MULTI-COMPARTMENT INVENTORY ROUTING PROBLEM WITH FLEXIBLE PRODUCT TYPES AND FIXED COMPARTMENT SIZES

A Thesis

by

Sena Artarslan

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

Master of Science

in the
Department of Industrial Engineering

Özyeğin University
September 2023

MULTI-COMPARTMENT INVENTORY ROUTING PROBLEM WITH FLEXIBLE PRODUCT TYPES AND FIXED COMPARTMENT SIZES

Approved by:

Professor Ali Ekici, Advisor
Department of Industrial Engineering
Özyeğin University

Professor Okan Örsan Özener
Department of Industrial Engineering
Özyeğin University

Assoc. Professor Ertan Yakıcı
Department of Industrial Engineering
National Defence University

Date Approved: August 3, 2023

To my family...

ABSTRACT

This thesis introduces a *Multi-Compartment Inventory Routing Problem* (MCIRP). The addressed problem aims to minimize the total traveling costs while ensuring customers are not out of stock with multiple products over the given planning time horizon. The distribution is made with a homogenous fleet of vehicles with flexible product types and fixed compartment sizes, where each compartment can accommodate all product types and has a fixed capacity. The supplier manages customer inventory levels by creating a distribution plan respecting the capacity restrictions of the compartments and customers. We examine the application of this problem with liquid products that can be partially delivered to customers with compartments that have debit meters. To address this complex problem, we propose a mathueristic method that combines Adaptive Large Neighborhood Search (ALNS) with mathematical models. The success of the solution method has been demonstrated by comparing it against a lower bound, that is flow formulation from the literature. Our results show that with the generated comprehensive and large-scale instances, our solution algorithm achieves only 18.13% worse solutions than the conservative lower bound.

ÖZETÇE

Bu tez çalışmasında, Çok Bölmeli Envanter Rotalama Problemi, matematiksel modellerle entegre bir sezgisel çözüm yaklaşımı önerilerek çözümlenmektedir. Ele alınan problem, dağıtım ağının toplam seyahat maliyetlerini en aza indirmeyi ve aynı zamanda müşterilerin verilen planlama zaman ufku boyunca birden fazla ürünle stoksuz kalmamasını sağlamayı amaçlamaktadır. Dağıtım, çok kompartmanlı homojen bir araç filosu ile yapılır ve kompartmanlarda taşınan ürünler üzerinde kısıtlama yoktur, aynı zamanda kompartmanlar sabit bölme hacimlerine sahiptir. Tedarikçi, her bir ürünün envanter ve kapasite bilgilerine dayalı bir dağıtım planı oluşturarak müşterilerin envanter seviyelerini yönetir. Bu problemin uygulamasını, debimetreye sahip bölmeleri olan araçlarla müşterilere kısmen teslim edilebilen sıvı ürünlerle incelemekteyiz. Bu şekildeki uygulamalara üreticilerden farklı kalitedeki zeytinyağlarının toplanmasında, benzin istasyonlarına dağıtılan ürünlerde karşılaşmaktayız. Bu karmaşık problemi çözümleyebilmek için, Uyarlanabilir Büyük Komşuluk Arama ile matematiksel modelleri birleştiren bir yöntem öneriyoruz. Çözüm yönteminin başarısı, literatürde akış formülasyonu olan bir alt sınır ile karşılaştırılarak gösterilmiştir. Sonuçlar, oluşturulan kapsamlı ve büyük ölçekli örneklerle, çözüm algoritmamızın alt sınırdan yalnızca %18,13 daha kötü çözümler elde ettiğini göstermektedir.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude and appreciation to my advisor, Prof. Ali Ekici, for his invaluable guidance, support, and valuable insights throughout the research process. His expertise and encouragement have been instrumental in shaping the direction of this thesis.

I am also deeply thankful to Prof. Okan Örsan Özener and Asst. Prof. İhsan Yanıkoğlu for their constructive feedback and thoughtful suggestions throughout my undergraduate and graduate studies. Their academic mentorship has been a source of inspiration and motivation.

Furthermore, I extend my heartfelt thanks to my fellow graduate student colleagues, Berk, Huzeyfe, Taha, and Farzad, who have been a constant source of encouragement, understanding, and support during this challenging journey. Their encouragement and positivity have made this experience more enjoyable and meaningful.

Also, I would like to thank the Artarslan family; Yunus, Fatma, Işıl and my dearest friends Ece, İlayda, and Murat, for their unwavering support and motivation throughout my life. I feel very lucky to have their supportive and caring presence in my life.

Lastly, I am thankful for the financial support provided by the "BİDEB 2210-A Genel Yurtiçi Lisansüstü Burs Programı" and "ARDEB-1001 119M245" during my graduate studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Managing inventory levels and building a distribution network can be challenging for suppliers and customers, particularly in today's complex business environment. One approach to handle this inventory management challenge is Vendor Managed Inventory (VMI), a supply chain management approach where the supplier is responsible for monitoring and managing the customer's inventory levels with the demand and capacity information. VMI is a collaborative approach that benefits the supplier and the customer by reducing the frequency of last-minute orders, saving time and resources, and ultimately reducing costs. It allows the supplier to make informed decisions on distribution routes, customer visit frequency, and product distribution quantities, which can further minimize distribution and inventory holding costs. From the customer's perspective, VMI provides the advantage of leaving inventory management to the supplier, bargaining for more favorable pricing due to the cost advantage created by the supplier, and reducing the possibility of stock-outs due to shared information.

We investigate a version of the vendor-managed inventory routing problem where the supplier is responsible for optimizing the distribution of multiple products to customers over a multi-period planning time horizon. During this planning horizon, customers order different types of products without any predictable pattern, resulting in varying distribution requirements for different products at different times. Moreover, each customer has a storage capacity for each product they order, which the supplier must consider when planning the distribution. To handle these varying requirements, we propose separating vehicles' storage capacity into a specified number of parts

called compartments that can transport different products in a single route. By sharing the vehicle capacity among different products, we can meet customers' fluctuating demands for different products. In this study, we propose a Multi-Compartment Inventory Routing Problem (MCIRP), which aims to develop the least-cost distribution strategy using multi-compartmented vehicles while ensuring that no product runs out of stock at customer locations.

While examining routing problems with compartmentalized vehicles, three different types exist about the features of these compartments: (i) the compartments have a fixed size, and each product is dedicated to a unique compartment, (ii) the sizes of the compartments are flexible, but each product is dedicated to a unique compartment, and (iii) the sizes of the compartments are fixed, but there is no dedicated product for compartments.

In our thesis, we assume that the compartments have fixed sizes and flexible product types, which is the third type. This means the number of compartments and their storage capacity is fixed and the products to put in each compartment are a decision. Not dedicating a product to a compartment gives flexibility while meeting the unstable demand but this decision adds a new dimension to the problem which makes it hard to solve. Also, only one type of product can only be loaded in each compartment at a time, and different products can be transported at different times in one compartment. We may encounter this type of compartment arrangement in real-world examples, such as the collection of olive oils of different qualities from the producer [1], the distribution of different fuel types to gasoline stations [2, 3, 4, 5, 6], and the distribution of bulk liquid products via ships [7, 8]. With the explained compartment structure, we analyze the MCIRP with three different delivery assumptions that are frequently encountered in literature. These are, in a period a customer can be visited (i) only once, (ii) by multiple vehicles for each product (iii) more than once for each product.

In this thesis, we consider the problem of distributing several liquid products to customers who demands these products on a multi-period time horizon. This distribution is done with a homogeneous fleet of vehicles with a certain number of compartments. In [9], different types of multi-compartment routing problems are analyzed and it is seen that if the carried product in the truck is liquid, we have a restriction on whether to have a debit meter on the trucks or not. In our problem, we assume that the compartments are equipped with debit meters, allowing us to distribute the product in a compartment for more than one customer. Thus, we have to determine how much amount of product will be distributed from one compartment to which customers. These decisions make the problem complex to be solved optimally, especially in large instances. To handle this complex problem within a reasonable time frame, we implement mathematical models as part of an Adaptive Large Neighborhood Search (ALNS) framework called matheuristic. Briefly, the metheuristic approach merges mathematical models with metaheuristics and it is widely utilized in solving routing problems [10, 11, 12]. In our proposed algorithm, we systematically destroy, repair and improve our solution to achieve the best possible solution and also we are using mathematical models to check the feasibility of the current situation in any step.

The remainder of this thesis organized as follows. In Chapter 2, we present the literature on multi-compartment routing problems for both single and multi-periods. The definition of our problem and MCIRP model are included in Chapter 3. The developed matheuristic approach and the details of ALNS are presented in Chapter 4. In Chapter 5, we provide the results of computational experiments and report the performances of the proposed solution approach for MCIRP and the benchmark algorithm. Finally, we conclude in Chapter 6.

CHAPTER II

PREVIOUS WORKS

In this chapter, we examine multi-compartment routing problems with both single and multiple time horizons. Although some may exist, the studies that investigate the planning horizon that covers multiple periods of the problem are relatively limited. Despite the multi-period problems, many studies only consider one period [2, 13, 14, 15]. Since there are rare studies that cover multi-period time horizons, we also examine single-period studies in this chapter. First, we will classify the multi-compartment routing problems, then mention about the related routing problems.

2.1 Multi-compartment classifications

In a recent literature review by [9], the Multi-Compartment Vehicle Routing Problem (MCVRP) studies were examined with a focus on compartment structures, delivery assumptions, and solution approaches. It is observed that multi-compartment routing studies can be classified into three groups based on the structure of compartments within a vehicle: (i) compartments with fixed capacities that are dedicated to specific products [16], (ii) compartments with flexible capacities that are dedicated for specific products [17], and (iii) compartments with fixed capacities that are not dedicated to specific products [18].

In this study, we examine the inventory routing problem assuming that compartments are not dedicated to any specific products and they have fixed capacities. Compartments being not dedicated to any specific product can provide suppliers with the necessary flexibility to make distribution plans. However, assigning products to compartments on each route makes the problem more complex as compared to the first

two compartment structures. Incorporating such compartment structure into multi-compartment routing studies yields practical real-world applications. For instance, it facilitates the collection of various quality olive oils from manufacturers and the distribution of diverse fuel types to gas stations and, [19] utilized a multi-compartment routing approach to address the problem of collecting different quality olive oils from the manufacturer, while other researchers have applied different techniques to the distribution of fuel types to gas stations [2, 4, 20].

In liquid product distribution, some compartmentalized vehicles may have debit meters which can help to measure the amount delivered to customers. According to [1], it was presumed that vehicles have debit meters in each compartment, which enable splitting the product in a compartment between different customers. However, [5, 6] have chosen to ignore the presence of debit meters and they assume that when a customer is visited by a vehicle, the product should be discharged entirely for that specific customer. This precludes the possibility of splitting the product among different customers or making partial deliveries. This approach simplifies the problem since it limits the route length to the number of compartments in a vehicle. In our study, we assume that multi-compartmentalized vehicles have debit meters, providing us to distribute the same product to multiple customers while creating a distribution network.

