AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH FOR THE MULTI-COMPARTMENT INVENTORY ROUTING PROBLEM

A Thesis

by

Ceren Gültekin

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
June 2021

AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH FOR THE MULTI-COMPARTMENT INVENTORY ROUTING PROBLEM

Approved by:

Associate Professor Okan Örsan Özener, Advisor
Department of Industrial Engineering
Özyeğin University

Assistant Professor İhsan Yanıkoğlu
Department of Industrial Engineering
Özyeğin University

Associate Professor Ali Ekici, Co Advisor
Department of Industrial Engineering
Özyeğin University

Associate Professor Ertan Yakıcı
Department of Industrial Engineering
National Defence University

Date Approved: 10 June 2021

Assistant Professor Mehmet Önal
Department of Industrial Engineering
Özyeğin University

To my loving family and friends

ABSTRACT

In this thesis study, we concentrate on an inventory routing problem with a fleet of multi-compartment vehicles which enables the distribution of different products to customers on a delivery route. Using separate compartments on a vehicle increases profitability and customer satisfaction when customer demands vary over product and period basis. We assume that the compartment that each product can be loaded is known and the capacities of the compartments are fixed. Customers have preset storage capacities and distribution plans should be made in a way that no customers would face stock-outs for any product on any day. We observe the practices of this variant in the distribution of foods with different temperature needs to groceries, feed distribution to livestock farms, and collection of different types of recyclable wastes. We examine this problem separately for three assumptions considering different cases of allowing/disallowing split delivery to customers. We propose a matheuristic approach to solve the addressed problem where we systematically integrate an Adaptive Large Neighborhood Search algorithm with mathematical programming models. We generate a set of instances and test the performance of our algorithm by comparing it with the results obtained by a flow formulation adapted from the literature. We observe that the best results we find for each instance are only %11.7 worse than the solutions found by the flow formulation on average.

ÖZETÇE

Bu tez çalışmasında, farklı ürünlerin müşterilere aynı rotada dağıtımına olanak sağlayan çok bölmeli araçlardan faydalandığımız bir envanter rotalama problemini ele almaktayız. Müşteri taleplerinin gün ve ürün bazında farklılık gösterdiği bir durumda araç içerisinde ayrı bölmeler kullanmak hem karlılığı hem de müşteri memnuniyetini arttırmaktadır. Araçlarda her bölmede hangi ürünün taşınacağının belirli olduğu ve bölme kapasitelerinin sabit olduğu bir yapıyı benimsiyoruz. Müşterilerin belirli stoklama kapasiteleri olduğunu ve dağıtım planlarının hiçbir müşterinin hiçbir günde hiçbir üründen stok-dışı kalmayacağı şekilde yapılması gerektiğini varsayıyoruz. Bu bölme yapısının kullanıldığı çok bölmeli rotalama problemlerinin uygulamalarına marketlere farklı sıcaklıkta taşınması gereken gıdaların dağıtımı, hayvan çiftliklerine yem dağıtımı, ve çeşitli geri dönüştürülebilir atıkların toplanması alanlarında rastlanmaktadır. Bu problemi müşterilere bölünmüş dağıtımın izin verildiği/verilmediği farklı durumları göz önünde bulundurarak üç farklı varsayım için ayrı ayrı inceliyoruz. Problemin çözümü için Uygulanabilir Geniş Komşuluk Araması algoritmasını matematiksel modeller ile sistematik bir şekilde kullandığımız mat-sezgisel bir yöntem sunuyoruz. Algoritmamızın performansını literatürden uyarladığımız bir akış formülasyonundan elde ettiğimiz sonuçlar ile oluşturduğumuz bir dizi veri kümesi üzerinde karşılaştırarak değerlendiriyoruz. Her bir veri kümesinde algoritmamızın bulduğu en iyi sonucun akış formülasyonu ile bulunan sonuçtan ortalamada sadece %11.7 daha kötü olduğu gözlemlenmektedir.

