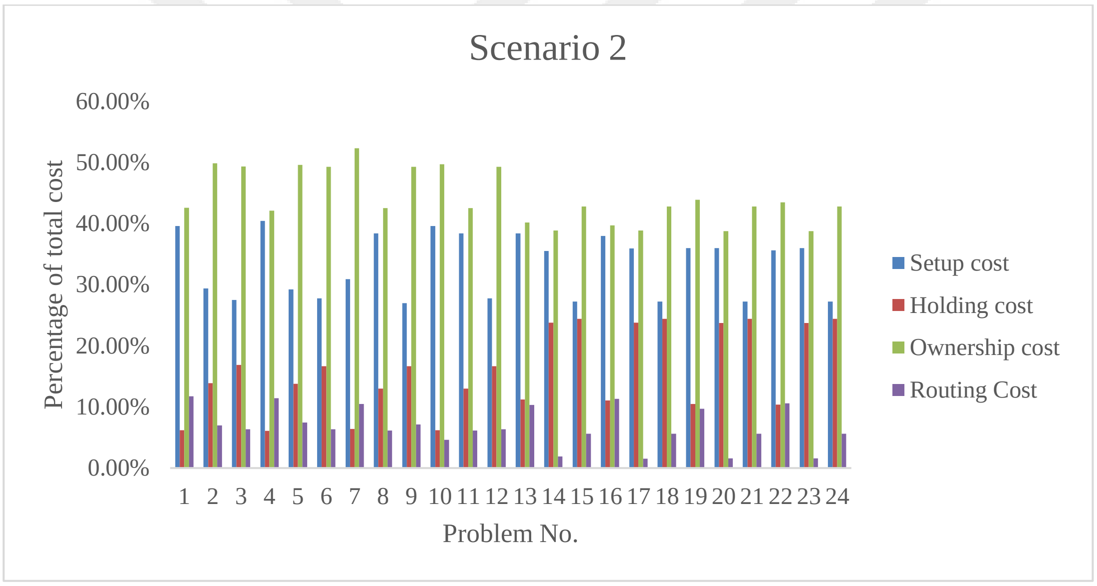


Figure 2.6: Cost element percentages for scenario 2.

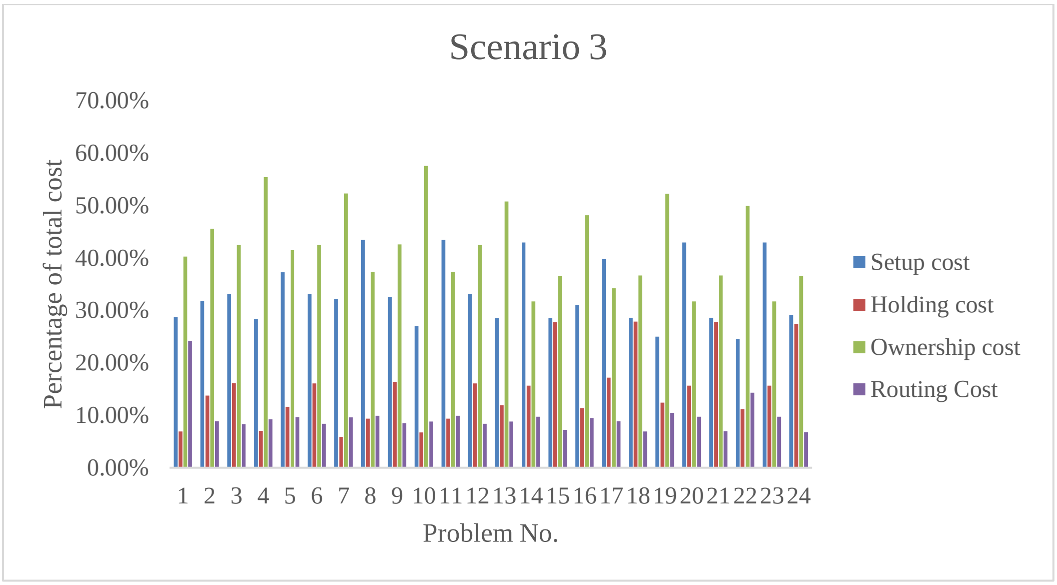


Figure 2.7: Cost element percentages for scenario 3.

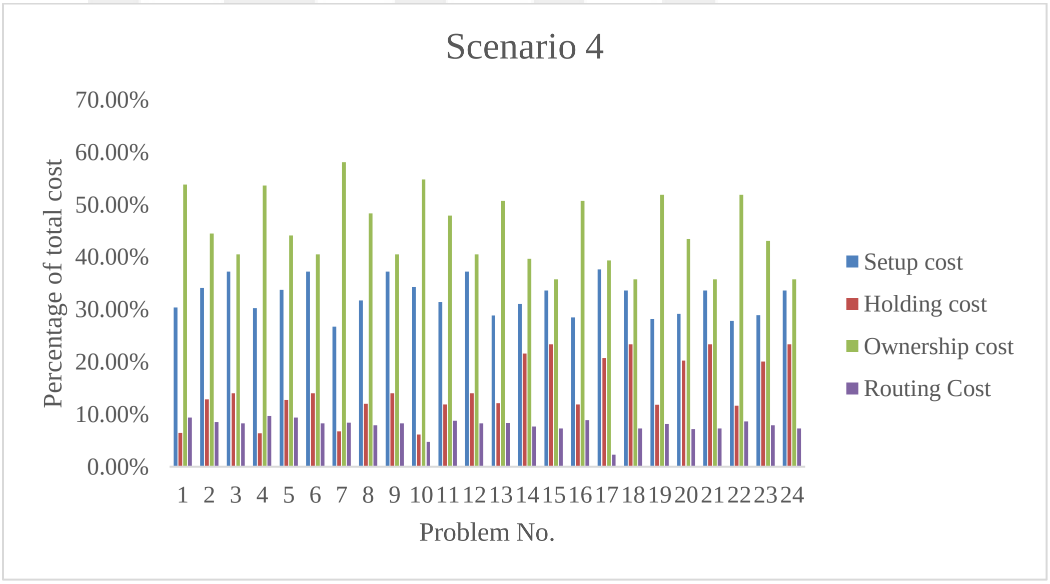


Figure 2.8: Cost element percentages for scenario 4.

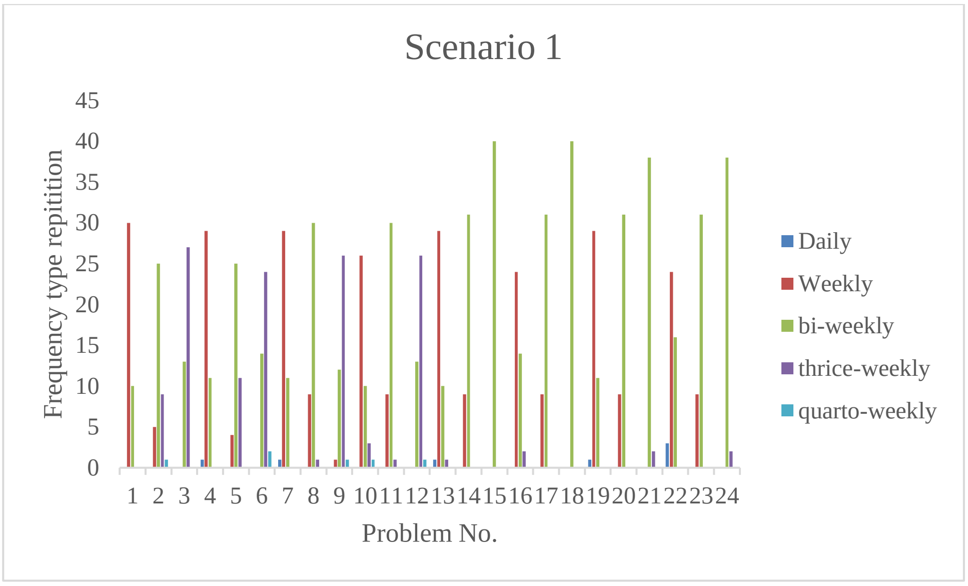


Figure 2.9: Frequency repetition for scenario 1.

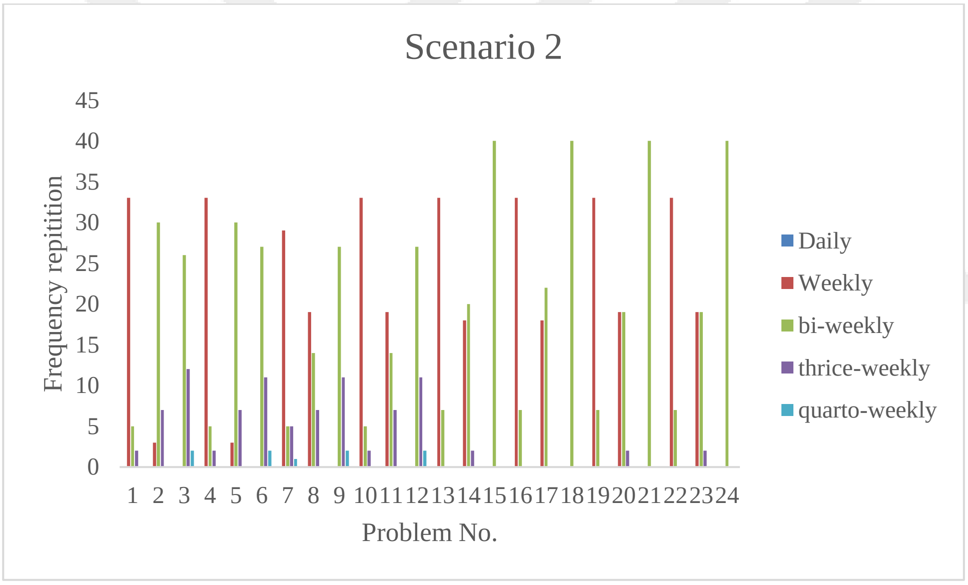


Figure 2.10: Frequency repetition for scenario 2.

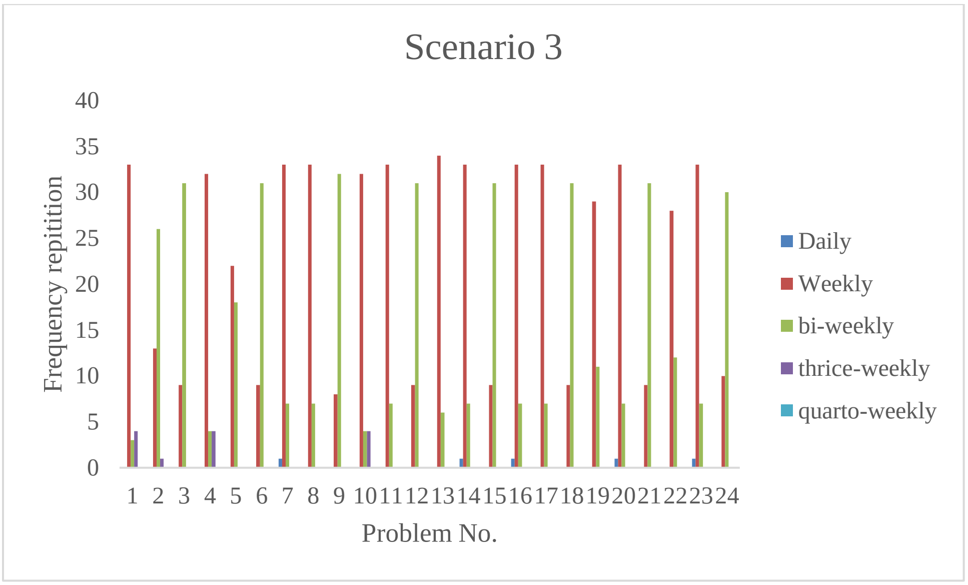


Figure 2.11: Frequency repetition for scenario 3.