2.2 Multi-compartment routing studies

Next, we review the existing literature on the single-period Multiple Compartment Vehicle Routing Problem (MCVRP) version that assumes compartments have fixed sizes and product types are flexible. [2] investigated the single-period MCVRP and presented a hybrid algorithm combining a genetic algorithm and a local search. The aim was to minimize the number of vehicles used while satisfying the customers' demands and vehicle capacity constraints. [13, 14, 15] focused on an MCVRP version

that maximizes the total profit obtained while distributing between the minimum and maximum amounts demanded by all gas stations. They used a heterogeneous vehicle fleet, each with different transportation costs, and presented a hybrid algorithm combining ant colony optimization and local search. In these papers, authors put an upper bound to the total customers to be visited in each route. [21] assumed that in vehicles, debit meters exist while solving MCVRP, allowing products in a compartment to be distributed among different customers. They allowed the demands to be distributed by allowing a customer to receive deliveries from multiple vehicles within a period. They developed an adaptable large neighborhood search algorithm to solve the problem and proved its efficiency. [22] utilized multi-compartment vehicles for collecting and transporting raw milk from farms. The aim was to minimize fuel consumption and vehicle cleaning costs by creating the shortest routes and minimizing the number of compartments used and the waiting time at the loading area. They developed a heuristic algorithm combining tabu search and variable neighborhood search. The literature on the MCVRP is extensive and each has different assumptions. These studies highlight the different aspects and applications of the multi-compartment routing problems, but these only consider one period which gives a simpler aspect to the existing problem. However, the MCIRP has limited literature compared to the MCVRP.

Next, we explore several versions of the MCIRP studied by different researchers. [23] analyzed one version of the MCIRP that focused on transporting livestock from farms to slaughterhouses using multi-compartment vehicles. To solve this problem, they developed an exact solution method based on column generation, assuming a heterogeneous fleet. This method allowed for only one visit per period for a specific type of animal per farm. The animals can only stay at the slaughterhouse for one night, thus the inventory is limited. Due to the small number of feasible routes resulting from the limited number of vehicle compartments, this version of the problem

is solved exactly for a three-day planning horizon.

[5] studied an MCIRP version that assumes distribution vehicles with homogeneous capacities and only distributes the fully loaded compartments. They allowed distribution to a maximum of three customers in a route, assuming that each customer's demand is constant. They solved the problem using the Variable Neighborhood Search method. [6] investigated a similar problem to [5], but outside of distribution fleets. They used a Variable Neighborhood Descent algorithm to study vehicles with two or four compartments of equal capacity. In both of these studies, the compartments have fixed sizes and any product is not dedicated to them, also they are not equipped with debit meters. Here, several restrictions make the problem simpler in contrast to our problem. In [1], the authors developed a solution method for a problem involving collecting olive oils of different grades, which requires a cleaning process before loading certain oils. They presented a branch-and-cut-based exact solution method for the problem. In this problem, the compartments are equipped with debit meters, which is the same assumption as our problem. However, the supplier has no control over the customers' inventory levels, differentiating this problem from ours.

Finally, [3] examined the MCIRP problem across various split delivery scenarios involving a heterogeneous fleet. They developed an exact solution method that utilizes the branch-and-cut method but it only gives optimal solutions for small data sets. Their numerical tests were performed on limited data sets with homogeneous fleets and compartments of equal capacity. This study considers fixed-size of compartments and also both flexible and fixed product types. Although their solution approach is successful, their approach is only tested in small data sets. The structure of compartments and the existence of debit meters make this problem similar to our problem.

In summary, the literature review has emphasized the need for research in the

field of MCIRP with flexible product types and fixed compartment sizes, particularly in larger data sets. To tackle this complex and large-scale problem with large data sets, heuristics have been considered as a viable solution. Lately, metaheuristics have been widely utilized to solve multi-compartment routing problems [5, 6], with ALNS being identified as a promising approach due to its ability to search different neighborhood structures. Moreover, an integrated version of ALNS with mathematical models, known as matheuristic, has been found to be effective in addressing various optimization problems, including routing [10, 24, 11]. To obtain good solutions on large datasets, we developed a mathueristic for our problem and we are testing it under three different delivery assumptions.

Although different versions of MCIRP have been studied in the literature, researchers have made restrictive assumptions about the problem and/or obtained solutions based on very small data sets during the testing developed methods. This has resulted in a gap in the literature when it comes to solving MCIRP with fixed compartment sizes and flexible product types. To address this gap in the literature and to obtain good solutions on large datasets, we developed a metheuristic method for our problem to get good solutions and test it under three different assumptions observed in the literature. These are, in a period a customer can be visited;

(i) only once [25, 18],

(ii) by multiple vehicles for each product [26, 27]

(iii) more than once for each product [3]

CHAPTER III

PROBLEM DEFINITION

This section provides a formal definition of the Multi-Compartment Inventory Routing Problem with flexible product types and fixed compartment sizes. We have identified the problem as a complex and mixed integer program that is challenging to solve optimally due to its large scale. However, we find it essential to define the problem precisely to eliminate any ambiguity in its interpretation.

In the studied version of MCIRP, there are n numbers of customers and m different product types, customers place orders throughout the T number of periods. These demands are met by only one vendor, which considers customers' inventory levels.

We define MCIRP with an undirected graph $G = (V_0, E)$, where V_0 is the vertex set and E is the edge set. The set $V_0 = \{0, 1, 2, \ldots, n\}$ contains index of the locations of the depot $\{0\}$ and the customers $V = \{1, 2, \ldots, n\}$, E is the set of edges (i, j) that connects V_0 nodes. A traveling cost b_{ij} is known for each edge (i, j). There are p different products that customers place order for T different planning horizons. The product set is defined as $M = \{1, 2, \ldots, p\}$ where the time periods is $\mathcal{T} = \{1, 2, \ldots, T\}$. Customer i demands d_{im}^t amount of product m at time t and each customer i has SC_{im} capacity for product m and has an initial inventory level of $\mathcal{I}_{im}^0$ from product m. The distribution from the central warehouse is carried out with a homogeneous vehicle fleet with compartment structures capable of transporting any product in any compartment. Each vehicle k in vehicle set $\mathcal{K} = \{1, 2, \ldots, K\}$ has C different compartments and for each vehicle the capacity of compartment $c \in \mathcal{C} = \{1, 2, \ldots, C\}$ is denoted with Q_c. Maximum tour length for each route is denoted with L.

We examine this problem under three different assumptions regarding the distribution limitations to customers. These assumptions are (i) only one vehicle can make a delivery to a customer in a period, (ii) one vehicle can make a delivery to a customer in a period for a single product, but different products can be delivered by different vehicles, and (iii) one customer can receive deliveries from multiple vehicles in a period. We adjust our solution approach for the different assumptions discussed above.

The MCIRP with mixed-integer programming can be formulated with these decision variables:

q_{imtk} = Amount of product m carried to customer i at time t using vehicle k

$$m \in M, t \in \mathcal{T}, i \in V, k \in \mathcal{K}$$

I^t_{im} = Amount of product $m$in the inventory for customer i at time period t

$$m \in M, i \in V, t \in \mathcal{T}$$

$$f_{cmtk} = \begin{cases} 1, & \text{if product } m \text{ carried in compartment } c \text{ to customer } i \text{ at time } t \\ 0, & \text{otherwise} \end{cases} \qquad c \in \mathcal{C}, m \in M, t \in \mathcal{T}, k \in \mathcal{K}$$

$$z_{itk} = \begin{cases} 1, & \text{if customer } i \text{ is visited by vehicle } k \text{ at time period } t \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, t \in \mathcal{T}, k \in \mathcal{K}$$

$$x_{ijtk} = \begin{cases} 1, & \text{If vehicle } k \text{ moves from node } i \text{ to node } j \text{ at time period } t \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, j \in V, t \in \mathcal{T}, k \in \mathcal{K}$$

$$y_{imtk} = \begin{cases} 1, & \text{if product } m \text{ is delivered to customer } i \text{ on day } t \text{ by vehicle } k \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, m \in M, t \in \mathcal{T}, k \in \mathcal{K}$$

Below, we represent the model with the first delivery assumption that considers a customer can be visited once by a vehicle in a period

Main:

$$\text{Min} \sum_{t\in T}\sum_{k\in K}\sum_{i\in V_0}\sum_{j\in V_0} b_{ij}x_{ijtk} \tag{1}$$

$$st \quad \mathcal{I}^0_{im} + \sum_{k\in\mathcal{K}} q_{im1k} = I^1_{im} + d^1_{im} \qquad \forall i \in V, m \in M \tag{2}$$

$$I^{t-1}_{im} + \sum_{k\in\mathcal{K}} q_{imtk} = I^t_{im} + d^t_{im} \qquad \forall i \in V, m \in M, t \in \mathcal{T}\backslash\{1\} \tag{3}$$

$$I^{t-1}_{im} + \sum_{k\in\mathcal{K}} q_{imtk} \le SC_{im} \qquad \forall i \in V, m \in M, t \in \mathcal{T}\backslash\{1\} \tag{4}$$

$$\mathcal{I}^0_{im} + \sum_{k\in\mathcal{K}} q_{im1k} \le SC_{im} \qquad \forall i \in V, m \in M \tag{5}$$

$$\sum_{i\in V} q_{imtk} \le \sum_{c\in\mathcal{C}} Q_c f_{cmtk} \qquad \forall m \in M, t \in \mathcal{T}, k \in K \tag{6}$$

$$\sum_{i\in V_0}\sum_{j\in V_0} b_{ij}x_{ijtk} \le L \qquad \forall t \in \mathcal{T}, k \in \mathcal{K} \tag{7}$$

$$u_i - u_j + n\sum_{k\in\mathcal{K}} x_{ijtk} \le n-1 \qquad \forall i, j \in V, t \in \mathcal{T} \tag{8}$$

$$q_{imtk} \le SC_{im} y_{imtk} \qquad \forall i \in V, m \in M, t \in \mathcal{T}, k \in \mathcal{K} \tag{9}$$

$$y_{imtk} \le \sum_{j\in V_0} x_{jitk} \qquad \forall i \in V, m \in M, t \in \mathcal{T}, k \in \mathcal{K} \tag{10}$$

$$\sum_{i\in V_0} x_{ijtk} = \sum_{i\in V_0} x_{jitk} \qquad \forall j \in V_0, t \in T, k \in \mathcal{K} \tag{11}$$