ACKNOWLEDGEMENTS

I would like to start by thanking my dear professors Dr. Okan Örsan Özener and Dr. Ali Ekici, who did not only guide me in this thesis study with their valuable comments but also always shared their experiences and pieces of advice with me which I benefit enormously. I also owe Dr. Burcu Balçık a debt of gratitude for all the support and guidance she has provided me during my master's. I feel very lucky to have studied with them and witnessed their manner of work. I would also like to thank Dr. İhsan Yanıkoğlu for the insightful feedback he provided during this study.

I am deeply grateful for the support and companionship of Berk. We have started our master's degree with a similar sense of ambition and worked hard to accomplish our goals. I would like to thank him for all the things I have learned from him as we work together, and for reminding me of my motivation at the times I felt tired.

My dear family has always been respectful of my decisions and supported me in any way. I would like to thank my mom for ambitiously trying her best to make sure my brother and I are capable individuals with hunger in improving ourselves. I would like to thank my dad for his never-ending compassion and trust in me. I would like to thank my brother for all the joy he brings in my life and for being the best housemate for five years. I would like to thank my grandmother for her endless support in growing us. She is my role model with her attitude, diligence, and joy of life.

My office friends have made this journey memorable to me. I would like to thank Birce, Ahmet, Eren, Kaan, and Merve for helping me approach anything with humour even when we feel stressed. I will always remember the moments of laughter in our office with a smile on my face.

Finally, I would like to thank my dear friends and family members, for whom I cannot express my gratitude individually in this restricted space. I feel blessed to have their caring and supportive presence in my life.

This research is supported in part by TÜBİTAK grant 119M245.

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER I

INTRODUCTION

Theoretical studies and practical applications show that cooperation and coordination in supply chain networks provide significant benefits for all stakeholders. In the proposed study, the supplier manages customers' inventories, hence plans the distribution to different customers in coordination. Thus, the supplier can reduce transportation and inventory costs by determining distribution routes, frequency of visits, and distribution amounts simultaneously. From the customer point of view, the advantages are: (i) customers are exempted from dealing with inventory management, (ii) it is less likely to have stock-outs, (iii) the bull-whip effect tends to decrease.

This application, studied as the Inventory Routing Problem (IRP) in the literature, brings along difficult decision-making processes. Technically, IRP arises as a generalization of the capacitated version of the VRP, which is known to be NP-hard [1]. Hence, IRP is classified as NP-hard as well. While Vehicle Routing Problems (VRP) usually deal with a single-period problem, where inventory-related decisions are not considered, IRP considers a multi-period planning horizon, where customers can keep inventory. Determining vehicle routes and inventory levels simultaneously in the IRP while ensuring that customers' demands are satisfied on time, storage capacities, and vehicle capacities are not exceeded make the problem challenging. In the IRP, the decisions made within one period critically affect the following periods. For example, a low amount of product delivered to a customer, necessitates an urgent visit to that customer in the following periods. Similarly, such a trade-off that the customers regularly being served of low amount of products need to be visited more frequently, is an important point in determining the distribution plans. In the light

of all these, making the routing and inventory decisions simultaneously for a multi-period planning horizon brings a lot of complexity and causes the solution space to be quite large. When we consider different periods and different distribution strategies together, a vast number of feasible solutions arise even when the numbers of customers and periods in the planning horizon are low, so it can be very difficult to find the optimal solution even for small-sized instances of IRP [2].

To cope with these difficulties, various simplifying assumptions are applied in the literature. First of all, the problem is assumed to deal with single product distribution and the interaction between different products is ignored. For example, studies that focus on industrial gas distribution [3, 4], formulate the problem over a single product and develop solution methods accordingly, whereas indeed, the supplier distributes more than one industrial gas (oxygen, hydrogen, nitrogen, and argon) to customers. Similarly, in [5], the authors assume that there is only one product in the distribution of different products to supermarket chains. Secondly, in some studies, a very short planning horizon is used for the problem (see 3 and 6 periods) and at the same time, vehicle capacities are assumed to be too high compared to customer storage capacities. Therefore, the problem is very similar to the Traveling Salesman Problem (TSP) and exact solution methods can be developed owing to this assumption [6, 7, 8, 9]. Finally, to simplify the problem (i) order-up-to-level [7, 10] and (ii) zero-inventory ordering strategy [11, 12], are assumed as the replenishment strategy. Despite these assumptions make it easier to solve the problem, they reduce the effectiveness of coordination as they significantly reduce the supplier's distribution options.