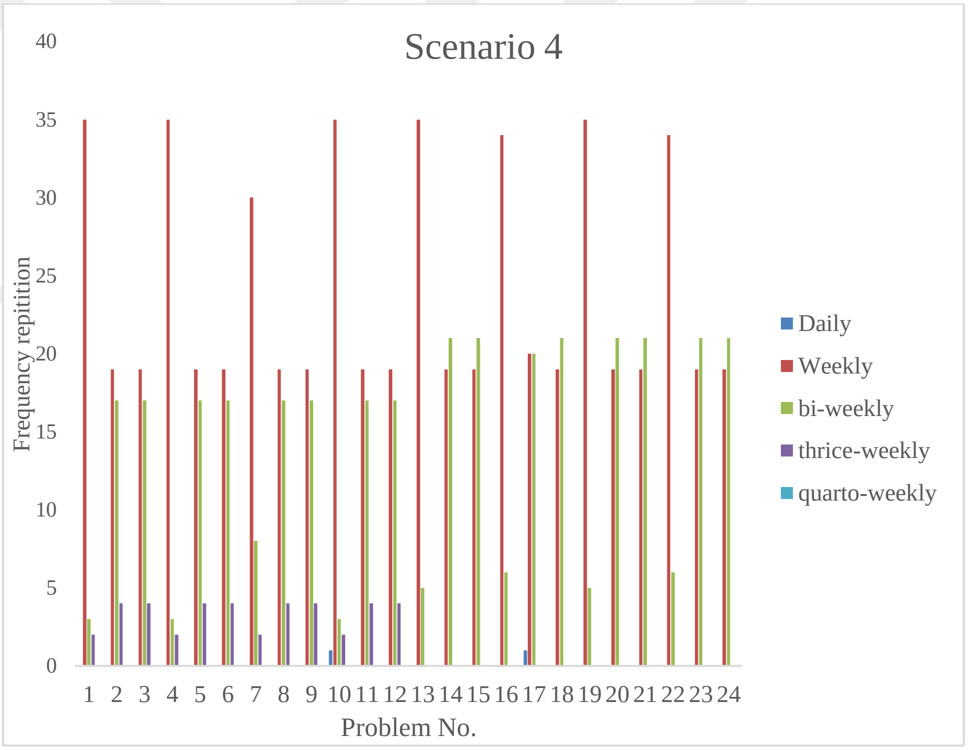


Figure 2.12: Frequency repetition for scenario 4.

2.6.3 PSS Results

The perturbation factor in the PSSI algorithm is considered in two different values, denoted as β_1 and β_2, with values set to 10% and 20%, respectively. For each β value, 50 instances were generated and used in the PSSI metaheuristics.

In this section, we analyze and discuss the results obtained from PSSI, PSSII, and the basic FO approach. We aim to observe the improvements brought by PSSI and PSSII compared to the simple FO method. The results for the two versions of PSS and FO are presented in several tables, labeled from Table 2.6 to Table 2.13, covering the gaps from the optimal or best bounds obtained from CPLEX, fleet composition details, and average solution durations. In these tables, L-S stands for large and small vehicles.

For the first set of tables that show the gaps, the values for the optimal solutions or best bounds are obtained using the MIP solver CPLEX. Since there is a time limit on the solutions from CPLEX, all results are obtained within three hours. The values marked with a star (*) indicate the divergence from the best lower bounds.

This section provides a comparison of the results from the PSS heuristics and FO, and the comments are categorized into three subsections based on the data presented in the tables, as described above.

2.6.3.1 Percentage gaps from optimal/best bounds

As explained in previous section, we developed two versions of PSS metaheuristic for the integrated fleet sizing and replenishment planning problem. The first version, PSSI uses cost data perturbation for generating neighborhoods. Results from PSSI with perturbation factors of β_1 =10% and $\beta_2 =$ 20% are presented in Table 2.6 and Table 2.9. Comparing the gaps from PSS versions to FO, a significant decrease is observable. In general, addition of clusters leads to higher annual costs. The fact behind this objective function value growth is that, geographical clustering brings limitations to the transportation operations in terms of vehicle-customer assignment. Fixed vehicle capacities and the customers which cannot be served on the same routes, necessitate a higher number of vehicles for satisfying the same demand amount comparing to the case in which there is no rule for route generation. In our setting the highest cost is the ownership cost which is in accordance with the number of assigned vehicles. Hence, only one extra vehicle added to the fleet can cause a noticeable increase in total cost. In scenario 2 and scenario 4 we cluster customers into groups, and hence the highest gaps belong to these two scenarios. Highest gap for FO in Scenario2 is for problem number 16 with value of 16.07% from the best bound. As it is seen in Table 2.6 PSS reduced the gap value to 5.01%. The same pattern is tracible in Scenario4 with the highest FO gap belonging to 17. problem with value of 31.70% which is reduced to 7.12% by applying PSS. According to Table 2.9, PSS variations enhance the solution by generating a variety of neighborhoods through cost data perturbation and sub problem order randomization. Table 2.9 illustrates that the highest gaps in three of the four scenarios belong to FO which proves the effectiveness of PSS metaheuristic.

Considering PSS variations, it seems that PSSI with perturbation factor of 20% outperforms the other two versions. According to Table 2.9, minimum gaps three of four scenarios belong to PSS (β_2).

Table 2.6: Deviations from objective function value for FO and PSSI (β_1)

Problem No	Scenario1		Scenario2		Scenario3		Scenario4	
	FO	PSSI	FO	PSSI	FO	PSSI	FO	PSSI
1	14.70%*	9.95%*	3.76%*	3.76%*	17.61%	10.93%	15.62%*	15.62%*
2	4.57%	3.53%	7.46%*	7.46%*	1.34%*	1.64%*	21.17%*	6.94%*
3	1.65%*	1.41%*	10.29%*	10.26%*	2.06%	1.44%	12.93%*	12.28%*
4	9.90%	1.55%	16.58%*	4.77%*	17.05%	11.55%	12.00%*	12.00%*
5	4.15%	3.07%	2.18%*	2.18%*	1.43%	1.29%	5.60%*	5.60%*
6	1.90%*	1.30%*	10.06%*	10.04%*	1.47%	0.87%	7.67%*	7.66%*
7	13.66%	4.00%	14.72%*	6.56%*	13.91%	13.96%	11.50%*	11.50%*
8	2.47%*	1.62%*	8.38%*	6.27%*	11.71%	1.01%	20.78%*	10.04%*
9	0.98%	0.98%	10.19%*	10.17%*	2.20%*	2.04%*	6.83%*	6.83%*
10	12.98%	13.15%	14.49%	14.49%*	13.51%	7.26%	11.19%*	11.19%*
11	2.34%	1.60%	7.75%*	6.17%*	9.88%	1.01%	27.42%*	10.78%*
12	1.35%*	1.28%*	10.00%*	9.97%*	1.98%*	1.32%*	7.10%*	7.10%*
13	12.93%	9.46%	15.70%*	4.16%*	15.69%	9.99%	12.67%*	12.63%*
14	1.18%	0.22%	10.31%*	6.62%*	13.82%	0.31%	20.35%*	7.11%*
15	0.06%	0.23%	0.00%	0.00%	0.94%	0.76%	5.70%*	5.69%*
16	13.88%	9.94%	16.07%*	5.01%*	15.79%	11.96%	12.25%*	12.25%*
17	1.79%	1.02%	9.96%*	6.76%*	0.60%	0.60%	31.70%*	7.12%*
18	0.06%	0.46%	0.00%	0.46%	1.25%	0.76%	21.33%*	21.32%*
19	12.60%	13.05%	14.01%*	15.10%*	13.48%	7.66%	12.15%*	12.11%*

20	1.46%	0.81%	7.68%*	7.68%*	0.31%	0.60%	29.99%*	21.24%*
21	0.29%	0.23%	0.00%	0.46%	1.76%	0.76%	3.78%**	3.77%*
22	11.95%	12.24%	14.68%*	15.87%*	7.29%	7.29%	11.80%*	11.80%*
23	1.50%	0.81%	7.29%*	7.29%*	13.33%	0.32%	21.74%*	21.74%*
24	0.29%	0.23%	0.00%	0.46%	1.41%	0.76%	3.20%*	3.20%*
Ave	5.36%	3.84%	8.82%	6.75%	7.49%	4.00%	14.44%	10.73%

Table 2.7: Deviations from objective function value for FO and PSSI (β_2).

Proble m No	Scenario1		Scenario2		Scenario3		Scenario4	
	FO	PSSI	FO	PSSI	FO	PSSI	FO	PSSI
1	14.70%*	9.95%	3.76%*	3.76%*	17.61	11.22	15.62%	15.62%
2	4.57%	0.51%	7.46%*	7.46%*	1.34%	1.64%	21.17%	6.94%*
3	1.65%*	1.22%	10.29%	10.26%	2.06%	1.36%	12.93%	12.28%
4	9.90%	9.90%	16.58%	4.77%*	17.05	11.64	12.00%	12.00%
5	4.15%	1.35%	2.18%*	2.18%*	1.43%	1.12%	5.60%*	5.60%*
6	1.90%*	1.40%	10.06%	10.04%	1.47%	1.18%	7.67%*	7.66%*
7	13.66%	12.49	14.72%	6.56%*	13.91	7.53%	11.50%	11.50%
8	2.47%*	1.24%	8.38%*	6.27%*	11.71	1.01%	20.78%	10.04%
9	0.98%	1.02%	10.19%	10.17%	2.20%	1.90%	6.83%*	6.83%*
10	12.98%	4.13%	14.49%	14.49%	13.51	7.26%	11.19%	11.19%
11	2.34%	1.32%	7.75%*	6.17%*	9.88%	1.01%	27.42%	10.78%
12	1.35%*	1.11%	10.00%	9.97%*	1.98%	1.28%	7.10%*	7.10%*
13	12.93%	9.46%	15.70%	4.16%*	15.69	9.99%	12.67%	12.63%
14	1.18%	0.26%	10.31%	6.62%*	13.82	0.60%	20.35%	7.11%*
15	0.06%	0.46%	0.00%	0.00%	0.94%	0.76%	5.70%*	5.69%*
16	13.88%	9.94%	16.07%	5.01%*	15.79	11.33	12.25%	12.25%
17	1.79%	1.07%	9.96%*	6.76%*	0.60%	0.60%	31.70%	7.12%*
18	0.06%	0.46%	0.00%	0.46%	1.25%	0.70%	21.33%	21.09%
19	12.60%	4.54%	14.01%	15.10%	13.48	7.66%	12.15%	12.11%
20	1.46%	1.02%	7.68%*	7.68%*	0.31%	0.29%	29.99%	11.75%
21	0.29%	0.46%	0.00%	0.00%	1.76%	0.76%	3.78%*	3.77%*

22	11.95%	12.46	14.68%	15.87%	7.29%	7.29%	11.80%	7.60%*
23	1.50%	0.87%	7.29%*	7.29%*	13.33	0.60%	21.74%	12.59%
24	0.29%	0.23%	0.00%	0.46%	1.41%	0.76%	3.20%*	2.99%*
Ave	5.36%	3.62%	8.82%	6.73%	7.49%	3.73%	14.44%	9.76%

Table 2.8: Deviations from objective function value for FO and PSSII.

Problem No	Scenario1		Scenario2		Scenario3		Scenario4	
	FO	PSSII	FO	PSSII	FO	PSSII	FO	PSSII
1	14.70%*	1.69%*	3.76%*	2.55%*	17.61%	13.38%	15.62%*	14.23%*
2	4.57%	4.15%	7.46%*	7.39%*	1.34%*	1.61%*	21.17%*	6.09%*
3	1.65%*	1.34%*	10.29%*	10.26%*	2.06%	0.87%	12.93%*	7.77%*
4	9.90%	0.97%	16.58%*	4.04%*	17.05%	12.44%	12.00%*	10.15%*
5	4.15%	3.54%	2.18%*	1.57%*	1.43%	1.19%	5.60%*	3.91%*
6	1.90%*	1.75%*	10.06%*	32.71%*	1.47%	0.95%	7.67%*	2.75%*
7	13.66%	13.71%	14.72%*	10.90%*	13.91%	13.44%	11.50%*	7.52%*
8	2.47%*	1.62%*	8.38%*	7.00%*	11.71%	1.01%	20.78%*	4.86%*
9	0.98%	1.67%	10.19%*	32.74%*	2.20%*	1.39%*	6.83%*	1.58%*
10	12.98%	13.90%	14.49%	11.04%*	13.51%	11.77%	11.19%*	7.05%*
11	2.34%	1.32%	7.75%*	6.46%*	9.88%	1.01%	27.42%*	5.32%*
12	1.35%*	1.43%*	10.00%*	32.55%*	1.98%*	1.53%*	7.10%*	2.20%*
13	12.93%	13.86%	15.70%*	14.73%*	15.69%	9.99%	12.67%*	10.84%*
14	1.18%	1.30%	10.31%*	9.40%*	13.82%	8.54%	20.35%*	4.78%*
15	0.06%	0.00%	0.00%	19.20%	0.94%	0.78%	5.70%*	3.37%*
16	13.88%	13.67%	16.07%*	13.48%*	15.79%	16.00%	12.25%*	9.86%*
17	1.79%	1.12%	9.96%*	8.66%*	0.60%	0.60%	31.70%*	4.83%*
18	0.06%	0.00%	0.00%	19.01%	1.25%	0.74%	21.33%*	18.37%*
19	12.60%	12.65%	14.01%*	12.31%*	13.48%	11.94%	12.15%*	11.16%*
20	1.46%	1.30%	7.68%*	6.87%*	0.31%	8.54%	29.99%*	6.34%*
21	0.29%	0.13%	0.00%	19.11%	1.76%	0.76%	3.78%*	1.28%*
22	11.95%	12.07%	14.68%*	12.73%*	7.29%	11.28%	11.80%*	11.00%*
23	1.50%	1.02%	7.29%*	6.57%*	13.33%	8.54%	21.74%*	6.79%*

24	0.29%	0.13%	0.00%	18.78%	1.41%	0.76%	3.20%*	0.76%*
Ave	5.36%	4.35%	8.82%	13.34%	7.49%	5.79%	14.44%	6.78%

Table 2.9: Average, Min and Max gaps from optimal/best bound for FO, PSSI and PSSII.