$$\sum_{k\in\mathcal{K}} y_{imtk} \le 1 \qquad \forall i \in V, m \in M, t \in \mathcal{T} \tag{12}$$

$$q_{imtk} \le SC_{im} z_{itk} \qquad \forall i \in V, m \in M, t \in \mathcal{T}, k \in \mathcal{K} \tag{13}$$

$$\sum_{k\in\mathcal{K}} z_{itk} \le 1 \qquad \forall i \in V, t \in \mathcal{T} \tag{14}$$

$$\sum_{j\in V_0|j\neq i} (x_{ijtk} + x_{jitk}) \le 2z_{itk} \qquad \forall i \in V, t \in \mathcal{T}, k \in \mathcal{K} \tag{15}$$

$$\sum_{m\in M} f_{hmtk} \le 1 \qquad \forall c \in \mathcal{C}, t \in \mathcal{T}, k \in \mathcal{K} \tag{16}$$

$$z_{itk} \in \{0,1\} \qquad \forall i \in V, t \in \mathcal{T}, k \in \mathcal{K} \tag{17}$$

$$x_{ijtk} \in \{0,1\} \qquad \forall i,j \in V_0, t \in \mathcal{T}, k \in \mathcal{K} \tag{18}$$

$$y_{imtk} \in \{0,1\} \qquad \forall i \in V, m \in M, t \in \mathcal{T}, k \in \mathcal{K} \tag{19}$$

$$f_{cmtk} \in \{0,1\} \qquad \forall c \in \mathcal{C}, m \in M, t \in \mathcal{T}, k \in \mathcal{K} \tag{20}$$

$$q_{imtk} \geq 0 \qquad \forall i \in V, m \in M, t \in \mathcal{T}, k \in \mathcal{K} \tag{21}$$

$$I_{imt} \geq 0 \qquad \forall i \in V, m \in M, t \in \mathcal{T} \tag{22}$$

In this model, the objective (1) is to minimize the total traveling cost. Constraints (2 - 3) for inventory balance. Customers' inventory levels are controlled with constraints (4 - 5). Constraints (6) ensure a vehicle can deliver only the loaded amount in any route. Constraints (7) restrict the total distance traveled by a vehicle as L. Constraints (8) for subtour elimination. (9 - 10) decides to visit the customer for which products. Constraints (11) ensure that if a node is visited by a vehicle, the vehicle should leave it. Constraints (12) ensures the first and second assumptions. (13 - 15) ensures first assumpption. Constraints (16) ensures only one product can be put in each compartment. The remaining constraints for integrality.

As we discussed, to use this model for the second assumption where different vehicles are allowed to distribute different products to a customer in one period, we have to remove constraints (13 - 15) from the model used in the first assumption. In the third assumption, where a product is allowed to be distributed by different vehicles on the same period to a customer, constraints (12 - 15) should be removed from the model.

Example of the problem:

In order to understand the problem better, we give an example of a distribution network with 2 customers, 2 products and 2 time periods. The demanded units of products for each customer and period are represented in Table 1.

Table 1: Demands of the Example Instance

d_{imt}			
i	m	t	**value**
1	1	1	4
1	2	1	0
2	1	1	3
2	2	1	3
1	1	2	5
1	2	2	4
2	1	2	2
2	2	2	1

In Figure 1, the network is represented at time period 0. Here, the distances between the nodes are placed in the arcs. Some squares are represented near the customer nodes, which help us to see each customer's inventory levels for both products. The green one is for the product 1, blue one is for the product 2. As an initial inventory, customer 1 has 2 units for each product. And customer 2 has 3 unit for product 1 and 2 units for product 2.

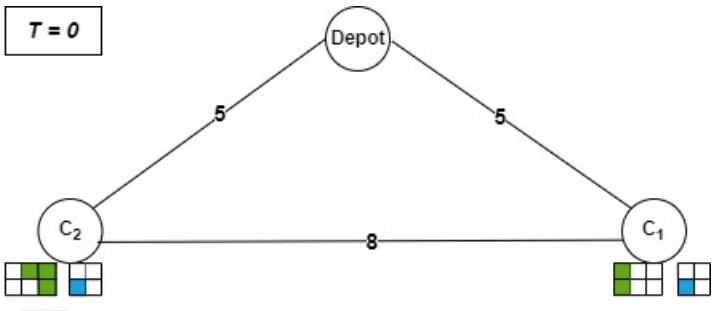

Figure 1: Distribution network of the example instance at time period 0

At the beginning of period 1, to satisfy the customer's demanded units of both products, a fully loaded truck leaves the depot, which is represented in Figure 2. It

drops 4 units of product 1 and 1 unit of product 2 to customer 1. The same truck drops 2 and 3 units of products 1 and 2 to customer 2, respectively. Then, the truck returns to the depot. After the deliveries are made and demanded amounts leave the customer, the left inventory levels can be seen in the images near the customer nodes. The total travel cost of this route *(depot - customer 1 - customer 2 - depot)* is 18.

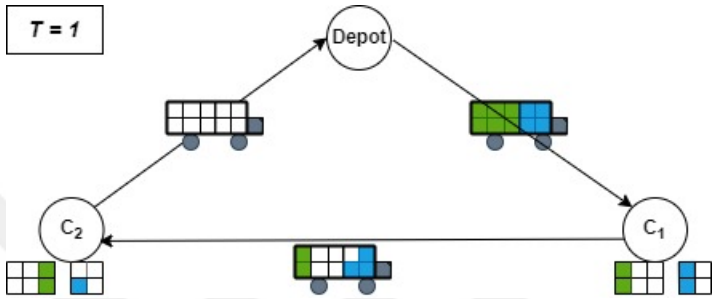

Figure 2: Distribution network of the example instance at time period 1

For period 2, to meet the demand, a truck leaves the depot only to visit customer 1 and it is represented in Figure 3. Customer 2's demanded amounts can be satisfied with the inventory levels. The cost of this route *(depot - customer 1 - depot)* is 10.

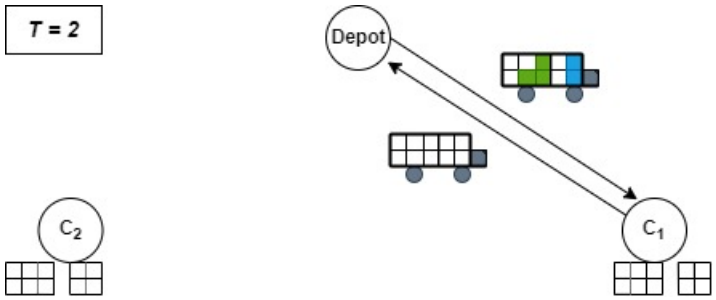

Figure 3: Distribution network of the example instance at time period 2

In this example, the total traveling cost is 26. If the trucks we are utilizing were not compartmentalized, then we would have to visit customer 2 twice in the second period, leading to a higher cost. Again, if the problem wasn't a vendor-managed inventory, then we wouldn't know customer 1's inventory levels, and we wouldn't distribute the product in period 1 wisely, which would lead us to visit customer 1 in period 2 and make the total cost higher.

CHAPTER IV

SOLUTION METHODOLOGY

To develop a good solution method for the introduced problem in Chapter 3, we propose a matheuristic that integrates mathematical models with Adaptive Large neighbourhood Search (ALNS). ALNS is a metaheuristic algorithm for solving optimization problems and it is first introduced by [28] as an extension of the Large neighbourhood Search (LNS) [29] by allowing the use of multiple destroy and repair methods within the same search process. neighbourhood structures are adjusted dynamically based on neighbourhoods' performance, making the heuristic approach adaptive. This enhances the search space and saves from getting stuck in local optima. The ALNS heuristic is used in the inventory routing problems [11, 24, 30] and its quality to find good solutions is proven.

As discussed above, the current solution is systematically being destroyed and repaired after finding an initial solution. In the following sections, we first mention how to find an initial solution and then explain the matheuristic algorithm used to solve MCIRP.

4.1 Initial Solution for MCIRP

We are applying a route-based mathematical model to construct an initial feasible solution to use this as an input to our solution approach. This model is a simplified version of the model used in equations (1 - 22). In this model, a predetermined set of routes is created and only those routes are used in the model. The model selects which routes to utilize in each period from this set. However, if we could provide this model with all possible routes, which is $2^{|V|}$ different routes, we would be able to obtain the optimal solution. Unfortunately, due to the limited number of routes, we

cannot guarantee that the optimal solution can be obtained from the output of the route-based model.

We now provide the route-based model to find an initial solution. From all possible $2^{|V|}$ routes, a certain number of them are chosen randomly and the set of selected routes forms the set $\mathcal{R}$. We calculate these routes' visiting sequences with using Travelling Salesmen Problem (TSP), since we know which customers will be visited on each route. To solve the TSP, we use the algorithm as applied in the literature [31] and after that we know the length of each route which is denoted as b_r. R^i is the set of routes that customer $i \in V$ visits. If customer i is visited in route r, then α_{ir} is 1.

Same as the main model, the inventory variables I^t_{im} give the amount of product m in the inventory at customer i, in period t; x^t_r is a new binary variable determines whether route r is used in period t. The variable q^t_{imr} represents the quantity of product m delivered to customer i via route r in period t. Finally, the variable e^t_{cmr} is a binary variable that equals 1 if compartment c in route r carries product m at time t.