Unlike most of the distribution systems, we consider that a single supplier distributes multiple products to customers, rather than the one supplier, one product assumption. Product distribution to food markets, distribution of petroleum products to gas stations by tankers, industrial gas distribution, and collection of different types

of recyclable wastes are examples of using the vehicle capacity effectively among different products, thus increasing the utilization rate. In such cases where the products to be delivered to customers from a warehouse/distributor are not homogeneous, multi-compartment vehicles are used to transport different products in the same vehicle, in separate compartments, in order to reduce transportation costs. In this context, it is possible to achieve higher operational efficiency compared to single product distribution models, however on the other hand, distribution of multiple products makes this distribution problem even more complex.

In the IRP variant examined in this study, we develop a solution method to determine the distribution plans in order for the supplier to meet the demands of many customers for different products in a timely and cost-effective manner. Customers request each product type from the supplier with different periodic demand quantities along the planning horizon, and there is a prespecified storage capacity for each product. The aim of the supplier is to develop and implement the least costly distribution strategy in a way that customer demands are fully satisfied on time by using the available vehicle fleet. Vehicles with multiple compartments are used for distribution and only one type of product can be carried in one compartment. Since mixing of different products is not allowed, they must be transported in separate compartments. This problem is named the Multi-Compartment - Inventory Routing Problem (MC-IRP).

The fact that there are multiple products to distribute in the MC-IRP and the demands for these products are different makes the problem more difficult compared to IRP, which is already a very difficult problem. Since customers consume the products at different rates, distribution needs may arise for different products at different times and the supplier may need to visit the same customer at different times for different products. At this point, instead of a simple approach, such as dispatching the vehicle with a high payload as in classical IRP, a complex decision

process requiring determining the amount loaded for each product in each tour and delivery volume to each customer arises.

We assume that vehicles have fixed and dedicated compartments, meaning that compartment sizes are not configurable and the compartment in which each product to be loaded is predetermined. This assumption is employed in various real-life applications of the multi-compartment routing studies in the literature; collection of different types of recyclable wastes (colored glass, normal glass, plastic, paper, etc.) [13, 14], feed distribution to livestock farms [15], distribution of foods that need to be transported under different storage conditions to markets [16]. Although similar multi-compartment routing problems are studied in the literature, studies address the problem in a single-period manner and solve the Multi-Compartment Vehicle Routing Problem (MC-VRP) [13, 14, 15, 16, 17, 18, 19, 20]. In this sense, an MC-IRP in which distribution is planned with dedicated and fixed compartments is thoroughly examined for the first time in this study. In line with relevant studies in the literature, we examine the MC-IRP under three different assumptions considering different cases of allowing/disallowing split delivery to customers; (i) a customer can be visited by only one vehicle on a day [13, 21, 22, 23],(ii) a customer can be visited by multiple vehicles on a day, as long as once for each product (i.e., delivery of one product to a customer on a day can be made by a single vehicle) [15, 17, 18, 19], (iii) delivery of one product to a customer on a day can be made by more than one vehicle [24].

To address the complex decision processes of the MC-IRP, we develop an effective and efficient Adaptive Large Neighborhood Search (ALNS) based matheuristic. Using a matheuristic approach enables us to tackle some decisions (e.g., distribution amounts to customers) by solving them to optimality in a reasonable time with the use of mathematical models. Another benefit is that when the decision-maker wants to change any assumptions, since it will be easy to update the mathematical models, the algorithm is likely to be modified easily as well.