	Average Gap			
	Scenario1	Scenario2	Scenario3	Scenario4
Methods				
FO	5.36%	8.82%	7.49%	14.44%
PSSI (β_1)	3.84%	6.75%	4.00%	10.73%
PSSI (β_2)	3.62%	6.73%	3.73%	9.76%
PSSII	4.35%	13.34%	5.79%	6.78%
Min	3.62%	6.73%	3.73%	6.78%
Max	5.36%	13.34%	7.49%	14.44%

2.6.3.2 Fleet composition

Table 2.10 and Table 2.11 present the fleet composition and average number of assigned vehicles for FO, PSSI, and PSSII, respectively. Table 2.12 shows the total average vehicle usage for all four solution techniques. In these tables, the columns "L" and "S" represent large and small vehicle types, respectively.

When analyzing the average vehicle numbers across different scenarios, it is evident that the addition of clusters leads to a similar pattern as observed in the objective function values. In Scenario 3, where routes can be generated freely without any limitations on customer addition except for s_{max}, the average number of large and small vehicles is significantly lower compared to Scenario 4, where not all customers can be served on the same route. For example, when applying PSSI with a perturbation factor of 10%, the average number of used vehicles in Scenario 3 is approximately 2.67, while in Scenario 4, it is around 2.75.

Demand increase is another crucial factor influencing fleet size and vehicle combinations. In Scenarios 3 and 4, where demand is 50% higher than in Scenarios 1 and 2, the average number of vehicles also increases accordingly. For instance, in Scenario 1, when applying PSSI (β₁), the average number of vehicles is 2.29, whereas in Scenario 3, it rises to an average of 2.79.

Comparing the solution methods and the results summarized in Table 2.12, it becomes evident that the highest number of vehicles is used when employing the FO approach. Implementing the PSS metaheuristic brings improvements in terms of vehicle usage and cost savings. However, there is no significant difference between the results obtained from the two PSS variations.

Table 2.10: Average vehicle numbers from FO and PSSI variations.

	Scenario1						Scenario2						Scenario3						Scenario4					
	FO		PSSI (β_1)		PSSI (β_2)		FO		PSSI (β_1)		PSSI (β_2)		FO		PSSI (β_1)		PSSI (β_2)		FO		PSSI (β_1)		PSSI (β_2)	
	L	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**	**L**	**S**
A	0	1	0	1	0	1	0	1	0	1	0	1	0.46	2	0.54	2	0.67	2	0.92	2	1.21	1	1.38	1

Table 2.11: Average vehicle numbers from FO and PSSII.

	Scenario1				Scenario2				Scenario3				Scenario4			
	FO		PSSII		FO		PSSII		FO		PSSII		FO		PSSII	
	L	S	L	S	L	S	L	S	L	S	L	S	L	S	L	S
Ave	0.50	1.79	0.50	1.75	0.46	1.83	0.5	1.8	0.46	2.33	0.67	1.92	0.92	2.17	0.8	1.92

Table 2.12: Average total vehicle usage for FO, PSSI and PSSII.

	Average Vehicle Usage			
	Scenario1	Scenario2	Scenario3	Scenario4
Method				
FO	2.29	2.29	2.79	3.08
PSSI (β_1)	2.13	2.08	2.67	2.75
PSSI (β_2)	2.08	2.13	2.67	2.75
PSSII	2.25	2.25	2.58	2.67
Min	2.08	2.08	2.58	2.67
Max	2.29	2.29	2.79	3.08

2.6.3.3 Computational time.

Regarding computational time, as shown in Table 2.13, the longest solution times are observed in Sc2 and Sc4. These scenarios involve grouping customers into specific clusters, where not all customers are allowed to be included. Consequently, serving all customers requires a higher number of vehicles, leading to an increase in potential solution choices. More choices for vehicle assignment result in longer solution durations. Among the different variations of the PSS metaheuristic, PSSII, which randomizes the subproblem order, achieves the shortest solution duration. This is because in FO, where the main problem is divided into integer and non-integer blocks of subproblems, the order of solving these subproblems significantly impacts the computation. Randomizing the solution order of subproblems leads to a decrease in CPU time.

Comparing the solution times for FO and the PSS versions while considering the number of solved instances, it is evident that the CPU times for the PSS variations are reasonable, given the improvements they bring in terms of solution quality.

Table 2.13: Average solution durations in seconds for FO, PSSI and PSSII.

	Average Solution Time			
Method	Scenario1	Scenario2	Scenario3	Scenario4
FO	746.28	135.12	38.76	231.72
PSSI (β_1)	991.94	5818.16	1056.24	9152.84
PSSI (β_2)	991.36	5758.25	1029.40	9043.18
PSSII	820.11	5947.11	813.44	7908.26
Min	746.28	135.12	38.76	231.72
Max	991.94	5947.11	1056.24	9152.84

2.7 CONCLUSION

The initial part of this thesis explores two new approaches to address the previously proposed problem of integrated fleet sizing and delivery planning, where candidate replenishment frequencies are pre-determined. Our objective is to determine replenishment plans for clients based on defined delivery frequencies, which are determined by the number of weeks between deliveries, including a daily replenishment option. The demands are transported to their respective destinations using a diverse fleet of trucks. Since this problem involves strategic decision-making, we need to use approximations for routing costs rather than precise details. Our main goal is to find the optimal customer-vehicle-frequency assignment at the lowest cost. However, due to the NP-hard nature of the problem, conventional tools like CPLEX may not provide the best solutions within a reasonable timeframe, particularly for larger instances.

In this study, we improved the ADP algorithm and showcased its efficacy by applying it to a range of randomly generated instances with diverse characteristics.

The second solution method implemented, the Problem Space Search (PSS) is a straightforward metaheuristic method that utilizes a heuristic to create a new search space within the original problem space by temporarily modifying the data or parameters of the problem-specific heuristic. We consider two versions of PSS in this research. In PSSI, cost data is perturbed and fed to the fix and optimize, while in PSSII, the order in which the fix and optimize subproblems are handled is randomized.

The results showed that the inclusion of clusters in the fleet sizing and replenishment planning problem led to more challenging problems with higher annual costs. This increase was attributed to the limitations imposed by geographical clustering on transportation operations and vehicle-customer assignment.

Applying PSS-based metaheuristics significantly reduced the gaps in comparison to the Fix-and-Optimize (FO) approach. The PSS variations demonstrated their effectiveness in enhancing the solution quality by generating diverse neighborhoods through cost data perturbation and subproblem order randomization. Among the PSS variations, PSS-I with a perturbation factor of 20% outperformed the other versions, exhibiting the minimum gaps in three out of the four scenarios.

Regarding fleet composition, the average number of assigned vehicles aligned with the patterns observed in objective function values. The addition of clusters led to increased average numbers of large and small vehicles. Demand increases also influenced fleet size and composition.

In terms of computational time, scenarios involving customer clustering and limitations on customer insertion exhibited higher solution times. PSSII demonstrated the lowest solution duration among the PSS variations. Comparing solution methods, it was evident that the PSS variations offered substantial improvements in both solution quality and computational efficiency, making them viable alternatives to the FO approach.

3 EXTENDED PROBLEM WITH GENERALISED DELIVERY PATTERNS

3.1 Introduction

Retailers are under pressure due to narrow profit margins, compelling them to consistently pursue operational excellence in logistics. Enhancing the effectiveness of product delivery from distribution centers to stores remains a continuous endeavor within the retail industry. As stated in Agrawal & Smith (2015) [1], retail establishments must ascertain the most advantageous delivery strategies and streamline the routes taken by vehicles to restock store inventory from warehouses. In the realm of grocery retail, stores often adhere to recurrent weekly demand trends. As a result, these retailers address both customer and store requirements by periodically restocking their stores through specific delivery schedules. These delivery patterns encompass a particular selection of weekdays when deliveries from the distribution center consistently reach a specific store within standard weeks – those that lack public holidays – over a designated planning period. Research evidence confirms that most grocery businesses adopt such delivery patterns, typically tailored based on sales volume and store dimensions (Kuhn & Sternbeck (2013) [32]).

Implementing repetitive and store-specific delivery patterns brings forth several advantages:

(1) Organizing the workforce for the task of replenishing shelves becomes notably simpler, as the restocking orders consistently arrive at stores on the same weekdays every week (Gaur & Fisher (2004) [20]).
(2) Similarly, from a transportation standpoint, these delivery patterns provide an opportunity to establish fundamental cyclic routes during each corresponding planning phase.
(3) In the distribution center, the allocation of staff and the planning of shifts can be aligned with projected cumulative picking volumes, which are contingent upon the selected delivery patterns across all stores
(4) Retailers generally adopt policies of periodic inventory assessment (Cachon (2001) [11]; Van Donselaar et al. (2010) [55]).

Whenever the inventory on store shelves reaches or dips below a predetermined reorder level, an order for replenishment is initiated. The adoption of a cyclic strategy for ordering and delivery permits regular adjustments to reorder levels. This reduction in management oversight simplifies logistics planning in subsequent planning phases (Schöneberg et al. (2010) [43], Hübner et al. (2013) [26]). As a consequence, the chosen delivery patterns significantly impact efficiency within the operational segments of an internal retail supply chain, encompassing the distribution center, transportation, and in-store logistics. Since the intervals between orders are a product of the applied delivery pattern, a series of volume effects arise along the supply chain, exerting substantial influence on logistics costs (Cachon (1999) [12]). Thus, the retail industry acknowledges the selection of delivery patterns as a vital lever to harmonize the requirements of the distribution center, transportation, and in-store aspects. Despite the complexity associated with constructing suitable models and tools, the majority of retailers haven't yet adopted comprehensive approaches that span all subsystems during the allocation of delivery patterns to stores. This underscores the need for effective decision support mechanisms to guide retail enterprises in achieving decisions that are either "optimal" or at the very least "nearly optimal.

Total cost of the retail distribution systems depends on routing and fleet related costs on a great deal. Considering the role of the effective fleet sizing in profitability and efficiency of the distribution systems, we integrated fleet dimensioning and assignment to our model.

Moving on to the problem explored in the second section of the thesis, it revolves around a collection of stores spread across a distinct geographical area. Here, the demand for the stores is deterministic but exhibits seasonal trends. The fleet utilized consists of both owned and rental heterogeneous trucks. Routing costs are approximated through clustering, where stores are assigned to seed points. Replenishments are conducted using suitable delivery patterns that not only fulfill the store demands but also minimize the overall cost of transportation, delivery pattern assignment, vehicle ownership/renting, and routing.

3.1.1 Motivation and related literature

In our literature survey, we mainly focused on a review of retail-specific articles that define models for the problem of delivery pattern assignment and provide solutions to the developed models. Sternbeck & Kuhn (2014) [48] focuses on the tactical problem of determining delivery patterns based on which grocery stores are routinely supplied with products from various order categories by retail-owned distribution hubs. They aim at minimizing the costs occurring in the store and the transportation DC. The aspect which makes our problem different from the one studied in Sternbeck & Kuhn (2014) [48], is that we determine the fleet size and combination in two demand seasons as well as considering the costs of owning and renting fleets. They solve the suggested model by CPLEX and analyze the results. In general, DP planning papers incorporate pattern-dependent costs and examine their impact on total planning. They specifically consider how the delivery sizes each day are affected by the DPs chosen. Gaur & Fisher (2004) [20] present a method for determining a weekly delivery schedule based on a periodic inventory routing problem. Ronen & Goodhart (2008) [40] analyze a similar situation that includes DC expenses as well as additional extensions such as restricted picking capacity, a heterogeneous fleet, and daily minimum utilization rates for DC and transportation subsystems. They assign similar stores to clusters and set patterns for them as we do in our problem, however, it is not performed sequentially. Furthermore, they overlook in-store operations expenditures. Sternbeck & Kuhn (2014) [48] are the first to thoroughly study the logistical processes in DCs, transportation, and retail, as well as their dependencies on DPs. Their binary integer program minimizes the sum of all specified costs and is applied to a real-world instance. A cost matrix based on distance and order size is used to estimate transportation expenses. Actual tours are neglected. Holzapfel et al. (2016) [24] consider DC, transportation, and in-store logistics, and provide an innovative solution strategy that clusters stores and approximates transportation costs using Fisher & Jaikumar (1981) [19] logic. There are some similarities between the problem studied in Holzapfel et al. (2016) [24] and the problem we consider here. We both put customers into clusters based on their proximity and then assign delivery patterns. In-store costs and transportation related costs are calculated the same way. Our problem is distinguishable from the one developed in Holzapfel et al. (2016) [24] in the following aspects:

(1) We consider monthly delivery patterns while they consider weekly delivery patterns.
(2) The vehicle fleet is heterogeneous in terms of carrying capacity, cost per kilometer and ownership costs.
(3) Owned and rental vehicle fleet is available in both seasons.
(4) The seed point of a cluster is chosen based on the proximity of a customer to a dummy seed.
(5) A limited number of stores can be served on daily clusters.