The following model is developed as route-based mathematical model:

MIP-RB:

$$\text{Min} \quad \sum_{r\in R}\sum_{t\in T} b_r x_r^t \tag{23}$$

$$st \quad \sum_{r\in R_i} x_r^t \leq 1 \qquad \forall i \in V, \forall t \in T \tag{24}$$

$$I_{im}^{t-1} + \sum_{r\in R} q_{imr}^t = I_{im}^t + d_{im}^t \qquad \forall i \in V, m \in M, t \in \mathcal{T}\backslash\{1\} \tag{25}$$

$$\mathcal{I}_{im}^0 + \sum_{r\in R} q_{imr}^1 = I_{im}^1 + d_{im}^1 \qquad \forall i \in V, m \in M \tag{26}$$

$$I_{im}^{t-1} + \sum_{r\in R} q_{imr}^t \leq SC_{im} \qquad \forall i \in V, m \in M, t \in \mathcal{T}\backslash\{1\} \tag{27}$$

$$\mathcal{I}_{im}^0 + \sum_{r\in R} q_{imr}^1 \leq SC_{im} \qquad \forall i \in V, m \in M \tag{28}$$

$$q_{imr}^t \leq \sum_{c\in\mathcal{C}} Q_c e_{cmr}^t \qquad \forall i \in V, m \in M, r \in R, t \in \mathcal{T} \tag{29}$$

$$\sum_{i\in V}\sum_{m\in M}\sum_{t\in\mathcal{T}} q_{imr}^t (1-\alpha_{ir}) = 0 \qquad \forall r \in R \tag{30}$$

$$\sum_{m\in M} e_{cmr}^t \leq 1 \qquad \forall c \in \mathcal{C}, r \in R, t \in \mathcal{T} \tag{31}$$

$$e_{cmr}^t \leq x_r^t \qquad \forall i \in V, c \in \mathcal{C}, m \in M, r \in R, t \in \mathcal{T} \tag{32}$$

$$q_{imr}^t \geq 0 \qquad \forall i \in V, t \in \mathcal{T}, m \in M, r \in R \tag{33}$$

$$I_{im}^t \geq 0 \qquad \forall i \in V, t \in \mathcal{T}, m \in M \tag{34}$$

$$e_{cmr}^t \in \{0,1\} \qquad \forall i \in V, c \in \mathcal{C}, m \in M, r \in R, t \in \mathcal{T} \tag{35}$$

$$x_r^t \in \{0,1\} \qquad \forall r \in R, t \in \mathcal{T} \tag{36}$$

The objective function (23) targets to minimize the total travelling costs of used routes. Constraints (25 - 26) for inventory balance and constraints (27 - 28) controls the capacity. Compartment capacity is controlled with constraints (29). Delivery to a customer can be made if that customer is visited in that route with constraints (30). At most one type of product can be carried in any compartment is guaranteed with

(31). Constraints (32) ensure that a compartment is used in a vehicle if that route is used in a period. The model described above applies to the first assumption, where a customer can only be visited once per period. However, to account for the second assumption, which allows for delivery by different vehicles for different products to a customer within a day, it is necessary to define a new variable y^t_{imr} in the model.

$$y^t_{imr} = \begin{cases} 1, & \text{if product } m \text{ is delivered to customer } i \text{ in route } r \text{ at time } t \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, m \in M, r \in R, t \in \mathcal{T}$$

Then, constraint (24) needs to be deleted and the following constraints (37 - 40) should be added to the model and these ensure the second assumption's needs, a customer can be visited for a product at most once.

$$y^t_{imr} \leq x^t_r \qquad \forall i \in V, c \in \mathcal{C}, m \in M, r \in R, t \in \mathcal{T} \tag{37}$$

$$\sum_{c \in \mathcal{C}} q^t_{imr} \leq \sum_{c \in \mathcal{C}} Q_c y^t_{imr} \qquad \forall i \in V, m \in M, r \in R, t \in \mathcal{T} \tag{38}$$

$$\sum_{r \in R} y^t_{imr} \leq 1 \qquad \forall i \in V, m \in M, t \in \mathcal{T} \tag{39}$$

$$y^t_{imr} \in \{0, 1\} \qquad \forall i \in V, m \in M, r \in R, t \in \mathcal{T} \tag{40}$$

To use the route-based model for the third assumption, we have to delete constraints (24).

4.2 A Matheuristic for the MCIRP

In our matheuristic solution approach, we employ Adaptive Large neighbourhood Search with integrated mathematical models. Our metheuristic uses the solution derived from route-based model (23 - 40) as an initial solution. After that, ALNS algorithm starts.

ALNS is a metaheuristic that utilizes various destroy, repair, and improve neighbourhoods in a search process. The probability of a destroy neighbourhood being chosen is influenced by its assigned weight with the roulette-wheel mechanism. The weights are updated after every segment based on its performance in the search process. While some ALNS applications utilize a similar method for selecting repair/improvement neighbourhoods, some follow the approach of previous studies [11] and [32], where these neighbourhoods are applied in a predetermined order during each iteration.

We first start with defining the mathematical models that are used in our matheuristic approach, then we continue with defining destroy, repair, improvement neighbourhoods and finish with the algorithm structure.

4.2.1 Used Mathematical Models

In our metheuiristic solution approach, two different mathematical models are used within the algorithm. These models are used after destroy, and in repair and improve neighbourhoods to check the feasibility, get the stock-out amounts for each customer, and determine how much product is delivered to customers by which routes. One of these models is a linear program that minimizes the total stock-out amount of the current solution. The other is a mixed-integer program again with the same objective.

The first model which is linear program is called as stock-out model (SO-LP). The used stock-out model is as follows and it is used for all of the delivery assumptions.

The decision variables and parameters are same as in the RB-MIP (23 - 40) model. Additionally, decision variable S_{im}^t is added to the model and it gives how much the customer i is out of stock for product m at time t.

SO-LP:

$$\text{Min} \sum_{i\in V}\sum_{m\in M}\sum_{t\in T} S_{im}^t(\mathcal{T}-t+1) \tag{41}$$

$$st. \quad I_{im}^{t-1} + \sum_{r\in R} q_{imr}^t \leq SC_{im} \qquad \forall i\in V, m\in M, t\in \mathcal{T}\backslash\{1\} \tag{42}$$

$$\mathcal{I}_{im}^0 + \sum_{r\in R} q_{imr}^1 \leq SC_{im} \qquad \forall i\in V, m\in M \tag{43}$$

$$I_{im}^{t-1} + \sum_{r\in R} q_{imr}^t = d_{im}^t - S_{im}^t + I_{im}^t \qquad \forall i\in V, m\in M, t\in \mathcal{T}\backslash\{1\} \tag{44}$$

$$\mathcal{I}_{im}^0 + \sum_{r\in R} q_{imr}^1 = d_{im}^1 - S_{im}^1 + I_{im}^1 \qquad \forall i\in V, m\in M \tag{45}$$

$$\sum_{i\in V} q_{imr}^t \leq \sum_{c\in\mathcal{C}} Q_c e_{cmr}^t \qquad \forall m\in M, r\in R, t\in\mathcal{T} \tag{46}$$

$$\sum_{i\in V}\sum_{m\in M}\sum_{t\in\mathcal{T}} q_{imr}^t(1-\alpha_{ir}) = 0 \qquad \forall r\in R \tag{47}$$

$$S_{im}^t \geq 0 \qquad \forall i\in V, t\in\mathcal{T}, m\in M \tag{48}$$

$$q_{imr}^t \geq 0 \qquad \forall i\in V, t\in\mathcal{T}, m\in M, r\in R \tag{49}$$

$$I_{im}^t \geq 0 \qquad \forall i\in V, t\in\mathcal{T}, m\in M \tag{50}$$

The presented model in (41 - 50) aims to minimize the total amount of stock-out products by giving higher weight to the last days' stock-outs. In this model, e_{cmr}^t is used as a parameter that comes from the current solution. Constraints (44 - 47) have the same purposes as in the model (25 - 30).

The other model is a mixed-integer program with the same purpose as SO-LP and its purpose is to minimize the total stock-out amount and it is called MIP-SO. In some of the situations, we come up with a newly generated route set R' and we need to determine which product will be carried in those vehicle's compartments. To

decide that, we change the parameter e^t_{cmr} as a decision variable like in the MIP-RB for R'. Here, we use the x_{rt} as a parameter that comes from the current solution which determines whether route r is used in period t.

Here are the constraints added to the LP-SO model to create MIP-SO model:

R' : the set of routes that are newly generated

$$\sum_{m \in M} e^t_{cmr} \leq 1 \qquad \forall c \in \mathcal{C}, r \in R', t \in \mathcal{T} \tag{51}$$

$$e^t_{cmr} \leq x^t_r \qquad \forall c \in \mathcal{C}, m \in M, r \in R', t \in \mathcal{T} \tag{52}$$

$$e^t_{cmr} \in \{0, 1\} \qquad \forall c \in \mathcal{C}, m \in M, r \in R', t \in \mathcal{T} \tag{53}$$

Constraints (51) makes sure that in each compartment only one product can be carried and (52) ensures if route r is visited in period t, then the related e^t_{cmr} must be 1.

The above MIP-SO is generated for the first and third assumptions. To use this model in the second assumption, the following variables and constraints should be added to the model:

$$z^t_{imr} = \begin{cases} 1, & \text{if product } m \text{ is delivered to customer } i \text{ in route } r \text{ at time } t \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, m \in M, r \in R, t \in \mathcal{T}$$

$$q^t_{imr} \leq \sum_{c \in \mathcal{C}} Q_c z^t_{imr} \qquad \forall i \in V, m \in M, r \in R, t \in \mathcal{T} \tag{54}$$

$$\sum_{r' \in R \setminus \{r\}} q^t_{imr'} \leq \sum_{c \in \mathcal{C}} Q_c (1 - z^t_{imr}) \qquad \forall i \in V, m \in M, r \in R, t \in \mathcal{T} \tag{55}$$

$$z^t_{imr} \in \{0, 1\} \qquad \forall i \in V, m \in M, r \in R, t \in \mathcal{T} \tag{56}$$

Constraints (54-56) consider the second assumption's needs.

4.2.2 Neighbourhoods

4.2.2.1 Destroy Neighbourhoods

We employ 18 different neighbourhoods that used to destroy the solution to search different solution spaces. These destroy neighbourhoods are listed below.

(i) exclude β visits: Randomly a period, route and customer are selected. Then, this customer is removed from that random tour and period. This process is repeated β times.

(ii) *insert β visits*: A random period is chosen from the solution. Then, a random customer is chosen from the set of unvisited customers during that period. The selected customer is added to a randomly chosen route so that the cost increase is minimized. This step is repeated β times.

(iii) *exclude the weakest β visits*: The customer that will decrease the total travelling cost mostly by being removed from its own tour will be removed from that tour. This process is repeated β times.

(iv) *insert best β visits*: An insertion is made with the customer that will increase the total travelling cost the least while inserting it to the solution is added to the relevant tour. This process is repeated β times.

(v) *remove all β customers*: A random customer is chosen. The selected customer is excluded from all periods. This process is repeated β times.

(vi) *insert all β customers*: A random customer is chosen. The selected customer is added to the route that results in the lowest cost increase for all days/periods where they have not been visited before. This process is repeated β times.

(vii) *empty β periods*: A random period is chosen from the solution and all routes in that period are removed from the solution.

(viii) *swap routes*: Two periods are randomly selected from the solution. All of the routes in the selected periods are exchanged with each other.

(ix) *move β visits*: A customer is selected randomly from a route during a random period. Then the selected customer is removed from the current route and added to the route that will result in the slightest increase in cost compared to the other time routes and periods. This process is repeated β times.

(x) *remove β routes*: β different routes are removed randomly.

(xi) *insert β routes*: This neighbourhood works oppositely to the above (x). Random β routes are added to random periods.

(xii) *break routes*: β rounds are randomly selected and randomly divided into two.

(xiii) *merge 2β routes*: Randomly 2β routes are selected. Then, these routes are combined with solving TSP.