CHAPTER II

LITERATURE REVIEW

Despite there is vast literature regarding the Multi-Compartment - Vehicle Routing Problem (MC-VRP), in which the distribution is planned for only one period/day, MC-IRP studies that examine the problem by considering a planning horizon of multiple periods/days are relatively limited. Therefore, we review the studies on MC-VRP, which are thought to be related to our problem, in addition to the restricted MC-IRP literature.

There exist many applications of MC-IRP and MC-VRP in various industries. While the distribution of petroleum products by trucks to tankers located underground at gas stations [25, 26, 27, 28, 29, 30, 31, 32], domestic waste collection for recycling purposes [14, 19, 33, 34], fresh, frozen and/or ready-made food transportation [35, 36, 37, 38, 39], are the most common applications of this variant among routing problems, the studies on MC-IRP are mostly concentrated in the maritime transportation industry [13, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50]. These studies are named as the Maritime Inventory Routing Problem (MIRP) in the literature. Routing and scheduling problems in maritime transportation show many differences compared to the problems in land transportation. These operational differences are discussed in detail by the authors in [42] and [49]. The most prominent features that distinguish the MIRP from the MC-IRP can be explained as follows:

- There are multiple supply points in maritime transportation and it is important not to exceed the storage capacities at these supply points in order to maintain production.
- In maritime transportation, vessels unload their loads completely at one or more

nodes and travel times between demand nodes take very long (i.e., days), while the amount of products delivered to customers in land transportation is much less than the capacity of the vehicles and therefore many more demand nodes are visited in one tour.

- While in land transportation, the vehicle fleet usually consists of homogenous trucks, it is common to use vessels that differ from each other in terms of size, compartment structure and cost in maritime transportation.

Due to these differences, we cover the studies on the applications of multi-compartment routing problems in land transportation. We first discuss different compartment structures, and delivery strategies in these studies.

2.1 Classification of the multi-compartment routing studies

In some applications of the multi-compartment routing problems, there is a specific compartment in the vehicle dedicated to each product shipped, and each product can only be transported in the compartment dedicated to that product [15, 14, 17, 18, 19, 51, 52]. In food transportation, where we observe this feature most frequently in daily life, when distributing fresh and frozen products to the markets, multi-compartment vehicles are used to provide separate temperature conditions in a vehicle [35, 39, 53]. In [15], authors provide another example in animal feed distribution, where each compartment is dedicated to transport a specific animal feed due to hygiene rules. This requirement arises from the fact that rabbits have low immunity against viruses, hence the potential threat that compartments used to carry bovine feed pose to rabbits.

While the number and capacity of compartments in a vehicle can be constant [13, 37, 54, 55, 56, 57, 58], they can also be configurable in the presence of separators [34, 53, 59, 60, 61]. In [59], the authors study an MC-VRP in a waste glass collection framework, where glasses are assorted by color. Here, waste glasses of various

colors are transported in separate compartments, whose capacities can be adjusted discretely. On the other hand, in [34] and [36], authors consider the compartment capacities as continuous variables. In [34], the authors compare the case in which compartment capacities are determined among a variety of scenarios (i.e., discretely) with the case where compartment capacities are defined as continuous variables, by using exact methods. They conclude that with a high number of products, the second option generates better objective function values, but there is no significant difference between these two options when the product variety is low.

In [24], the authors thoroughly discuss the splitability feature of the compartments in a fuel delivery context. When there are split compartments in a vehicle, it means that the amount of product loaded in compartments can be shared among the visited demand nodes (i.e., an unsplit compartment has to discharge all its load at any visited demand node). In the transportation of liquids such as petroleum and its derivatives, debit meters are frequently used to split the load in the compartment between customers [24, 41, 62]. Compartments without debit meters have to discharge all the load in them when the unloading begins, therefore the transported product cannot be split between multiple stations [25, 26, 27, 28, 29, 30, 63, 64].

The splitability of customer demands, on the other hand, determines if a customer can be visited by multiple vehicles on a day/period. Here, we consider three different cases: (i) each customer can only be visited by one vehicle/tour per day [13, 21, 22, 23], (ii) a customer can be visited once for each product on a day [15, 17, 18, 19], (iii) A product can be delivered to a customer by more than one vehicle/tour on a day [24]. In this study, we concentrate on the proposed MC-IRP under these three assumptions.