Here, we develop a planning concept to define repetitive delivery patterns according to which stores of a grocery retailer are supplied from a distribution center. A delivery pattern is a set of weekdays on which a delivery from the DC arrives to a certain store on a regular basis during all normal weeks, i.e., weeks without public holidays, in a planning horizon. As Kuhn & Sternbeck (2013) [31] prove, the majority of grocery companies use such delivery patterns, which are designed considering the volume of sales and the size of the stores.

According to Holzapfel et al. (2016) [24], using recurring and store-specific delivery patterns has various advantages:

1) Scheduling the workers for the shelf replenishment process is significantly easier because order replenishments arrive at a shop on the same weekdays each week (Gaur & Fisher (2004) [20])
2) Similarly, in terms of transportation, delivery patterns allow for the creation of basic cyclic routes during each planning period.
3) Staff deployment and shift planning can be altered at the DC based on predicted aggregate picking volumes, which are determined by the delivery patterns chosen across all stores.
4) Periodic inventory reviews are typically implemented by retailers. When shelf inventory falls to or below a reorder level, a replenishment order is issued. Using a cyclic ordering and delivery strategy allows reorder levels to be adjusted on a frequent basis, which reduces overhead steering and simplifies logistics planning in following planning modules.

As a result, the delivery patterns chosen have a significant impact on the operational subsystems of an internal retail supply chain, such as DC, transportation, and in-store logistics. As mentioned previously, effective vehicle fleet utilization has a significant impact on profitability of the companies who own or plan to rent a fleet. Hence, bringing quality solutions to strategic fleet sizing and composition problem is a critical logistical decision. Strategic fleet sizing problem mainly focuses on optimizing the profitability of the companies by determining the most suitable fleet size and composition. Detailed routing problems is normally not included at a highly detailed level in strategic fleet management problems since routing decisions are daily operational decisions. Hence, in our study, we consider routing costs in an approximated fashion because detailed routing decisions changes daily and routing costs constitute relatively small percentage of the total cost.

As far as we know, literature lacks research which considers integrated delivery pattern planning and fleet sizing as a whole. We made contribution to the literature in sense of developing a mixed integer programming model for the integrated delivery pattern planning and fleet sizing problem and proposing effective and novel solution techniques for the model considered.

3.2 Problem Description And Mathematical Formulation

This section, formulates a mixed integer programming model for selecting store-specific delivery patterns, the fleet size and composition used for replenishing stores and transportation tour clusters to minimize the total costs of the whole retail distribution chain. This model is extension of the model suggested Dastjerd & Ertogral (2019) [14] Each of the stores denoted by f, $f \in F$, is supplied from a DC considering the different delivery pattern r, $r \in R$. Delivery pattern defines the days of delivery in a specific repetitive delivery period, which is, one month in our problem setting. Each delivery pattern r is assigned to one store f in season s and is denoted by the binary variable $x_{frs} \in \{0,1\}$. Cost of assigning a distinctive delivery pattern to a store is illustrated by $\hat{c}_{frs}$ which includes ordering costs occured in stores, handling costs for the stores and inventory holding costs of the stores in each season. We use an incidence matrix g_{rt} to indicate whether a pattern includes a specific day or not. If day t is included in pattern r, the corresponding g_{rt}value is 1, and otherwise it is equated to zero. Here we assume that the delivery pattern of a store is repeated in all following periods, e.g., months, till a major change occurs. The products delivered to store f on day t in seaspn s is quantified by $pall_{frts}$ and it is assumed that each day the stores receive an amount of the products that fulfill the required demand for all articles until next delivery day. The quantity is measured in terms of receiving pallets of goods. Each store is considered to have a receiving capacity cap_f^{rec} which ensures that the received products fit into the store capacity. Likewise, the DC is considered to have picking capacity on each day $maxcap_t^{pick}$ and $mincap_t^{pick}$. The picking volume for store f on day t in season s is denoted by $pick_{frts}$.

Additionally, the retailer is supposed to have two types of owned vehicles which differ in terms of capacity, cost per kilometer and ownership costs. In high demand periods retailers are allowed to rent vehicles from a third-party company. Each store is assigned to a delivery tour k, a vehicle type v, a pattern r, on day t and season s if pattern r includes day t and that specific store is assigned to pattern r, $x_{fkrtvs} \in \{0,1\}$. Binary variable y_{krtvs} indicates whether a vehicle type v is assigned to the delivery tour k, pattern r on day t in season s.

The cost of serving a seed point and the additional costs occurring when a truck of type v assigned to a delivery tour k adds store f to its tour are distinguished by $c_{kv}^{trenstour}$ and $c_{kv}^{transstop}$, respectively. Any point in the distribution area can serve as a seed point. Here, in this study, we used a combination of k-means clustering method and a version of the technique suggested by Savelsbergh (1990) [41]. The transportation and routing costs are approximated by the same logic as the one presented in Fisher & Jaikumar (1981) [19].

Some assumptions we made in model development are as follows:

- Stores receive deliveries a maximum of once per day.
- Each store gets replenished by a store-specific monthly delivery pattern.
- The monthly delivery pattern repeats all months of the entire planning horizon, e.g., one year.
- Store delivery lead times are deterministic and predetermined.
- The product range is homogeneous in the sense of products that can be packaged together on a common pallet or roll cage.

- The transportation fleet is heterogeneous and unlimited.
- There are owned and rental fleets available for different demand seasons.
- Each truck can carry out a maximum of one delivery tour per day.
- The capacity for receiving goods at a store is limited and implies limited storage space in the backroom.
- The workload at the DC is restricted between a minimum and maximum level.
- Each truck can visit a limited number of stores in a single tour on a single day of each delivery pattern.

The notations used and the mathematical model is presented below:

Sets:

F: Stores, $F = \{1,2,\dots,f,\dots,|F|\}$ and $f = 0$ symbolizing the DC.

K: Clusters, tours, $K = \{1,2,\dots,k,..|K|\}$; $|K|$ quantifies the number of tours

R: Delivery patterns, $R = \{1,2,\dots,r,\dots,|R|\}$

T: Days, $T = \{1,2,\dots,t,\dots,|T|\}$, $|T|$ quantifies the length of one delivery cycle

V: Vehicles, $V = \{1,2,\dots,v,\dots,|V|\}$

S: Seasons, S= $\{1,2,\dots,|S|\}$, $s = 1$ and $s = 2$ represent low and high seasons, respectively

Parameters:

$\widehat{C_{frs}}$:	Costs at the subsystems DC and store, independent of tour setting, applying pattern r for store f in season s.
$C_{kv}^{transtour}$:	Transportation costs supplying cluster k with vehicle v
$C_{fkv}^{transstop}$:	Stoppage costs for cluster k with vehicle v
$C_{v}^{ownership}$:	Ownership cost of vehicle v
C_{v}^{rental}:	Renting cost of vehicle v

Cap_f^{rec}:	Receiving capacity of store f
Cap_v^{store}:	Number of stores that can be visited on a single delivery tour by vehicle v
Cap_v^{trans}:	Truck v capacity
g_{rt}:	Binary parameter; 1 if a delivery on day t takes applying pattern r; otherwise 0
$maxCap_t^{pick}$:	Maximum picking capacity to be used at DC on day t
$minCap_t^{pick}$:	Minimum picking capacity to be used at DC on day t
$pall_{frts}$:	Pallets delivered on day t to store f applying pattern r in season s
$pick_{frts}$:	Picking effort at DC on day t for store f applying pattern r in season s

<u>Decision Variables:</u>

x_{frs}:	Binary variable; 1 if pattern r is assigned to store f in season s; otherwise 0
x_{fkrtvs}:	Binary variable; 1 if pattern r is assigned to store f and store f is assigned to cluster k and vehicle v on day t in season s; otherwise 0
y_{ktvs}:	Total number of the vehicles of v assigned to cluster k on day t in season s
N_{vs}:	Total number of assigned rented vehicle v in season s
O_v:	Total number of assigned owned vehicle v

Mathematical Formulation:

$$TC_s = \sum_{f\in F}\sum_{r\in R}\sum_{s\in S}\widehat{C_{fr}}x_{frs} + \sum_{k\in K}\sum_{t\in T}\sum_{v\in V}\sum_{s\in S}C_{kv}^{transtour}y_{ktvs} + \sum_{f\in F}\sum_{k\in K}\sum_{r\in R}\sum_{t\in T}\sum_{v\in V}\sum_{s\in S}C_{fkv}^{transstop}x_{fkrtvs} + \sum_{v\in V}\sum_{s\in S}C_v^{rental}N_{vs} \qquad 3.1$$

$$\text{Min TC} = \sum_{s\in S}\alpha_s TC_s + \sum_{v\in V}C_v^{ownership}O_v$$

Subject to:

$$\sum_{r\in R} x_{frs} = 1 \qquad \forall f \in F, \forall s \in S \qquad 3.2$$

$$\sum_{v\in V} \sum_{k\in K} x_{fkrtvs} = g_{rt}\, x_{frs} \qquad \forall\, f \in F, r \in R, t \in T, \forall s \in S \qquad 3.3$$

$$minCap_t^{pick} \le \sum_{f\in F} \sum_{r\in R} pick_{frts}\, x_{frs} \le maxCap_t^{pick} \qquad \forall t \in T, \forall s \in S \qquad 3.4$$

$$\sum_{r\in R} pall_{frts} x_{frs} \le Cap_f^{rec} \qquad \forall\, f \in F, t \in T, \forall s \in S \qquad 3.5$$

$$\sum_{f\in F} \sum_{r\in R} pall_{frts}\, x_{fkrtvs} \le Cap_v^{trans} y_{ktvs} \qquad \forall\, v \in V, k \in K, t \in T, \forall s \in S \qquad 3.6$$

$$\sum_{k\in K} y_{ktvs} \le O_v \qquad \forall\, v \in V\backslash\{|V|\}, t \in T, \forall s \in S \qquad 3.7$$

$$\sum_{k\in K} y_{ktvs} \le N_{vs} \qquad \forall\, v = |V|, t \in T, \forall s \in S \qquad 3.8$$

$$\sum_{r\in R} \sum_{f\in F} g_{rt} x_{fkrtvs} \le Cap_v^{store} \qquad \forall\, v \in V, k \in K, t \in T, \forall s \in S \qquad 3.9$$

$$x_{frs} \in \{0,1\} \qquad \forall\, f \in F, r \in R, \forall s \in S \qquad 3.10$$

$$x_{fkrtvs} \in \{0,1\} \qquad \forall\, f \in F, r \in R, v \in V, \quad 3.11$$
$$k \in K, t \in T, \forall \in S$$

$$y_{ktvs} \in\in \{0,1\} \qquad \forall\, v \in V, k \in K, t \in T, \forall s \in S \qquad 3.12$$

$$N_{vs} \in Z^{\ge 0} \qquad \forall\, v \in V, \forall s \in S \qquad 3.13$$

$$O_v \in Z^{\ge 0} \qquad \forall\, v \in V \qquad 3.14$$

Our objective aims at minimizing the total annual costs caused by delivery pattern assignment, serving stores, adding stores to delivery tours, using owned and rented vehicle fleets. With the first constraint (3.2) we assure that in each of the two seasons only one delivery pattern is assigned to each of the stores. In second constraint (3.3), we assign a store to a vehicle and delivery tour with pattern r on day t in season s if and only if that store is assigned to pattern r and the day t is included in pattern r in that season. Constraint number (3.4) is used to assure that the picking effort at DC for each day is between its maximum and minimum picking capacity. Store receiving capacity is considered in constraint number (3.5). Constraint (3.6) restricts that the number of pallets transported by vehicle v on a delivery tour k on day t in season s must be less than or equal to the vehicle's capacity. By constraints (3.7) and (3.8) we calculate the total number of owned and rented vehicles which are used to perform the delivery operations. Due to time, distance and limited vehicle capacities each vehicle can serve a limited number of stores in a tour on a specific delivery day. We assured this by adding constraint (3.9) to our model. Constraints (3.10)-(3.14) gives the domain for the decision variables.