(xiv) *shaw removals*: This neighbourhood's logic is generated in [33]. Randomly, a customer is selected on a random route in random period. The distance to the selected customer is calculated from the customer who is in the same tour and closest to selected customer. With centering the selected customer, all customers within covered an area equal to twice the diameter of that distance. are excluded.

(xv) *shaw insertions*: Same as the (xiv), To begin, a random route and period are chosen, along with a customer that has not been visited by any vehicle during that period. The process begins by calculating the distance between the chosen customer and its closest neighbor. Then, the chosen customer takes on the role of the center, and all remaining customers, not yet visited, are assessed to see if they lie within twice the previously calculated distance from this center. Those

qualifying customers are then added to the selected route, resulting in the least possible increase in overall cost.

(xvi) *remove β compartments*: In this neighbourhood, randomly β compartment is chosen and emptied.

(xvii) *swap 2β compartments*: 2β compartments will be randomly selected, and the products they contain will be swapped.

(xviii) *change β compartments*: After randomly choosing a compartment, the carried product will be changed with another product. This process is repeated β times.

β is a random number between [1,3]. After applying a destroy neighbourhood to a solution, we check its feasibility and calculate the stock-out amounts for each customer with the LP-SO model.

4.2.2.2 Repair Neighbourhoods

After implementing a destroy neighbourhood to the current solution and checking its feasibility with the LP-SO model, if the solution is infeasible, we fix it sequentially with two different repair neighbourhoods. The first repair neighbourhood is (i)β *minimum ratio insertion*; after applying the first repair neighbourhood, if the solution is still infeasible, we use the second repair neighbourhood (ii) *Break routes*.

The β *minimum ratio insertion* neighbourhood tries to insert customers with stock-out products into routes/periods until a feasible solution is found or no insertion is possible and these insertions are made so that the total stock-out amount decreases with the least cost increase. We try to insert β different customers in an iteration manually and at the end of each iteration, LP-SO calculates the new deliveries and stock-out amounts of the β customers. After each iteration, the neighbourhood restarts until the stopping condition is achieved.

Before implementing this neighbourhood, we choose β customers to be added to routes in an iteration and sort the customers based on their total out-of-stock quantities. We choose the β customers with the highest out-of-stock quantity, if the total number of customers that are out of stock is less than β, then we set β as all of the customers. Then, to determine where to insert each of these β customers, we calculate the ratio of increasing cost to decreasing amount of out-of-stock products for each possible insertion scenario for each customer. Then, the selected customer is inserted into the route in the scenario with the minimum ratio. This insertion is repeated for all β customers in an iteration.

To calculate the ratio, we need to determine the extent of the cost increase and the reduction in out-of-stock products resulting from the addition of a customer to a potential route. The cost is calculated by adding the customer to the route and recalculating the overall route cost and the decrease in out-of-stock products is estimated through an approximate calculation. Considering the restriction of the assumptions, we determine whether a stock-out product can be transported on a route or during a specific period. Then, we calculate the available space for each stock-out product in the considered vehicle. Finally, we can estimate how much the out-of-stock products will decrease by adding a customer to a possible route.

These calculations are made for each possible insertion scenario. Before trying to insert any customer to any route, we make a list of possible insertion scenarios. To find these scenarios for each customer, we first determine the periods that the chosen customer can be added. For example, a customer can only be visited once in each period under the first assumption. If a customer has already been visited in a period, we remove that period and its routes from the list of possible insertion scenarios. After finding possible insertion periods, we examine routes in those periods to find where an insertion can be made. We also consider adding the customer to a new route as direct delivery. If the total capacity is reached in any route, we eliminate

that route from the list.

After generating all possible insertion scenarios, we select the scenario with the minimum ratio for each of the β customers. We repeat this process for all β customers to make β different insertions and ensure a feasible solution. After these insertions, we call SO-LP to determine how much product is delivered to customers by which routes and the remaining stock-out amounts for each customer. After applying the β *minimum ratio insertion* neighbourhood, if customers are still out of stock, we repeat the process until all customers are no longer out of stock or until no further insertions are possible.

While considering the restrictions of the assumptions, for the third one, we do not have any limitation on inserting customers into periods or routes. Under the first assumption, we don't consider any insertion to the periods that the customers are visited as explained in the above example. Under the second assumption, we avoid adding a customer to a period if that customer has already visited more than the number of products available. This is because we limit each customer to being visited at most once for each product during a specific period.

The second repair neighbourhood *Break routes* is used if we can't make the solution feasible by adding the out-of-stock customer to any route. We break the route where the out-of-stock customer is located to reduce the amount of stock out and create two different routes. One of these routes makes a direct delivery to the out-of-stock customer, which helps deliver all stock-out products to that customer. For example, if customer A has an out-of-stock quantity for the first product in the second period, and we cannot add the same customer to any route in the first two periods because of the restrictions, then the second repair neighbourhood steps in and breaks the customer's route on the first or second period. Thus, one of the two broken routes makes a direct delivery to the out-of-stock customer and delivers the out-of-stock quantity with the new route.

Like the first neighbourhood, we look at the ratio of increasing cost to decreasing out-of-stock products to decide which route to break in the solution. Then, possible breaking scenarios' ratios are listed, and each scenario's ratio is calculated. To calculate the decreased amount of product, we use the MIP-SO model defined in (44-56), after breaking the selected route in the scenario.

To apply the second repair, out-of-stock customers are first ranked from largest to smallest according to their total out-of-stock quantities, similar to the first step of the repair. Then, after choosing the first customer in the list, the routes that can be broken in the periods where the customer is being visited are listed. For all of these scenarios breaking procedure is tried and if a feasible solution can be obtained by testing this solution with MIP-SO (44-56), this stage is terminated. However, if the out-of-stock quantity cannot be completely eliminated, the ratio of the increase in cost to the decrease in out-of-stock among different route-breaking alternatives is calculated, and the lowest one among the ratios is selected, and that breaking operation is applied. As explained above, if zero stock out is reached with the MIP-SO model, the process stops in any step.

These two repair neighbourhoods are sufficient to make an infeasible solution feasible under the first and third assumptions. However, it has been observed that adding a constraint to LP-SO that satisfies the second assumption's restriction would disrupt linear programming which costs time in the algorithm. In the second assumption's LP-SO a product may be delivered to a customer with multiple routes within a period. Consequently, while solving LP-SO, under the second assumption, there may be feasible scenarios that actually violate feasibility. To address this issue, we propose a repair heuristic method to be employed in the second assumption.

In the repair heuristic method, scenarios that violate the second assumption are checked when the out-of-stock quantity is zero at the end of the LP-SO. If a feasibility-violating scenario occurs, all deliveries of the relevant product going to the relevant

customer on the same period are first attempted to be combined in a single vehicle. To do this, it is attempted firstly by not assigning a compartment if possible, then it is tried by assigning an empty compartment to the corresponding product if available. If these scenarios cannot be implemented due to vehicle and compartment constraints, direct delivery to the relevant customer is generated as a last option, and the relevant compartment assignments are made without violating the second assumption.

4.2.2.3 Improvement Neighbourhoods

After destroy and repair neighbourhoods, a feasible solution comes to improve neighbourhoods which are (i) *Remove route neighbourhood*, (ii) *Remove customers neighbourhood* to decrease the total traveling cost of the current solution.

For the *Remove route neighbourhood*, we use the MIP-RB(24-40) model which tries to remove unnecessary routes from the solution. This model's objective is to minimize the total traveling costs of the routes. At the same time, if it is possible this model eliminates unnecessary routes from the solution which leads to a solution with lower cost. We need to make the same modifications in the MIP-RB model to use this neighbourhood in the second and third assumptions.

In the *Remove customers neighbourhood*, an attempt is made to remove the customer from the tour for each customer-tour pair. It is made by calculating the maximum amount of inventory that can be obtained from the distribution periods before the route. Additionally, the minimum amount of inventory required in the periods after the customer's route is calculated. These maximum and minimum inventory values are determined based on unused vehicle capacities.

To calculate the out-of-stock quantity, the maximum inventory is subtracted from the minimum inventory along with the customer's demand for the removed period. If the result is a non-negative value, the LP-SO (44-50) model is utilized to calculate the quantity of out-of-stock products and the deliveries. This calculation is performed for

all customer-tour pairs, and the customer offering the lowest cost among the feasible solutions is removed from the tour. Subsequently, the neighbourhood terminates.

4.2.3 Algorithm for the MCIRP

The ALNS algorithm starts after finding an initial feasible solution to the MCIRP using the MIP-RB model. We provide the MIP-RB model with 5000 different random routes. In our algorithm, we begin by selecting a destroy neighbourhood from a pool of 18 neighbourhoods. Each of these destroy neighbourhoods has changing probability to be selected and we utilize a roulette wheel mechanism in each iteration. Within our algorithm, we also have segments consisting of 50 iterations each, and after each segment, the neighbourhood selection probabilities are refreshed. The probability of selecting destroy neighbourhood d in segment j is determined by a specific probability denoted as p_{dj}. This probability is calculated based on the weight of each neighbourhood in the corresponding segment, represented as w_{dj}. The calculation is described by the following equation (57). Here, D represents the set of destroy neighbourhoods.

$$p_{dj} = \frac{w_{dj}}{\sum_{d \in D} w_{dj}} \tag{57}$$

In every 50-iteration segment, the weight of any given destroy neighbourhood undergoes an update using the formula 58 This depends on the situation of the reached solution.

$$w_{d,j+1} = w_{dj}(1 - r) + r\frac{\pi_i}{\theta_i} \tag{58}$$

The weight of destroy i to be chosen in segment $j + 1$ is based on two different components as shown in (58). The first component comprises the weight from the preceding segment, while the second component is the ratio of the total score achieved by destroy neighbourhood i to the number of times it is utilized within that segment (θ_i). These two components are combined with a reaction factor denoted as r (where

$0 \leq r \leq 1$), and for our algorithm, we set r to 0.7. The score π_i starts at zero and increases by σ each time it enhances the solution.

We use the LP-SO model to recalculate the stock-out amounts once we choose and apply a destroy neighbourhood to the current feasible solution. If the objective value from this model is greater than zero, indicating an infeasible solution, we employ repair neighbourhoods to improve this infeasible solution. On the other hand, if the objective value is 0, indicating a feasible solution, we skip the repair process. After making an infeasible solution feasible with repair neighbourhoods or after directly destroy neighbourhoods, we implement improve neighbourhoods to the solution to decrease the routing cost.

Next, we present our mathueristic algorithm for MCIRP;

In this context, "obj()" denotes the value of the objective function for a particular solution, whereas the symbols R and I refer to the repair and improvement neighbourhoods, respectively.