2.2 Multi-compartment inventory routing studies

There is very limited literature on the MC-IRP [24, 25, 29, 30, 58, 64, 65] and all of these studies assume that the compartment capacities are fixed and there is

no product-compartment match. In [65], the authors examine an MC-IRP where livestocks are collected from farms and transported to slaughterhouses by multi-compartment vehicles. Due to health/hygiene conditions, one of the breeding farms may be required to be the first destination on a tour, likewise, one of the diseased farms, if any, may be needed to be visited last. Finally, the animals can only stay one night in the slaughterhouse prior to slaughter, so there is not much inventory held. They develop a column generation based exact solution method. Since vehicles have a low number of compartments, the number of feasible tours is considerably reduced and 3-day test instances can be solved by the exact solution method.

In [25, 29, 30, 64], the authors study the problem of fuel distribution to gas stations on a multi-period basis and as a simplifying assumption, they assume that compartments are not equipped with debit meters which implies that when a customer is visited to be delivered a product, all the content in the respective compartment must be emptied. This way, they also limit the number of customers to be visited on a route by the number of compartments in the vehicles. In [25] the authors limit the feasible region assuming that at most two stations are visited in each tour. They solve the problem with an exact decomposition method to deal with the truck loading and routing problems separately. In [29], the authors examine an MC-IRP in which they assume a homogenous fleet of vehicles and compartments with equivalent capacities. Similar to [25], they do not allow more than three customers to be visited in a route. Finally, they assume that each customer's demand for each product does not change from one period to another. Under all these very restrictive assumptions, they develop a Variable Neighborhood Search (VNS) method to solve the problem. In [30], the authors analyze the same problem under the same assumptions, except for the vehicle fleet. Distribution is performed by vehicles with two or four compartments each of which has equivalent capacities. They develop a Variable Neighborhood Descent algorithm to solve the addressed problem. In [64], authors first use an inventory model

to determine the optimal order quantities and time windows for each customer. Then, based on the output of the inventory models, they determine distribution routes by solving a sequence of mixed-integer programming models, which are further improved by a VNS procedure. In [24], the authors approach the MC-IRP for two cases in terms of whether there is a debit meter or not. They consider different split delivery assumptions with a heterogeneous distribution fleet. They develop a branch-and-cut exact solution method to solve the problem, by which they can obtain the optimal solution for only very limited data sets (10 customers and 3 periods).

Finally, the authors in [58] examine the optimal planning of production, inventory, and distribution of chemical fluids. Due to the challenging nature of the problem, they decompose the problem, wherein the first stage they use a column generation approach to generate a set of multi-period routes, and in the second stage they feed these routes to a mixed integer-linear programming model to solve the periodic IRP.

The contribution of this study can be summarized as follows; (i) We are the first to examine the multi-compartment inventory routing problem (MC-IRP) with fixed and dedicated compartments. We ignore the simplifying assumptions that are commonly applied for restricting the feasible region/problem size in solving inventory routing problems so that our study has higher applicability in real-world operations. In this regard, we assume that the customer demands vary on the product and period basis, hence instead of a static problem structure, we address a dynamic and challenging problem. (ii) In recent years, for various optimization problems (especially for the routing problems) many matheuristic approaches have been successfully developed, in which metaheuristic approaches and mathematical models are used in an integrated manner [8, 66, 67, 68, 69, 70, 71]. However, there is no matheuristic approach developed to solve the MC-IRP in the literature. Therefore, the Adaptive Large Neighborhood Search (ALNS) algorithm we propose in this study is the first matheuristic in the literature to solve the MC-IRP. The problem addressed in

this study takes the special conditions and constraints of the real-world distribution systems into account, hence it is expected to be quite beneficial in increasing the profitability and customer satisfaction of companies in their supply chain operations.