3.3 Complexity Analysis Of The Problem

In this section we show that the problem tackled in here is a NP-Hard problem through polynomial time reduction from the "One- dimensional bin packing problem" which is proved to be strongly NP hard in E. G. Coffman et al. (1997). In the bin packing problem, objects of different volumes must be packed into a finite number of bins or containers each of volume V in a way that minimizes the number of bins used.

Considering our problem, under some assumptions we can transform the current problem to bin packing problem in pseudo-polynomial time:

- A single season
- Single type owned vehicles
- single delivery tour/seed point
- single delivery pattern containing a single delivery day
- $C_{kv}^{transtour} = 1$
- $\widehat{C_{fr}} = C_{fkv}^{transstop} = C_{v}^{ownership} = 0$
- $C_{v}^{rental} = \infty$

Under these assumptions our problem turns into a one-dimensional bin packing problem, where we try to minimize the number of vehicles.

3.4 Solution Techniques

As solution method, we implemented three different versions of Fix and optimize heuristic with an additional improvement step. The Fix and Optimize heuristic is a two-phase approach which works by dividing a problem into smaller subproblems that are solved repeatedly. In the first phase, we break the problem down into subproblems/subgroups of variables and then solve it by setting variables in only one subproblem as integers, and taking the remaining variables as either fixed to the values found in the previous iterations or as linear variables. The solution for binary variables found in an iteration is fixed and used in the next one. This procedure is repeated until all variable values are found. The exact steps of first phase are given below.

FO heuristic –Phase I

1. Divide the customers into P sub groups with equal number of customers, or as equal as possible. Divide the decision variables in to P sub groups corresponding to P customer groups. Let A_i be the set of binary variables in ith sub group.
2. Set the iteration counter i to 1.
3. If $i > 1$, Fix the binary variables in A_j to the values of variables in iteration $i - 1$ for $j = 1..i - 1$.
4. Set the variables in A_i as binary variables.
5. Relax the binary variables in A_j for $j = i + 1..P$ as linear variables.
6. Solve the complete model
7. Set $i = i + 1$. If $i <= P$ go to step 3. If $i > P$, STOP.

The second phase aims at bringing improvement to the solution of the first phase. At each iteration, we set variables of one subproblem as binaries and reoptimize them while we keep all other variables fixed to the solution found in the previous iteration. If there is any improvement in the solution, the results are saved and procedure is continued till all the subproblems are checked. The steps for the second phase are given below.

FO heuristic –Phase II

1. Take the solution from Phase I as the current solution.
2. Set the iteration counter i to 1.

3. If $i = 1$ Fix the values of variables in A_j for $j = i + 1 \ldots P$ to the values found in the current solution.
4. If $i > 1$ Fix the values of variables in A_j for $j = 1 \ldots i - 1$ and $j = i + 1 .. P$ to the values found in iteration $i - 1$
5. Define the variables in A_i as binary variables. (Note that all other variables in other subgroups are fixed)
6. Solve the complete model and set $i = i + 1$.
7. Check the objective value, if the objective value is better than best solution so far and $i < P$, update the best solution as the current solution (improved solution) and go to step 4. If no improvement occurred in the current iteration and $i < P$ go to step 4. If no improvement occurred in the current iteration and $i > P$ STOP and RETURN the solution.

Fix and Optimize versions proposed here differ in terms of subproblem generation criterion:

1. Store based subproblems
2. Delivery day based subproblems
3. Cluster based subproblems

3.4.1 Store based subproblems

As it is apparent from the title, each sub group contains a fixed number of stores and the variables for one of those groups is defined as binaries and other groups are left as LPs. The solution of each iteration is fixed for the binary part and the iterations continue till all the variables are set to binaries or integers. The steps for the second step are the same as the ones described in previous reports.

3.4.2 Delivery day based subproblems

In delivery day based subproblem generation, we divide problem based on planning period. At each iteration, we define variables for a set of a number of t values (depending on the appropriate size of subproblems) as IPs and relax variables in other subproblems. In FO, the matter is always about the trade-off between solution quality and solution time reduction. That is, when choosing larger sized subproblems, the higher number of IP variables produce a better solution in terms of deviation from optimal/best bounds but may cause solution durations to increase significantly. On the other hand, smaller sized subproblems, may produce a worse solution in much more shorter time due the higher number of relaxed variables. So, subproblem size determination is a key decision in these types of heuristics.

3.4.3 Cluster based subproblems

In this version, we generated a new index set for the customers based on the ones that are served on the same delivery tour or are assigned to the same seed point. In preprocessing step for model solution, we used K-means algorithm for finding seed points and developed a model for assigning customer to their nearest seeds. Thus, when customers of the same delivery tour are taken as a subproblem in fact the customers which are nearest to each other are put into the same subproblem. FO, divides the main problem into smaller subproblems for one of which the variables are defined as binaries and for the others as linearly relaxed ones.

Stores are assigned to a seed point/cluster in preprocessing step by using a k-means clustering algorithm and a mathematical model. Then, the indices for the stores which are assigned to the same cluster are stored and passed to the FO for being used in subproblem generation. In this version of FO, we put the customers of same clusters in the same subproblem and apply FO steps. The logic behind this division criteria is that the customers are assigned to the seeds/clusters which are closest to them in terms of distances. In our main model, we aim at minimizing total annual costs where one of the cost components is the cost of transportation that considers distances from stores to seed points and from stores to other stores. When stores in the same cluster are put into the same subproblem, the distance is automatically minimized hence the solution quality increases.

3.5 Improvement Algorithm

We introduce a two-phase enhancement algorithm aimed at further refining the outcomes achieved through the FO approach. Upon dissecting the characteristics of the solutions produced by FO, it becomes apparent that the notable deviations from best-bound values primarily stem from the excessive utilization of vehicles. Given that vehicle ownership costs constitute a predominant expense in our context, the addition of an extra vehicle compared to the optimal solution significantly impacts the objective function's value. The detailed procedure during the steps of the improvement algorithm is described in the following subsections.

3.5.1 Improvement algorithm phase I

The enhancement algorithm concentrates on the optimization of fleet composition and size. In the initial phase, we meticulously investigate solutions in which each vehicle type in the FO-derived solution is systematically augmented and reduced by one unit, while keeping the quantities of other vehicles constant.

In the first phase of the algorithm, after fixing all the variables to integer values by fix and optimize implementation, the final vehicle fleet composition is saved in an array V as values for x. Next, one large vehicle is added to the fleet and the total number of large vehicles is used as an upper bound for large vehicle assignment (the new vehicle set is called V'). If the objective value for large vehicle addition, $Cost'$ is smaller than the value from the FO, $Cost$, we add another large vehicle to the fleet and update the objective function value. The process goes on in this fashion until there is no improvement in the total cost. On the other hand, if no improvements occur, the algorithm moves to the next step. That is, one large vehicle is excluded from the fixed vehicles set taken from FO. This operation goes on again until there is no improvement in the total cost. The next move is to add a small vehicle to the fleet and repeat it until stops improving the objective. The last step of first improvement phase is to exclude one small vehicle from the fleet and reapply FO to improve the objective function value as much as possible.

3.5.2 Improvement algorithm phase II

In the second phase of the improvement approach, we defined two neighborhoods for the existing solution: exclude one large vehicle and add a small vehicle instead and remove two small vehicles and add a large vehicle to the fleet. The revised vehicle set V'' is used as upper bound for vehicle assignment constraint. FO is applied to the problem using defined neighbors. If the solution from second step, $Cost''$ is better than the one from the first step, $Cost'$, the solution is updated and first phase is re-started using new vehicle composition. If not, algorithm stops and reports the solution for the problem.

The flowchart for improvement algorithm steps is given in Figure **3.1**.

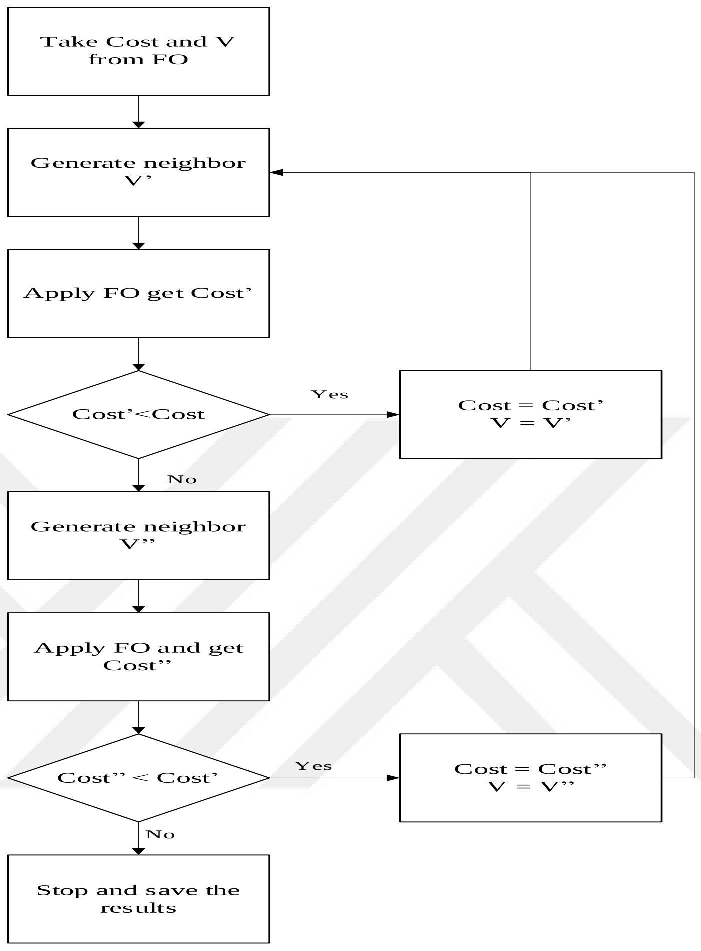


Figure 3.1: Steps for improvement algorithm.

3.6 Numerical Analysis

For solving the MIP, we need to pre-calculate some parameters at first. That is, the location of seeds should be known in order to calculate the transportation and tour generation costs. In the following subsections, we describe the process to seed generation, transportation cost calculation and also explain the dataset generation method.

3.6.1 Seed point generation

Number of clusters, $|K|$, is needed as an input for the model. Number of clusters is equal to the maximum number of feasible delivery tours per day and the number of the available trucks for seed point generation. We approximated this number by dividing the number of stores by the maximum possible number of stores which can be visited in a single tour/cluster and multiplied the results with seasonal increase parameter 1.5; that is, for example for $f = 20$ we have $k = \lceil 20/6 \rceil * 1.5 = 6$. Rounding up this quantity to the next integer value results in the number of trucks that are necessary to ensure that all stores can receive their required products on all days of the delivery cycle. This approximation generates a reasonable number of trucks and reasonable delivery cluster sizes. In the unlikely case that the number of trucks approximated does not lead to a feasible solution, the number is increased by one until a feasible solution can be generated. As a result, we have 6, 9, 12 and 18 seed points/clusters for 20, 30, 40 and 60 stores, respectively.

After approximating the number of possible seeds and trucks, we use the following algorithm in order to determine the exact seed locations and generate delivery tours:

(1) Defines the initial locations of seed points for all clusters k. There are many different ways to generate initial seeds, here we used the points which are randomly distributed along the delivery area. Random coordinates are generated using MATLAB and then feed into JAVA for the rest of the procedure. Distance between seed points, represented by $crdSP_k^{x(l)}$ and $crdSP_k^{x(l)}$, and the store coordinates, given by $crdSt_f^x$ and $crdSt_f^y$ is calculated using a Euclidean metric. The iteration index is denoted by l.

(2) Step 2 entails assigning clusters and determining the location of seed points

(2.1) An MIP is used to minimize the travel distances TD between seed points and their associated stores by assigning store f to cluster k, expressed by $z_{f,k} \in \{0,1\}$ (Table 3.1 presents the notations for the MIP). Each store is assigned exactly to one cluster (see (2)). Constraint (3) ensures that the volume to be delivered to all stores assigned to a specific cluster- assuming daily deliveries- must be equal or less than the truck capacity, cap^{trans},on each day of the delivery cycle. This guarantees that all of the stores in a certain cluster can potentially be supplied on one single tour each day.