Algorithm 1 *Mathueristic algorithm for MCIRP*

Input: Instance of MCIRP
Output: s_{best}
Set all π_i to 0 and w_{dj} to 1
$time, iteration = 0$
s, s_{best} = Initial solution, (the output of MIP-RB model)
while $time \leq 7200$ and $iteration \leq 10000$ **do**
    j = 1
    while $j \leq 50$ **do**
        Select the destroy neighbourhood $d_i \in$ D
        $(\theta_i) = (\theta_i) + 1$
        $s' = d_i(s)$
        s'= LP-SO (s')
        if stock-out amount of $s' > 0$ **then**
            $s'' = R(s')$
            $s''' = I(s'')$
        else
            $s''' = I(s')$
        end if
        if $obj(s''') \leq obj(s_{best})$ **then**
            $s_{best} = s'''$
            $\pi_i = \pi_i + \sigma$
        end if
        $j = j + 1$
        $iteration = iteration + 1$, update the time
    end while
    update all w_{dj} and set all π_i to 0
end while

CHAPTER V

NUMERICAL RESULTS

5.1 Test Instances

In order to create data sets to test our algorithm and evaluate its performance in large-scale analyses, the literature on the MCIRP is reviewed. The data sets were created in a similar way to the one proposed in [5]. The data sets were created on a 100 x 100 map with the number of customers from 20 up to 50. The warehouse is located at coordinate (0,0) and the traveling costs between the points are calculated based on the Euclidean distance. Three types of products are studied with a planning horizon of 3, 5 and 7 days. Customer demands (d_{im}^t) were randomly generated for each customer, product and day with 40% probability falling in the range [1,2] tonnes, 40% probability falling in the range [2,3] tonnes and 20% probability falling in the range [3,6] tonnes. Customers' stocking capacity (SC_{im}) is taken as 40 tonnes for each customer-product pair. Any problem can be simplified in a data set because the initial inventories fulfill a large fraction of customer demands. To avoid this, initial inventories ($\mathcal{I}_{im}^0$) are randomly generated for each customer-product pair in the range of [0, 3] tonnes instead of [6, 12] tonnes as suggested in [5]. In [5], the range of vehicles employed in their study varies in terms of overall capacity, with the smallest vehicle capable of carrying 18 tons and the largest one accommodating 30 tons. In our case, we opt for larger vehicles with 36 unit compartments. These compartments are divided into four categories with capacities of 6, 8, 10, and 12 units respectively.

To assess the performance of our matheuristic using the ALNS algorithm, we conduct tests on the instances mentioned earlier. The evaluation is carried out on a system equipped with an i7-10750H processor running at 2.60 GHz and having

32GB RAM. We implement the algorithm using Python and rely on CPLEX 22.1 with default parameter settings for the computational results.

5.2 Results and Changes on the Algorithm

To evaluate the performance of the ALNS algorithm described in Chapter 4 (called ALNS-1), two hourly runs were taken on 14 datasets with a homogeneous fleet of vehicles with the number of periods being 3, 5 and 7, respectively. After analyzing the ALNS-1 algorithm described in Chapter 4 under the first assumption, it is found that some of the 18 destroy operators contributed to the best solution less than the others and it took a long time to repair the infeasible solution that they create.

5 destroy operators ((vii) remove all ρ customers, (ix) empty ρ period, (x) swap routes, (xi) move ρ visits, (xv) merge 2ρ routes) are selected to be removed because of their effect to the best solution found at the end of the 2-hour run, and a different version of the algorithm called ALNS-2 application without these operators is tested. Since the 5 destroy operators selected here are particularly time-consuming to repair, we eliminated them and aimed to perform more operations with the other destroy operators. The objective function values found by the ALNS and ALNS-2 algorithms with 8 different data sets and periods of 5 and 7, respectively, within two hours are compared in Table 2 and 3 on the first delivery assumption. In these tables, the value in each cell shows the percentage deviation from the best objective in the row.

When the results are examined, it is observed that removing 5 operators in the first delivery assumption provided an improvement of around 1.30% with these instances. Therefore, we decided to continue with the ALNS-2 application by removing these five operators for all of the assumptions.

After testing various methods, another change has been made based on the results obtained. It is observed that the integer model used in the first step of the improvement process (*Remove route operator*) takes a certain amount of time when

Table 2: Gap of of ALNS and ALNS-2 in the first delivery assumption, T=5

Instance		**Algorithms**	
no.	***n***	***ALNS-1***	***ALNS-2***
5	30	0.87	0.00
6	30	0.00	3.53
7	35	8.37	0.00
8	35	0.00	1.08
9	40	0.13	0.00
10	40	5.77	0.00
11	45	3.12	0.00
12	45	0.00	7.21
Avg.		**2.28**	**1.48**

Table 3: Gap of ALNS and ALNS-2 in the first delivery assumption, T=7

Instance		**Algorithms**	
no.	***n***	***ALNS-1***	***ALNS-2***
5	30	0.00	0.67
6	30	0.53	0.00
7	35	1.01	0.00
8	35	4.17	0.00
9	40	0.00	1.25
10	40	5.22	0.00
11	45	0.82	0.00
12	45	4.55	0.00
Avg.		**2.04**	**0.24**

attempting to eliminate routes. To decrease this time, the repaired solution that comes to improvement operators is examined to see how far its cost is from the current best solution's total cost. If the solution that comes to the *Remove route operator* is worse than the current best solution by more than a threshold value of $x\%$, this process is skipped. For example, suppose our current best solution's objective is 1000 and the solution that comes to the *Remove route operator* after being destroyed and repaired is 1100. In that case, this solution is 10% away from the best solution. Depending on the given threshold value, it is decided whether this solution will be included in the first improvement or not. The aim is to avoid spending more time on results that are far from the best solution.

While deciding on the threshold value, tests were conducted to examine how far the solution is from the best solution in each iteration and whether it improved. After these examinations, it is decided to conduct experiments with threshold values of 5% and 10%, as well as without a threshold. These experiments are performed with 8 different data sets and 5-7 periods in the first delivery assumption, in the order of no threshold, 5% threshold, and 10% threshold. The values for $T = 5$ are presented in Table 4, and the values for T= 7 are presented in Table 5. The values in the tables indicate the percentage by which the relevant solution deviates from the best result in the row.

As a result of the threshold analyses, it has been observed that the threshold value of 5% provides significantly better results in the first delivery assumption. Consequently, it has been decided to run the algorithm with this threshold value, and this updated version is named as ALNS-3. Eliminating some of the destroy operators and the determination of threshold values have shown positive effects on the solutions. Therefore, all assumptions are conducted using the ALNS-3 algorithm.

Table 4: Gap of improve operator thresholds in first delivery assumption, T=5

Instance		**Threshold**		
no.	***n***	***w/o***	***5%***	***10%***
5	30	0.00	0.65	1.21
6	30	0.00	1.09	0.76
7	35	8.40	0.00	1.38
8	35	2.57	4.09	0.00
9	40	2.20	0.00	10.55
10	40	2.73	0.00	4.91
11	45	3.97	0.00	4.83
12	45	6.84	0.00	4.58
Avg.		**3.34**	**0.73**	**3.53**

Table 5: Gap of improve operator thresholds in first delivery assumption, T=7

Instance		**Threshold**		
no.	***n***	***w/o***	***5%***	***10%***
5	30	1.54	0.51	0.00
6	30	4.91	0.00	1.46
7	35	2.73	1.37	0.00
8	35	6.48	0.00	7.73
9	40	3.27	0.00	0.31
10	40	1.28	0.00	4.05
11	45	1.88	0.43	0.00
12	45	4.80	0.00	4.52
Avg.		**3.36**	**0.29**	**2.26**

5.3 Lower Bound

A large-scale numerical analysis of the performance of the ALNS-3 algorithm is performed against a lower bound. Here, a lower bound is derived using the flow formulation developed by [34]. This flow formulation is organized for three delivery assumptions regarding the split distribution of MCIRP. The following notations and the parameters used for the first MCIRP assumption are also used.

$$E = \{(i,j) : i < j; i,j \in V_0\}$$

$$A = \{(i,j),(j,i) : (i,j) \in E\}$$

$$x_{ij}^t = \begin{cases} 1, & \text{if } (i,j) \text{ edge used at time period } t \\ 0, & \text{otherwise} \end{cases} \qquad (i,j) \in A, m \in M, t \in \mathcal{T}$$

$$z_i^t = \begin{cases} 1, & \text{if } i \text{ is visited at time period } t \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, t \in \mathcal{T}$$

$$f_{ijcm}^t = \begin{cases} 1, & \text{if product } m \text{ carried in compartment } c \text{ at } (i,j) \text{ edge at time period } t \\ 0, & \text{otherwise} \end{cases} \qquad (i,j) \in A, c \in \mathcal{C}, m \in M, t \in \mathcal{T}$$

q_{im}^t = Amount of product m delivered to customer i at time period t

$$m \in M, i \in V, t \in \mathcal{T}$$

I_{im}^t = Amount of inventory for product m at customer i at end of period t

$$m \in M, i \in V, t \in \mathcal{T}$$

y_{ij}^t = how many times the edge (i,j) used in period t

$$(i,j) \in E, t \in \mathcal{T}$$

l_{ijp}^t = Amount of product m being transported from node i to node j on period t

$$(i,j) \in A, m \in M, t \in \mathcal{T}$$

$$\min \quad \sum_{(i,j)\in E} \sum_{t\in T} b_{ij} y_{ij}^t \tag{59}$$

$$st. \quad \mathcal{I}_{im}^0 + q_{im}^1 = I_{im}^1 + d_{im1} \qquad \forall i \in V, m \in M \tag{60}$$

$$I_{im}^{t-1} + q_{im}^t = I_{im}^t + d_{imt} \qquad \forall i \in V, m \in M, t \in T\backslash\{1\} \tag{61}$$

$$q_{im}^1 \leq SC_{im} - \mathcal{I}_{im}^0 \qquad \forall i \in V, m \in M \tag{62}$$

$$q_{im}^t \leq SC_{im} - I_{im}^{t-1} \qquad \forall i \in V, m \in M, t \in T\backslash\{1\} \tag{63}$$

$$q_{im}^t \leq SC_{im} z_i^t \qquad \forall i \in V, m \in M, t \in T \tag{64}$$

$$\sum_{j:(i,j)\in E} y_{ij}^t + \sum_{j:(i,j)\in E} y_{ji}^t = 2z_i^t \qquad \forall i \in V, t \in T \tag{65}$$

$$\sum_{j|(i,j)\in A} l_{ijp}^t - \sum_{j|(j,i)\in A} l_{jip}^t = -q_{im}^t \qquad \text{if } i \in V, \forall i \in V_0, m \in M, t \in T \tag{66}$$

$$\sum_{j|(i,j)\in A} l_{ijp}^t - \sum_{j|(j,i)\in A} l_{jip}^t = \sum_{i\in V} q_{im}^t \qquad \text{if } i = 0, \forall i \in V_0, m \in M, t \in T \tag{67}$$

$$l_{ijp}^t \leq \sum_{h\in H} Q_c f_{ijcm}^t \qquad \forall (i,j) \in A, m \in M, t \in T \tag{68}$$

$$\sum_{j|(j,i)\in A} x_{ji}^t - \sum_{j|(i,j)\in A} x_{ij}^t = 0 \qquad \forall i \in V_0, t \in T \tag{69}$$

$$x_{ij}^t + x_{ji}^t = y_{ij}^t \qquad \forall (i,j) \in E, t \in T \tag{70}$$

$$\sum_{p\in P} f_{ijcm}^t \leq x_{ij}^t \qquad \forall (i,j) \in A, h \in H, t \in T \tag{71}$$

$$y_{ij}^t \in \{0,1\} \qquad \forall (i,j) \in E\backslash\{i,j=0\}, t \in T \tag{72}$$

$$y_{0j}^t \in \{0,1,2\} \qquad \forall j \in V, t \in T) \tag{73}$$

$$z_i^t \in \{0,1\} \qquad \forall i \in V, t \in T \tag{74}$$

$$x_{ji}^t \in \{0,1\} \qquad \forall (i,j) \in A, t \in T \tag{75}$$

$$f_{ijcm}^t \in \{0,1\} \qquad \forall (i,j) \in A, h \in H, m \in M, t \in T \tag{76}$$

The function (59) aims to minimize the overall cost incurred by the utilized edges.