Table 3.1: Notations used in cluster generation model.

Parameters	
$vol_{f,t}$	Demand volume at store f on day t, measured in pallets
$crdSt_f^x, crdSt_f^y$	x and y coordinates of store f
Decision and auxiliary variables	
$c_{f,k}^{dist}$	Distance between store f and the seed point of cluster k
$crdSP_k^x, crdSP_k^y$	x and y coordinates if the seed point of cluster k
$z_{f,k}$	Binary variable; *1* if store f is assigned to cluster k; otherwise, *0*

$$Min\, TD^l = \sum_{f\in F}\sum_{k\in K} c_{f,k}^{dist(l)}.z_{f,k} \tag{3.15}$$

s.t.

$$\sum_{k\in K} z_{f,k} = 1 \qquad \forall f \in F \tag{3.16}$$

$$\sum_{f\in F} vol_{f,t}.z_{f,k} \le cap^{trans} \qquad \forall k \in K, t \in T \tag{3.17}$$

$$z_{f,k} \in \{0,1\} \qquad \forall f \in F, k \in K \tag{3.18}$$

(2.2) If $TD^{(l)}$ is not equal to the previous iteration $TD^{(l-1)}$ or at least there is a difference greater than a small number ϵ between them, we update the seed point coordinates for the iteration $l+1$ as follows:

$$crdSP_k^{x(l+1)} = \sum_{f\in F} crdST_f^x \,.\, z_{f,k}^{(l)} / \sum_{f\in F} z_{f,k}^{(l)}$$

$$crdSP_k^{y(l+1)} = \sum_{f\in F} crdST_f^y \,.\, z_{f,k}^{(l)} / \sum_{f\in F} z_{f,k}^{(l)}$$

That is, the coordinates of the seeds in iteration $l+1$ is set to the mean of the coordinates of the stores assigned within the previous iteration. Then the value of $c_{f,k}^{dist(l+1)}$ is updated and this procedure continues until there is no further improvement. The steps for k-means algorithm are described below:

```
(1)     Initialize seed points and distances
            Set $l = 1$ and $TD^{(0)} = 0$
            Initialize $crdSP_k^{x(l)}$ and $crdSP_k^{y(l)}$   $\forall k \in K$
            Calculate $c_{f,k}^{dist(l)}$   $\forall f \in F, k \in K$
(2)     Iteration
            (2.1) Assign stores to the clusters by solving the MIP according to
(3.15)- (3.18)
            (2.2) Check objective value and update location of seed point if necessary
                If $|TD^{(l)} - TD^{(l-1)}| > \epsilon$ then
            Set $l = l + 1$, calculate $crdSP_k^{x(l)}$ and $crdSP_k^{y(l)}$   $\forall k \in K$   and  $c_{f,k}^{dist(l)}$
$\forall f \in F, k \in K$
            Continue with (2.1)
Else stop iteration
```

The process yields the location of the seed points for each cluster k, as well as an exhaustive and disjunctive assignment of all stores f to a single cluster k. The calculation of the transportation cost parameters $c_{kv}^{transtour}$ and $c_{fkv}^{transtop}$ is based on this assignment.

3.6.2 Transportation and routing cost calculation

The calculation of $c_{kv}^{transtour}$ and $c_{fkv}^{transtop}$ is performed based on the method suggested in Holzapfel et al. (2016) [24]. They apply the logic of Fisher & Jaikumar (1981) [19] and approximate the real tour costs on a tactical level.

Figure 3.2 illustrates the determination of the two transportation cost parameters for a certain assignment on the basis of distance- dependent transportation costs. For example, there are two possibilities for supplying store 4 from the DC, i.e., *f* = 0: either on a tour based on the cluster with the seed point *A* or on a cluster with the seed point *B* (A, *B* ∈ *K)*. The calculation example shows that assigning store 4 to cluster *A* results in lower costs than assigning it to cluster *B*. Supplying store 4 as part of the tour of cluster *B*, however, can be advantageous, if for example stores 1, 2 and 3 are not supplied on certain days, but stores 5 and 6 are. Then it is favorable to include store 4 on a tour with seed point *B* and not to start a new tour with seed point *A* on this day.

In our dataset we took cost per kilometer as 7,8.5 and 10 for small owned, large owned and rental vehicle, respectively. Ownership and rental cost are taken as 70,85 and 100. In terms of carrying capacity, we assumed that small trucks are capable of carrying average monthly demand of the stores with a tolerance of 30 pallets. Hence, the capacities are 110,160 and 160 pallets, respectively.

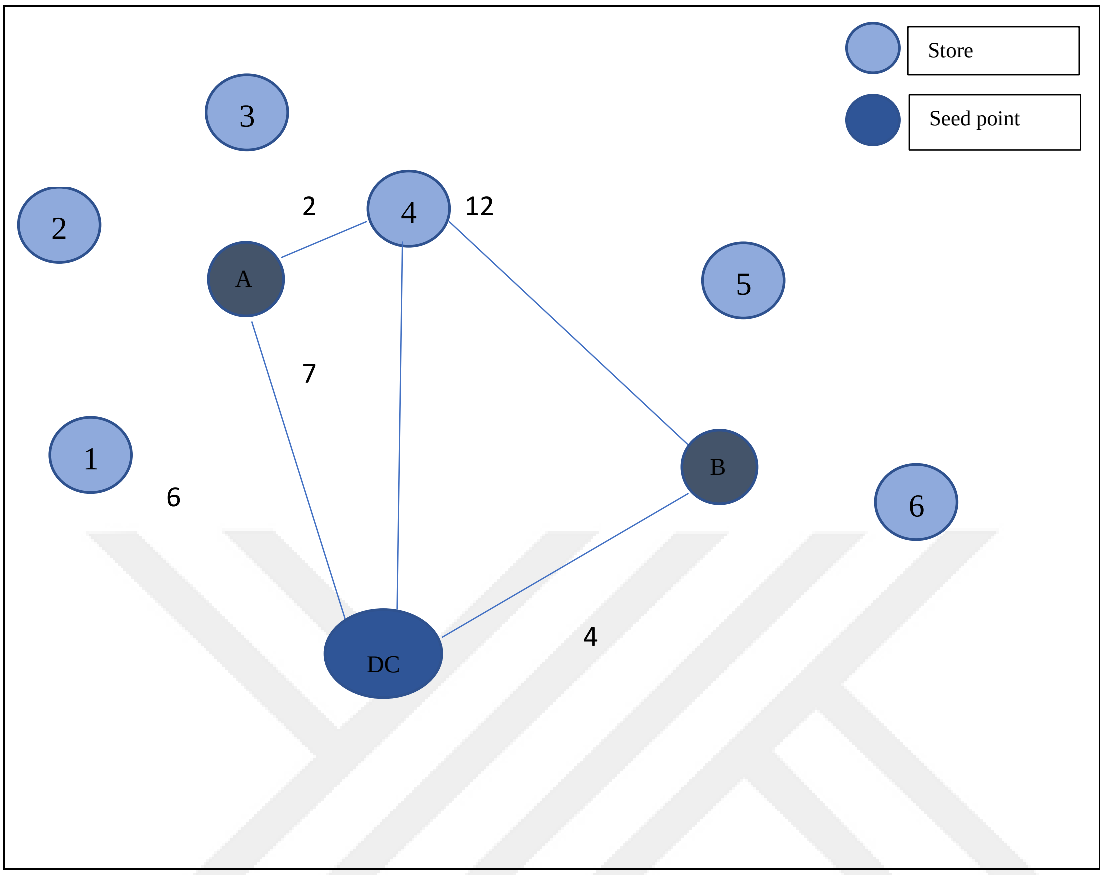
$$c_A^{transtour} = 2.c_{0A} = 2.6 = 12$$

$$c_B^{transtour} = 2.c_{0B} = 2.4 = 8$$

$$c_{4A}^{transtop} = c_{04} + c_{4A} - c_{0A} = 6 + 2 - 7 = 1$$

$$c_{4B}^{transtop} = c_{04} + c_{4B} - c_{0B} = 6 + 12 - 4 = 14$$

Figure 3.2: Example of calculating cost parameters $c_{kv}^{transtour}$ and $c_{fkv}^{transstop}$.



3.6.3 Dataset generation

We defined 8 general scenarios based on the combination of two different sizes of delivery areas, i.e., “District” (200 kilometer ×200 kilometer) and “State” (400 kilometer × 400 kilometer), three different store quantities located in these areas, i.e., $|F| = 20, 30, 40{,}60$; and one delivery pattern sizes i.e., $|R| = 46$. According to Kuhn & Sternbeck (2013) [31], several managers mentioned that they prefer order patterns with equidistant intervals between the deliveries. Hence, we designed our delivery patterns in a way that the interval between two consecutive replenishments is as equal as possible. Scenarios and their corresponding IDs are illustrated below in Table 3.2.

Table 3.2: Scenario IDs.

Scenarios	Number of stores	Area size	Number of patterns
1	20	200*200	46
2	20	400*400	46
3	30	200*200	46
4	30	400*400	46
5	40	200*200	46
6	40	400*400	46
7	60	200*200	46
8	60	400*400	46

Daily demand for each of the corresponding store sizes was generated randomly in the range [1,10]. In our case, we have 4 months of low and 8 months of high demand seasons in a typical delivery year. The daily store demands in high season periods are assumed to be 1.5 times the daily demand quantities during the low season periods. Delivery patterns are taken as four weeks of one month which will be repeated during a year without any change. For further illustration, a part of the patterns is expressed in Table 3.3.

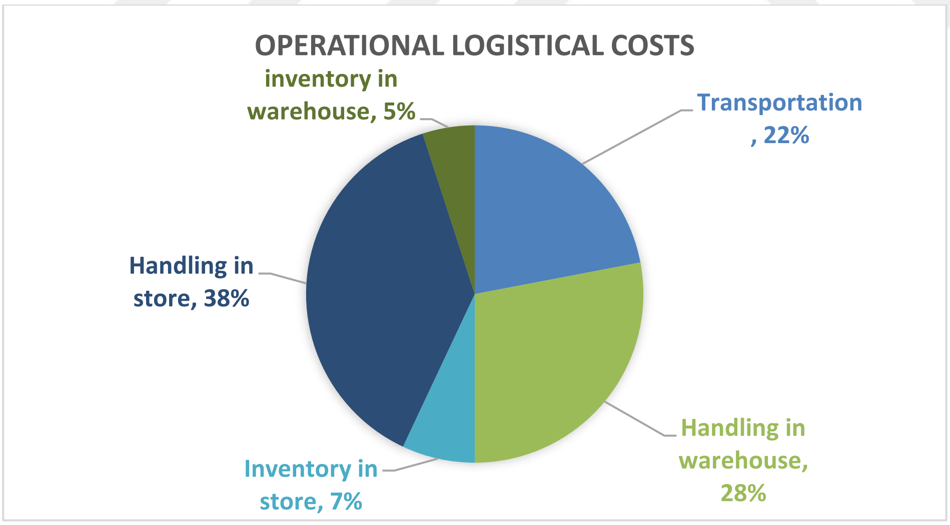
Table 3.3: Part of days for delivery pattern of size $|\boldsymbol{R}| = \mathbf{46}$.

	M	**T**	**W**	**T**	**F**	**S**	**M**	**T**	**W**	**T**	**F**	**S**	**M**	**T**	**W**	**T**	**F**	**S**	**M**	**T**	**W**	**T**	**F**	**S**
1	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0
2	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0
3	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0
4	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0
5	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0
6	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X	0	0	0	0	0	X
7	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0

8	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X	0	X
9	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0
10	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0
11	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X	0	0	X
12	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0
13	0	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0	0
14	0	0	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0
15	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0
16	0	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0
17	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0
18	X	0	0	0	0	X	0	0	0	0	0	0	X	0	0	0	0	X	0	0	0	0	0	0
19	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Cost parameters were generated based on the percentage chart presented in Van Zelst et al. (2009) [56]. It is proved that, the operational logistical costs in the retail supply chain for non-perishable goods follows the trend given in Figure 3.3.

Figure 3.3: operational logistical costs in the retail supply chain for non-perishable goods.



Considering Figure 3.3, related cost parameters are presented in Table 3.4:

Table 3.4:Operational costs of stores.

Operational Costs at	Value
Handling	50
Ordering	140
Inventory holding	20

Receiving capacities for each of the stores is equal to the highest delivery volume among all of the different delivery patterns. Maximum picking capacity at DC is approximated with the total picking effort assuming all outlets are supplied daily with their average daily demand volume in high season periods. Minimum picking capacity is taken as 10% of the maximum picking effort.

3.7 Numerical Results

In this section, we analyze the results from CPLEX and investigate the performance of the suggested heuristic versions.

3.7.1 CPLEX results

Results from CPLEX are represented in Table 3.5. In the first row of the table, V1 and V2 represent the total number of small and large owned vehicles used in both seasons. R1 and R2 show the number of vehicles that were rented for low and high demand seasons. All of the problems are solved under a time limit of 10800 seconds. None of the problems were solved to optimality and the average deviation percentage from optimal objective function value is equal to 2.72% . As it is seen from Table 3.5, the gaps from optimal solution increase with the increase in area size and store numbers. Combination of vehicle fleet changes with the problem size, as well. Generally, smaller trucks are preferred for carrying out the distribution operations. As the number of stores increase, we observe that the total number of used vehicles increase. This pattern is specially noticed in high demand seasons. The fact behind this increase is that the size of the batches to be carried grow as the number of stores increase and the carrying capacity limitation necessitates assigning more vehicles for satisfying the needs of stores.

Table 3.5: CPLEX results.

	CPLEX						
	Best Bounds	Gap	V1	V2	R1	R2	CPU (s)
1	1.75E+08	0.60%	2	0	0	0	10814.57
2	1.93E+08	3.75%	3	0	0	0	10825.89
3	2.76E+08	1.56%	4	1	0	0	10844.58
4	2.98E+08	2.22%	5	1	0	0	10830.77
5	3.97E+08	2.03%	7	1	1	1	10856.71
6	4.42E+08	4.33%	6	2	1	1	10852.25
7	6.20E+08	2.73%	7	3	1	1	10896.24
8	6.96E+08	4.51%	7	3	1	0	10894.39

3.7.2 Results for FO

In this section we discuss the results from three different types of the FO heuristic and the additional improvement step we applied to the existing problem settings. Results are classified and investigated under 3 categories: percentage deviations from best bounds of the CPLEX, computational times and fleet composition.

Gaps from best bounds for FO-I, FO-II and FO-III are presented in Table 3.6. FO-I, is the fix and optimize version in which the subproblems are generated based on stores. Main problem is divided to subproblems each containing five stores and the FO procedure is implemented. Objective function values for this version of the FO are at most 3.41% away from the best bounds obtained from CPLEX which means that the results may be closer to the optimal value.

FO-II and FO-III, show the same performance as well, the gaps from best bounds for FO-II is at most 0.94% and for the FO-III it is 4.53%. FO-II is the version in which the subproblems are generated based on delivery days. The fixation manner for delivery days influences the objective function value the most. It influences inventory, transportation and fleet ownership/rental costs. It is obvious that FO-II makes better decisions in terms of delivery days which leads into lower difference between best bounds and FO-II solutions.

In FO-III the stores on the same delivery tour are put in the same subproblem. The rationale behind this division type is that the customers are assigned to the clusters based on their proximity. Hence, the distance minimization is done in a much more effective manner in this type of subproblem division. On the other hand, the obligation for serving some specific stores all together in a single delivery tour limits the fleet and pattern assignments. Truck capacities are fixed and limited, since patterns and delivery days are selected in a way that trucks can carry demands of the all stores in a single tour. As we stated, each delivery tour is served using a single truck on a specific day. To be more illustrative, take the example of problem number 2. Gap for FO-II is 0.87% while gaps for FO-III is equal to 4.53%. Comparing the detailed solution values for these to heuristic versions, it is spotted that the delivery day based division utilizes fewer daily patterns than the cluster based division. This detail proves the fact that carrying all the demands for all the stores on the same delivery tour in a specific day forces the model to use smaller batch sizes and more frequent deliveries. This results in the growth of the transportation costs. Comparing the transportation cost components for FO-II and FO-III, which are equal to 4.00E+07 and 5.21E+07, respectively, confirms the difference in pattern selection manner for these methods. Inventory related costs are the components which are greatly influenced by pattern selection as well.

Comparing the heuristics with each other, it is obvious that FO-II outperforms the other two versions with an average gap of 0.56% which is the least among all.

For further evaluation of FO-II, 5 random instances are generated for each problem setting. These instances differ in terms of demand value, store and seed point coordinates. That is, in total, FO-II is tested on 40 instances and the average gaps and solution times are presented in Table **3.7**. As it is obvious, the percentage deviations are under 2% which is a prove for efficiency of suggested method in terms of producing quality solutions.

Table 3.8 demonstrates the solution durations for three heuristic versions. As we previously mentioned, the problems are first solved by CPLEX within a time limit of three hours and the best bounds and gaps from optimal solution is reported. As it is obvious from Table 3.8, suggested heuristic methods are efficient in terms of computational time since the average CPU times in second for all of them is under 3 hours. There is not any meaningful difference between solution times among three heuristic versions but we can claim that FO-II performs better than the other two methods. The largest computation time belongs to FO-I, which is due the fact that the subproblem handling order and generation criteria directly affects the complexity of each subproblem. Considering average solution times presented in Table 3.7, it is seen that all of the problems are solved in a period under 3 hours which is the time limit we set for the CPLEX. That is, FO-II is able to yield quality solutions n acceptable durations.

Considering fleet composition, as it is presented in Table 3.9, there is a general pattern being repeated. That is, in heuristic methods, the number of assigned vehicles are less than the ones in CPLEX solution but the total costs are higher. Investigating cost components and pattern-delivery day combinations, we found out that heuristics generally prefer deliveries with smaller batches which results in less vehicle utilization and more frequent replenishments. This replenishment schedule, reduces vehicle ownership/rental cost and inventory holding costs but increases transportation costs significantly.

Overall, we can claim that the proposed FO versions and improvement step are efficient solution techniques in our problem case.

Table 3.6: Percentage deviations from best bounds for FO-I, FO-II and FO-III.

	FO-I	**FO-II**	**FO-III**
1	2.93%	0.70%	3.03%
2	3.41%	0.87%	4.53%
3	0.98%	0.20%	4.13%
4	1.71%	0.21%	1.59%
5	0.76%	0.42%	0.87%
6	0.29%	0.94%	1.13%
7	0.59%	0.75%	0.86%
8	0.21%	0.38%	2.67%
Average	1.36%	0.56%	2.35%

Table 3.7: Average gaps and CPU (s) times for FO-II.

	Average gaps	**Average CPU**
1	0.70%	632.99
2	1.55%	840.24
3	0.51%	1060.99
4	0.58%	2742.49
5	0.46%	2228.59
6	1.20%	3047.9
7	1.06%	9719.82
8	0.55%	7849.91

Table 3.8: CPU times in seconds for FO-I, FO-II and FO-III.

	FO-I	**FO-II**	**FO-III**
1	708.09	388.46	616.20
2	501.84	805.13	462.79
3	1704.38	858.31	2227.71
4	2262.84	3229.64	1456.04
5	4560.39	2516.61	6356.74
6	8277.98	3650.72	8059.22
7	8700.93	7378.76	3690.91
8	9120.68	7528.35	3502.99
Average	4479.64	3294.49	3296.58

Table 3.9: Fleet composition for CPLEX, FO-I, FO-II and FO-III.

	CPLEX				**FO-I**				**FO-II**				**FO-III**			
	V1	V2	R2	R2	V1	V2	R2	R2	V1	V2	R2	R2	V1	V2	R2	R2
1	2	0	0	0	3	0	0	0	3	0	0	0	3	0	0	0
2	3	0	0	0	3	0	0	0	3	0	0	0	3	0	0	0
3	4	1	0	0	3	0	0	0	4	1	0	0	4	0	0	0
4	5	1	0	0	4	1	0	0	4	0	0	0	5	0	0	0
5	7	1	1	1	5	0	1	0	5	0	0	0	5	0	0	0

6	6	2	1	1	5	0	1	0	4	0	0	1	5	0	0	0
7	7	3	1	1	9	0	0	0	7	0	0	0	9	0	0	0
8	7	3	1	0	9	1	0	0	7	0	0	0	8	0	0	0

3.8 Conclusion

In the second part of this thesis, we first developed a mixed integer programming model for the integrated fleet sizing and replenishment planning problem with delivery patterns. Empirical findings support the notion that the majority of grocery retailers implement these delivery patterns, often customized according to their sales volumes and the physical dimensions of their stores. Holzapfel et al. (2016) [24] highlight the numerous advantages of employing recurring and store-specific delivery patterns in retail logistics as simplified workforce scheduling, efficient transportation planning, dynamic staff deployment, and streamlined inventory management. The choice of delivery patterns exerts a substantial influence on the functioning of operational components within an internal retail supply chain, including the distribution center (DC), transportation, and in-store logistics. Effective utilization of vehicle fleets significantly affects the profitability of companies that own or plan to rent fleets. Consequently, addressing the strategic fleet sizing and composition problem emerges as a critical logistical decision. This problem primarily aims to optimize a company's profitability by determining the most suitable fleet size and composition. Unlike detailed routing problems, which involve daily operational decisions, strategic fleet management problems typically do not delve into routing intricacies at a highly detailed level. As a result, our study approximates routing costs because detailed routing decisions vary daily and constitute a relatively small percentage of the total cost.

Next, a comprehensive examination of the results obtained from three distinct variations of the FO heuristic, supplemented by an additional improvement step applied to the existing problem settings is presented. The findings have been systematically categorized and scrutinized across three key dimensions: percentage deviations from the best bounds provided by CPLEX, computational time requirements, and fleet composition. In the realm of comparative analysis, it becomes evident that FO-II outperforms the other two versions, boasting an average gap of merely 0.56%, the lowest among all.

To further evaluate the heuristic's performance, random instances were generated for each problem setting, resulting in a comprehensive analysis based on 40 instances for FO-II. The results consistently demonstrate percentage deviations under 2%, a testament to the efficiency of the suggested method in delivering high-quality solutions.

In terms of computational time, FO-II outperforms the other two versions. All of the three versions solve the instances in a time less than 2 hours which indicates that they beat the CPLEX from this aspect, as well.

In summary, the proposed FO heuristic variations, complemented by the additional improvement step, have demonstrated their efficiency as solution techniques for the addressed problem case. These findings underscore the potential for optimizing delivery patterns, enhancing fleet sizing, and the importance of decision support mechanisms across all subsystems in the retail industry.

4 CONTRIBUTION TO THE LITERATURE AND FUTURE RESEARCH

In this thesis, we consider an important logistical problem. A basic and extended version of the mixed integer programming model is developed. The basic model aims at making decisions about fleet size and replenishment schedules simultaneously while tries to minimize the total annual cost of replenishing a set of geographically dispersed customers using a single customer specific delivery frequency and vehicle type. In this problem, all of the inventory related, routing related and fleet ownership related costs are included. A significant simplification brought to the problem of consideration is that the routing costs are handled in an approximate manner. Complexity of the problem is analyzed and it is proved to be an NP- hard one according to its reduction to one dimensional bin-packing problem. To solve the problem and bring quality solutions two effective metaheuristic techniques are proposed, namely: ADP with an improvement step and PSS. The effectiveness of the suggested methods is proved by testing them on a set of semi real-life dataset.

The second section of the thesis, handles an extended version of the same problem, where the delivery patterns are in a more generalized form, demand is exposed to seasonality and rental fleet option is available. The decisions to be made here are:

1. assignment of stores to delivery patterns, delivery tours and vehicles
2. assignment of vehicles to delivery tours and delivery days

The chief objective of the problem in chapter II is to minimize the total amount of the costs incurred by delivery pattern assignment, routing, vehicle ownership, using rental vehicles and holding inventories. Routing costs are calculated based on an approximation technique put forward by Fisher & Jaikumar (1981) [19]. The problem is proved to be NP-hard and three versions of fix and optimize heuristic with an additional improvement step are suggested. The results from FO implementation show the efficiency of the proposed method both in terms of computational time and solution quality. To sum up, we contribute to the literature by developing two mathematical formulations and effective solution techniques for a key logistical problem.As future work the following one can:

1. Propose different novel heuristics for the basic model
2. Develop the mathematical formulation for the case with stochastic demand and propose efficient solution techniques

Develop a two-echelon system considering the policies of the distribution center and generate efficient solution heuristics

REFERENCES

[1] **Agrawal, N., & Smith, S. A.,** (2015). *Retail supply chain management*. Springer.

[2] **Albrecht, A. R., Panton, D. M., & Lee, D. H.,** (2013). Rescheduling rail networks with maintenance disruptions using problem space search. *Computers & Operations Research, 40*(3), 703-712.