Constraints (60 - 61) refers to inventory balance constraints, and the capacity constraints for each product are constraints (62 - 63). Constraints (64) determine whether a customer visited in a period. In the scenario where a customer is visited once a period, the constraints (65) and (70 - 71) ensure that every time a customer is visited, two arcs must traverse from that node. These constraints ensure the proper flow of products to and from the customer. Constraints (66) and (67) guarantee that the amount of each product transported in a vehicle throughout the day is decreased by the quantity delivered to customers as the vehicle makes its visits. The constraints (68) ensure that the vehicle does not carry more than its capacity for each specific product. Lastly, the remaining constraints define the permissible values for the decision variables.

For the second delivery scenario, the variables' domains, namely z_i^t, y_{ij}^t, and x_{ji}^t, are adjusted to accommodate non-negative integer values from 0 to p and also, the y_{0j}^t is modified to permit any non-negative integer value from 0 to $2p$.

In the third delivery scenario, where a customer can receive a product from multiple vehicles during a given period, the variables z_i^t, y_{ij}^t, and x_{ji}^t are adjusted to allow any non-negative integer value.

To evaluate the performance of our developed matheuristic algorithm for three different delivery assumptions against the lower bound obtained by MIP-Flow, we run our generated algorithm for two hours with a 1000 iteration limit. We compare the results of our algorithm with the lower bound gained by the CPLEX with a one-hour time limit for each instance. The comparisons can be seen in Tables 6, 7, 8. These tables display the percentage deviation of our metheuristic's total cost from the lower bound determined by MIP-Flow using CPLEX.

Table 6: Optimality Gap of Flow vs ALNS-3 in first delivery assumption

Instance		**Time Periods**		
no.	***n***	***T=3***	***T=5***	***T=7***
1	20	20.44	14.99	12.63
2	20	24.77	15.06	15.01
3	25	24.36	15.97	14.59
4	25	14.96	15.46	14.17
5	30	19.63	15.63	14.56
6	30	17.57	15.70	12.29
7	35	17.44	11.75	13.82
8	35	20.46	17.13	11.63
9	40	20.27	12.60	13.45
10	40	23.53	14.12	11.93
11	45	20.64	15.06	14.33
12	45	22.39	25.04	11.30
13	50	23.77	17.18	15.57
14	50	23.98	18.74	18.26
Avg.		**21.02**	**16.03**	**13.82**

When these results are examined, it is seen that the ALNS-3 method gives solutions with a gap value of around 18% worse on average. It is observed that the distance of the solutions from the lower bound decreases as the number of periods in the problem increases. As the number of periods increases, it is observed that long-term costs are better managed by the solution method. It should be noted that these comparisons are against a conservative lower bound.

To enhance our comprehension of the algorithm's effectiveness, we used six different instances for evaluation. Within this analysis, we run the main model (1) - (22), to make comparisons against our algorithm. Since solving the MIP model is time-consuming,it was not possible to run the MIP model for each instance we have. Instead, we use 3 toy instances with 20 customers with time periods as 3, 5, 7 and also we selected 3 mid-level instances with 35 customers with time periods as 3, 5, 7. We are running the MIP model for each instance with a 3-day time limit. In Figure

Table 7: Optimality Gap of Flow vs ALNS-3 in second delivery assumption

Instance		**Time Periods**		
no.	***n***	***T=3***	***T=5***	***T=7***
1	20	21.58	19.47	21.19
2	20	23.50	14.68	19.90
3	25	20.10	17.22	19.66
4	25	21.09	18.39	20.72
5	30	22.08	22.24	16.14
6	30	20.02	21.05	16.42
7	35	27.05	15.00	17.97
8	35	22.25	17.93	17.34
9	40	24.53	14.49	15.76
10	40	23.62	16.24	17.17
11	45	27.27	18.38	16.56
12	45	23.12	20.28	13.57
13	50	26.78	22.34	16.02
14	50	24.53	18.05	16.57
Avg.		**23.39**	**18.27**	**17.50**

4 and 5, we can see the percentage deviation of ALNS algorithm results against the MIP model's best integer solution as the time changes for each different instance sets. Here it can be seen that if we could compare our results against a feasible solution taken from MIP, our results achieve better in a shorter time since our algorithm runs with a 2-hour time limit and the shown results are after 3-day run.

Additionally, the flow formulation we use for result comparison serves as a conservative lower bound. In Figure 6 and 7, using the same instances, we illustrate the percentage deviation between the lower bound of the flow formulation and that of the MIP model. It is evident that as time increases, the deviation becomes more obvious. Also, we see an overall positive percentage which validates that flow formulation is a conservative lower bound since it gives higher costs than the used lower bound.

Table 8: Optimality Gap of Flow vs ALNS-3 in third delivery assumption

Instance		**Time Periods**		
no.	***n***	***T=3***	***T=5***	***T=7***
1	20	22.47	15.69	16.76
2	20	15.46	11.72	16.11
3	25	18.03	17.94	14.86
4	25	21.33	19.91	14.54
5	30	16.70	22.04	18.78
6	30	13.54	19.19	16.15
7	35	14.02	15.70	13.82
8	35	25.78	18.17	13.49
9	40	20.17	15.79	15.83
10	40	24.75	17.06	16.74
11	45	23.00	18.15	16.45
12	45	23.32	16.71	14.49
13	50	19.97	17.43	15.59
14	50	22.17	17.76	16.37
Avg.		**20.05**	**17.38**	**15.71**

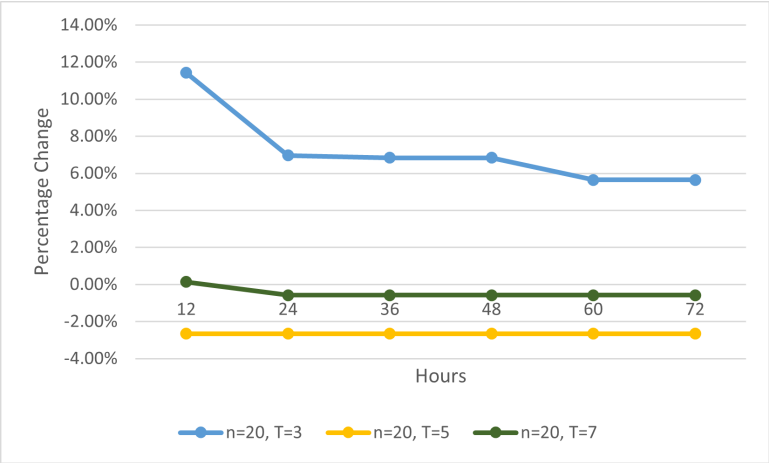

Figure 4: Gap of ALNS-3 results between MIP Model's best integer when n=20

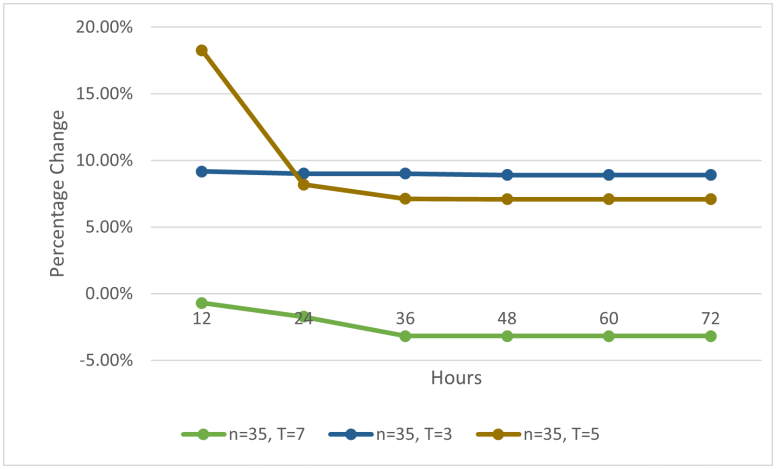

Figure 5: Gap of ALNS-3 results between MIP Model's best integer when n=35

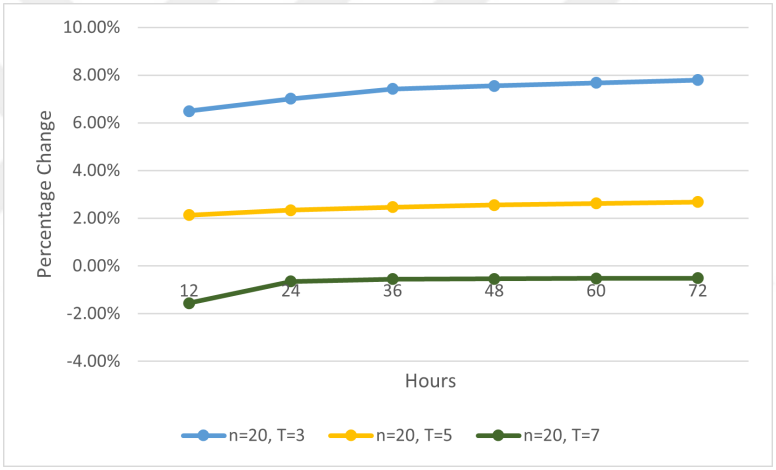

Figure 6: Gap of Flow Formulation's lower bound between MIP Model's lower bound when n=20

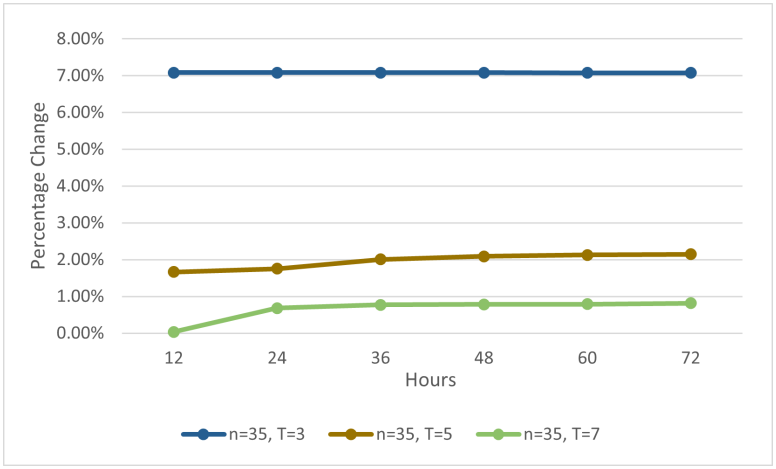

Figure 7: Gap of Flow Formulation's lower bound between MIP Model's lower bound when n=35

CHAPTER VI

CONCLUSIONS

This study addresses the Multi-Compartment Inventory Routing Problem (MCIRP) by utilizing a fleet of vehicles with flexible product types and fixed compartment sizes. Each of the compartments in these vehicles has a specific number of unit capacities and can accommodate all product types. The supplier manages the customer's inventory levels by creating a distribution plan with these vehicles with the knowledge of the customer's inventory. The used Vendor Managed Inventory (VMI) approach grants the supplier the ability to manage customer inventory levels by devising a distribution plan based on customer inventory and capacity knowledge.

Determining an allocation of each product to the compartments in different vehicles, accounting for varying routes and periods, as well as deciding the appropriate quantities to deliver to each customer, constitutes a complex decision-making process that cannot be effectively solved using exact methods. To address this challenge and develop a high-quality distribution network, we propose an Adaptive Large Neighborhood Search (ALNS) based matheuristic as our solution method. Subsequently, we investigate three different delivery assumptions within different scenarios, focusing on the number of daily customer visits.

The effectiveness of our solution method is demonstrated through a comparison with a lower bound established by [34], utilizing various test instances. These instances involve up to 50 customers and a 7-day inventory planning horizon. We compared the performance of our algorithm with a mathematical model employing flow formulation, we observed that our algorithm achieves solutions with an average cost 18.13% higher than the lower bound. This discrepancy is relatively understandable

since the flow formulation fails to provide feasible solutions for our problem, instead, it yields conservative lower bounds. To better understand our proposed algorithm's achievement, we run the main MIP model for 3 days and compare its best integer and lower bound against our results.

In conclusion, our study presents a novel solution method for the MCIRP, employing VMI and an ALNS-based matheuristic approach. Further research can explore the utilization of alternative heuristics or metaheuristics to address the MCIRP and other inventory routing problems. Future research endeavors may aim to bridge the existing gap in this area of study with starting our study.

REFERENCES

[1] R. Lahyani, L. C. Coelho, M. Khemakhem, G. Laporte, and F. Semet, "A multi-compartment vehicle routing problem arising in the collection of olive oil in tunisia," *Omega*, vol. 51, pp. 1–10, 2015.

[2] P. Avella, M. Boccia, and A. Sforza, "Solving a fuel delivery problem by heuristic and exact approaches," *European Journal of Operational Research*, vol. 152, no. 1, pp. 170–179, 2004.

[3] L. C. Coelho and G. Laporte, "Classification, models and exact algorithms for multi-compartment delivery problems," *European Journal of Operational Research*, vol. 242, no. 3, pp. 854–864, 2015.

[4] F. Cornillier, F. F. Boctor, G. Laporte, and J. Renaud, "A heuristic for the multi-period petrol station replenishment problem," *European Journal of Operational Research*, vol. 191, no. 2, pp. 295–305, 2008.

[5] D. Popović, M. Vidović, and G. Radivojević, "Variable neighborhood search heuristic for the inventory routing problem in fuel delivery," *Expert Systems with Applications*, vol. 39, no. 18, pp. 13390–13398, 2012.

[6] M. Vidović, D. Popović, and B. Ratković, "Mixed integer and heuristics model for the inventory routing problem in fuel delivery," *International Journal of Production Economics*, vol. 147, pp. 593–604, 2014.

[7] M. Christiansen, K. Fagerholt, T. Flatberg, Ø. Haugen, O. Kloster, and E. H. Lund, "Maritime inventory routing with multiple products: A case study from the cement industry," *European Journal of Operational Research*, vol. 208, no. 1, pp. 86–94, 2011.

[8] N. Siswanto, D. Essam, and R. Sarker, "Solving the ship inventory routing and scheduling problem with undedicated compartments," *Computers & Industrial Engineering*, vol. 61, no. 2, pp. 289–299, 2011.

[9] M. Ostermeier, T. Henke, A. Hübner, and G. Wäscher, "Multi-compartment vehicle routing problems: State-of-the-art, modeling framework and future directions," *European Journal of Operational Research*, vol. 292, no. 3, pp. 799–817, 2021.

[10] C. Archetti, N. Boland, and M. Grazia Speranza, "A matheuristic for the multi-vehicle inventory routing problem," *INFORMS Journal on Computing*, vol. 29, no. 3, pp. 377–387, 2017.

[11] L. C. Coelho, J.-F. Cordeau, and G. Laporte, "The inventory-routing problem with transshipment," *Computers & Operations Research*, vol. 39, no. 11, pp. 2537–2548, 2012.

[12] V. C. Hemmelmayr, K. F. Doerner, and R. F. Hartl, "A variable neighborhood search heuristic for periodic routing problems," *European Journal of Operational Research*, vol. 195, no. 3, pp. 791–802, 2009.

[13] F. Cornillier, F. F. Boctor, G. Laporte, and J. Renaud, "An exact algorithm for the petrol station replenishment problem," *Journal of the Operational Research Society*, vol. 59, pp. 607–615, 2008.

[14] F. Cornillier, G. Laporte, F. F. Boctor, and J. Renaud, "The petrol station replenishment problem with time windows," *Computers & Operations Research*, vol. 36, no. 3, pp. 919–935, 2009.

[15] F. Cornillier, F. Boctor, and J. Renaud, "Heuristics for the multi-depot petrol station replenishment problem with time windows," *European Journal of Operational Research*, vol. 220, no. 2, pp. 361–369, 2012.

[16] H. Yahyaoui, I. Kaabachi, S. Krichen, and A. Dekdouk, "Two metaheuristic approaches for solving the multi-compartment vehicle routing problem," *Operational Research*, vol. 20, pp. 2085–2108, 2020.

[17] A. Hübner and M. Ostermeier, "A multi-compartment vehicle routing problem with loading and unloading costs," *Transportation Science*, vol. 53, no. 1, pp. 282–300, 2019.

[18] L. van der Bruggen, R. Gruson, and M. Salomon, "Reconsidering the distribution structure of gasoline products for a large oil company," *European Journal of Operational Research*, vol. 81, no. 3, pp. 460–473, 1995.

[19] I. Moon, S. Salhi, and X. Feng, "The location-routing problem with multi-compartment and multi-trip: formulation and heuristic approaches," *Transportmetrica A: Transport Science*, vol. 16, no. 3, pp. 501–528, 2020.

[20] J. E. Mendoza, B. Castanier, C. Guéret, A. L. Medaglia, and N. Velasco, "A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands," *Computers & Operations Research*, vol. 37, no. 11, pp. 1886–1898, 2010.

[21] L. Wang, J. Kinable, and T. Van Woensel, "The fuel replenishment problem: A split-delivery multi-compartment vehicle routing problem with multiple trips," *Computers & Operations Research*, vol. 118, p. 104904, 2020.

[22] P. Chokanat, R. Pitakaso, and K. Sethanan, "Methodology to solve a special case of the vehicle routing problem: A case study in the raw milk transportation system," *AgriEngineering*, vol. 1, no. 1, pp. 75–93, 2019.

[23] J. Oppen, A. Løkketangen, and J. Desrosiers, "Solving a rich vehicle routing and inventory problem using column generation," *Computers & Operations Research*, vol. 37, no. 7, pp. 1308–1317, 2010.

[24] L. C. Coelho, J.-F. Cordeau, and G. Laporte, "Consistency in multi-vehicle inventory-routing," *Transportation Research Part C: Emerging Technologies*, vol. 24, pp. 270–287, 2012.

[25] M. M. Abdulkader, Y. Gajpal, and T. Y. ElMekkawy, "Hybridized ant colony algorithm for the multi compartment vehicle routing problem," *Applied Soft Computing*, vol. 37, pp. 196–203, 2015.

[26] M. Alinaghian and N. Shokouhi, "Multi-depot multi-compartment vehicle routing problem, solved by a hybrid adaptive large neighborhood search," *Omega*, vol. 76, pp. 85–99, 2018.

[27] P. V. Silvestrin and M. Ritt, "An iterated tabu search for the multi-compartment vehicle routing problem," *Computers & Operations Research*, vol. 81, pp. 192–202, 2017.

[28] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation Science*, vol. 40, pp. 455–472, Nov. 2006.

[29] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *Principles and Practice of Constraint Programming — CP98*, pp. 417–431, Springer Berlin Heidelberg, 1998.

[30] Y. Adulyasak, J.-F. Cordeau, and R. Jans, "Optimization-based adaptive large neighborhood search for the production routing problem," *Transportation science*, vol. 48, no. 1, pp. 20–45, 2014.

[31] K. Helsgaun, "General k-opt submoves for the lin–kernighan tsp heuristic," *Mathematical Programming Computation*, vol. 1, pp. 119–163, 2009.

[32] D. Aksen, O. Kaya, F. S. Salman, and Ö. Tüncel, "An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem," *European Journal of Operational Research*, vol. 239, no. 2, pp. 413–426, 2014.

[33] P. Shaw, "A new local search algorithm providing high quality solutions to vehicle routing problems," *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, vol. 46.

[34] C. Archetti, N. Bianchessi, and M. G. Speranza, "Branch-and-cut algorithms for the split delivery vehicle routing problem," *European Journal of Operational Research*, vol. 238, no. 3, pp. 685–698, 2014.

VITA

Sena Artarslan graduated from İstanbul Anatolian High School in 2016. She earned her B.Sc. degree in Industrial Engineering from Özyeğin University in June 2021. She has been pursuing her M.Sc. degree in Industrial Engineering at Özyeğin University since September 2021, under the supervision of Prof. Ali Ekici. Her research focuses on route planning, optimization and machine learning applications.