[3] **Astaraky, D., & Patrick, J.,** (2015). A simulation based approximate dynamic programming approach to multi-class, multi-resource surgical scheduling. *European Journal of Operational Research, 245*(1), 309-319.

[4] **Aziez, I., Côté, J.-F., & Coelho, L. C.,** (2022). Fleet sizing and routing of healthcare automated guided vehicles. *Transportation research part E: logistics and transportation review, 161*, 102679.

[5] **Bellman, R., & Kalaba, R.,** (1957). Dynamic programming and statistical communication theory. *Proceedings of the National Academy of Sciences, 43*(8), 749-751.

[6] **Bertazzi, L., Paletta, G., & Speranza, M. G.,** (2005). Minimizing the total cost in an integrated vendor—Managed inventory system. *Journal of heuristics, 11*, 393-419.

[7] **Bertazzi, L., & Speranza, M. G.,** (1999). Inventory control on sequences of links with given transportation frequencies. *International Journal of Production Economics, 59*(1-3), 261-270.

[8] **Bertazzi, L., Speranza, M. G., & Ukovich, W.,** (1997). Minimization of logistic costs with given frequencies. *Transportation research part B: Methodological, 31*(4), 327-340.

[9] **Bertazzi, L., Speranza, M. G., & Ukovich, W.,** (2000). Exact and heuristic solutions for a shipment problem with given frequencies. *Management Science, 46*(7), 973-988.

[10] **Bertsimas, D., & Demir, R.,** (2002). An approximate dynamic programming approach to multidimensional knapsack problems. *Management Science, 48*(4), 550-565.

[11] **Cachon, G.,** (2001). Managing a retailer's shelf space, inventory, and transportation. *Manufacturing & Service Operations Management, 3*(3), 211-229.

[12] **Cachon, G. P.,** (1999). Managing supply chain demand variability with scheduled ordering policies. *Management Science, 45*(6), 843-856.

[13] **Chen, H.,** (2015). Fix-and-optimize and variable neighborhood search approaches for multi-level capacitated lot sizing problems. *Omega, 56*, 25-36.

[14] **Dastjerd, N. K., & Ertogral, K.,** (2019). A fix-and-optimize heuristic for the integrated fleet sizing and replenishment planning problem with predetermined delivery frequencies. *Computers & Industrial Engineering*, *127*, 778-787.

[15] **Desrochers, M., & Verhoog, T.,** (1991). A new heuristic for the fleet size and mix vehicle routing problem. *Computers & Operations Research*, *18*(3), 263-274.

[16] **Dorneles, Á. P., De Araújo, O. C., & Buriol, L. S.,** (2014). A fix-and-optimize heuristic for the high school timetabling problem. *Computers & Operations Research*, *52*, 29-38.

[17] **Drechsel, J., & Kimms, A.,** (2011). Cooperative lot sizing with transshipments and scarce capacities: solutions and fair cost allocations. *International Journal of Production Research*, *49*(9), 2643-2668.

[18] **Federgruen, A., Meissner, J., & Tzur, M.,** (2007). Progressive interval heuristics for multi-item capacitated lot-sizing problems. *Operations Research*, *55*(3), 490-502.

[19] **Fisher, M. L., & Jaikumar, R.,** (1981). A generalized assignment heuristic for vehicle routing. *Networks*, *11*(2), 109-124.

[20] **Gaur, V., & Fisher, M. L.,** (2004). A periodic inventory routing problem at a supermarket chain. *Operations Research*, *52*(6), 813-822.

[21] **Gintner, V., Kliewer, N., & Suhl, L.,** (2005). Solving large multiple-depot multiple-vehicle-type bus scheduling problems in practice. *Or Spectrum*, *27*, 507-523.

[22] **Gören, H. G., & Tunalı, S.,** (2015). Solving the capacitated lot sizing problem with setup carryover using a new sequential hybrid approach. *Applied Intelligence*, *42*, 805-816.

[23] **Helber, S., & Sahling, F.,** (2010). A fix-and-optimize approach for the multi-level capacitated lot sizing problem. *International Journal of Production Economics*, *123*(2), 247-256.

[24] **Holzapfel, A., Hübner, A., Kuhn, H., & Sternbeck, M. G.,** (2016). Delivery pattern and transportation planning in grocery retailing. *European Journal of Operational Research*, *252*(1), 54-68.

[25] **Hua, Z., Zhang, B., & Liang, L.,** (2006). An approximate dynamic programming approach to convex quadratic knapsack problems. *Computers & Operations Research*, *33*(3), 660-673.

[26] **Hübner, A. H., Kuhn, H., & Sternbeck, M. G.,** (2013). Demand and supply chain planning in grocery retail: an operations planning framework. *International Journal of Retail & Distribution Management*, *41*(7), 512-530.

[27] **Hulshof, P. J., Mes, M. R., Boucherie, R. J., & Hans, E. W.,** (2016). Patient admission planning using approximate dynamic programming. *Flexible services and manufacturing journal*, *28*, 30-61.

[28] **Jeet, V., & Kutanoglu, E.,** (2007). Lagrangian relaxation guided problem space search heuristics for generalized assignment problems. *European Journal of Operational Research*, *182*(3), 1039-1056.

[29] **John, M.,** (1958). Production Planning and Inventory Control. *Nova Iorque: McGraw-Hill*.

[30] **Koç, Ç., Bektaş, T., Jabali, O., & Laporte, G.,** (2016). Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research*, *249*(1), 1-21.

[31] **Kuhn, H., & Sternbeck, M. G.,** (2013). Integrative retail logistics: An exploratory study. *Operations Management Research*, *6*(1), 2-18.

[32] **Kuhn, H., & Sternbeck, M. G.,** (2013). Integrative retail logistics: An exploratory study. *Operations Management Research*, *6*, 2-18.

[33] **Leon, V. J., & Ramamoorthy, B.,** (1997). An adaptable problem-space-based search method for flexible flow line scheduling. *IIE transactions*, *29*(2), 115-125.

[34] **Liu, S., Huang, W., & Ma, H.,** (2009). An effective genetic algorithm for the fleet size and mix vehicle routing problems. *Transportation research part E: logistics and transportation review*, *45*(3), 434-445.

[35] **Naphade, K. S., David Wu, S., & Storer, R. H.,** (1997). Problem space search algorithms for resource-constrained project scheduling. *Annals of operations research*, *70*(0), 307-326.

[36] **Neves-Moreira, F., Da Silva, D. P., Guimarães, L., Amorim, P., & Almada-Lobo, B.,** (2018). The time window assignment vehicle routing problem with product dependent deliveries. *Transportation research part E: logistics and transportation review*, *116*, 163-183.

[37] **Perry, T. C., & Hartman, J. C.,** (2009). An approximate dynamic programming approach to solving a dynamic, stochastic multiple knapsack problem. *International Transactions in Operational Research*, *16*(3), 347-359.

[38] **Pochet, Y., & Wolsey, L. A.,** (2006). *Production planning by mixed integer programming* (Vol. 149). Springer.

[39] **Ronconi, D. P., & Powell, W. B.,** (2010). Minimizing total tardiness in a stochastic single machine scheduling problem using approximate dynamic programming. *Journal of Scheduling*, *13*(6), 597-607.

[40] **Ronen, D., & Goodhart, C.,** (2008). Tactical store delivery planning. *Journal of the operational research society*, *59*(8), 1047-1054.

[41] **Savelsbergh, M.,** (1990). A parallel insertion heuristic for vehicle routing with side constraints. *Statistica Neerlandica*, *44*(3), 139-148.

[42] **Sayarshad, H. R., & Ghoseiri, K.,** (2009). A simulated annealing approach for the multi-periodic rail-car fleet sizing problem. *Computers & Operations Research*, *36*(6), 1789-1799.

[43] **Schöneberg, T., Koberstein, A., & Suhl, L.,** (2010). An optimization model for automated selection of economic and ecologic delivery profiles in area

forwarding based inbound logistics networks. *Flexible services and manufacturing journal*, *22*, 214-235.

[44] **Silva, T. A., & de Souza, M. C.,** (2020). Surgical scheduling under uncertainty by approximate dynamic programming. *Omega*, *95*, 102066.

[45] **Simao, H. P., Day, J., George, A. P., Gifford, T., Nienow, J., & Powell, W. B.,** (2009). An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Science*, *43*(2), 178-197.

[46] **Speranza, M., & Ukovich, W.,** (1996). An algorithm for optimal shipments with given frequencies. *Naval Research Logistics (NRL)*, *43*(5), 655-671.

[47] **Speranza, M. G., & Ukovich, W.,** (1994). Minimizing transportation and inventory costs for several products on a single link. *Operations Research*, *42*(5), 879-894.

[48] **Sternbeck, M. G., & Kuhn, H.,** (2014). An integrative approach to determine store delivery patterns in grocery retailing. *Transportation research part E: logistics and transportation review*, *70*, 205-224.

[49] **Storer, R. H., Flanders, S. W., & David Wu, S.,** (1996). Problem space local search for number partitioning. *Annals of operations research*, *63*, 463-487.

[50] **Storer, R. H., Wu, S. D., & Vaccari, R.,** (1992). New search spaces for sequencing problems with application to job shop scheduling. *Management Science*, *38*(10), 1495-1509.

[51] **Sun, L., Zhang, Y., & Hu, X.,** (2021). Economical-traveling-distance-based fleet composition with fuel costs: An application in petrol distribution. *Transportation research part E: logistics and transportation review*, *147*, 102223.

[52] **Taleizadeh, A. A., Shokr, I., Konstantaras, I., & VafaeiNejad, M.,** (2020). Stock replenishment policies for a vendor-managed inventory in a retailing system. *Journal of Retailing and Consumer Services*, *55*, 102137.

[53] **Tanksale, A., & Jha, J. K.,** (2020). A hybrid fix-and-optimize heuristic for integrated inventory-transportation problem in a multi-region multi-facility supply chain. *RAIRO-Operations Research*, *54*(3), 749-782.

[54] **Topaloglu, H.,** (2005). An approximate dynamic programming approach for a product distribution problem. *IIE transactions*, *37*(8), 697-710.

[55] **Van Donselaar, K. H., Gaur, V., Van Woensel, T., Broekmeulen, R. A., & Fransoo, J. C.,** (2010). Ordering behavior in retail stores and implications for automated replenishment. *Management Science*, *56*(5), 766-784.

[56] **Van Zelst, S., Van Donselaar, K., Van Woensel, T., Broekmeulen, R., & Fransoo, J.,** (2009). Logistics drivers for shelf stacking in grocery retail stores: Potential for efficiency improvement. *International Journal of Production Economics*, *121*(2), 620-632.

[57] **Żak, J., Redmer, A., & Sawicki, P.,** (2011). Multiple objective optimization of the fleet sizing problem for road freight transportation. *Journal of advanced transportation*, *45*(4), 321-347.

PUBLISHMENTS, PATENTS AND PRESENTATIONS ADOPTED FROM THESIS:

- **N. D. Karimi,** K. Ertogral, 2018. Strategic Fleet Sizing Problem in A Vendor Managed Inventory System with Predetermined Delivery Frequencies; A Fix and Optimize Heuristic, IISE Annual Conference, Orlando, USA.
- **Niousha Karimi Dastjerd,** Kadir Ertoğral, Eda Yücel, 2019. Entegre Filo Büyüklüğü Belirleme Ve İkmal Planlama Problemi İçin Yaklaşık Dinamik Programlama Sezgiseli, YAEM, Ankara, Turkey
- **N. D. Karimi**, K. Ertogral**,** 2022. Problem Space Search Heuristics Using Fix and Optimize Approach for The Integrated Fleet Sizing and Replenishment Planning Problem, The 12th Annual International Conference on Industrial Engineering and Operations Management, Istanbul Turkey, March 7-10.
- **D.Aghazade,** K. Ertogral, 2022. An Improved Approximate Dynamic Programming Method for the Integrated Fleet Sizing and Replenishment Planning Problem with Predetermined Delivery Frequencies, 10th IFAC Conference on Manufacturing Modelling, Management and Control, June 22-24, Nantes, France
- **Aghazadeh, D.,** Ertogral, K, 2022. An Improved Approximate Dynamic Programming Method for the Integrated Fleet Sizing and Replenishment Planning Problem with Predetermined Delivery Frequencies, IFAC-PapersOnLine, 55(10), 3034-3039.
- **Duygu Aghazadeh,** Kadir Ertogral, 2023. Problem space search metaheuristics with fix and optimize approach for the integrated fleet sizing and replenishment planning problem. Journal of Industrial and Management Optimization. doi: 10.3934/jimo.2023107