CHAPTER III

PROBLEM DEFINITION AND FORMULATION

We define the problem on a Euclidean graph G = (V_0, E), where $V_0 = \{0, 1, \ldots, n\}$ is the set of customer locations ($V = \{1, \ldots, n\}$), and the depot (0), E is the set of edges connecting the nodes in V_0. The cost of traveling along an edge (i, j) is denoted by b_{ij}. Customers consume/demand m different products along T days, where the set of products is denoted by $P = \{1, \ldots, m\}$, the planning horizon is denoted by $\mathcal{T} = \{1, \ldots, T\}$ and the demand of customer i from product p on day t is denoted by d_{ipt}. The storage capacity of customer i for product p is denoted by C_{ip}. Each customer i has an initial inventory level of $\mathcal{J}_{ip}^0$ from product p. We do not allow any customer to be out of stock in any product at any time, therefore we do not consider backordering. A homogenous fleet of multi-compartment vehicles is dispatched from the depot and perform the distribution of products. Vehicle compartments are dedicated to carrying specific products and have fixed capacities. Q_p denotes the capacity allocated for product p in a vehicle. As in similar MC-VRP studies in the literature, the length of each route is restricted by L [13, 15, 17, 18, 21, 51, 72]. The objective is to construct the least costly distribution plan along the planning horizon, in a way that no customer will have stock-outs from any product. The cost is considered to be directly proportionate to the length of the distribution routes used.

We examine the MC-IRP under three different assumptions; (i) a customer can be visited by only one vehicle on a day, (ii) a customer can be visited once for each product on a day, (iii) delivery of one product to a customer on a day can be made by more than one vehicle. We modify our approach for each of these assumptions.

We formulate the problem as a mixed-integer programming model. Here, K represents the set of possible routes and can be identified as setting an upper bound for the maximum number of routes.

We use the following decision variables in our formulation:

q_{iptk} = Amount of product p delivered to customer i on day t by vehicle k

$$\forall i \in V, p \in P, t \in \mathcal{T}, k \in K$$

$$x_{ijtk} = \begin{cases} 1, & \textit{if} \text{ vehicle } k \text{ travels from node } i \text{ to node } j \text{ on day } t \\ 0, & \textit{otherwise} \end{cases}$$

$$\forall i, j \in V_0, t \in \mathcal{T}, k \in K$$

I_{ipt} = Amount of product p in the inventory at customer i on day t

$$\forall i \in V, p \in P, t \in \mathcal{T}$$

$$y_{iptk} = \begin{cases} 1, & \textit{if} \text{ vehicle } k \text{ delivers product } p \text{ to customer } i \text{ on day } t \\ 0, & \textit{otherwise} \end{cases}$$

$$\forall i \in V, p \in P, t \in \mathcal{T}, k \in K$$

We present below the model where we assume the delivery of a product to a customer on a day can be made by more than one vehicle.

$$\min \quad \sum_{i \in V_0} \sum_{j \in V_0} \sum_{t \in \mathcal{T}} \sum_{k \in K} b_{ij} x_{ijtk} \tag{1}$$

$$st. \quad \mathcal{J}_{ip}^{0} + \sum_{k \in K} q_{ip1k} = I_{ip1} + d_{ip1} \qquad \forall i \in V, p \in P \tag{2}$$

$$I_{ip,t-1} + \sum_{k \in K} q_{iptk} = I_{ipt} + d_{ipt} \qquad \forall i \in V, p \in P, t \in \mathcal{T} \backslash \{1\} \tag{3}$$

$$\sum_{i \in V} q_{iptk} \leq Q_p \qquad \forall p \in P, t \in \mathcal{T}, k \in K \tag{4}$$

$$\sum_{i \in V_0} \sum_{j \in V_0} b_{ij} x_{ijtk} \leq L \qquad \forall t \in \mathcal{T}, k \in K \tag{5}$$

$$I_{ip,t-1} + \sum_{k \in K} q_{iptk} \leq C_{ip} \qquad \forall i \in V, p \in P, t \in \mathcal{T} \backslash \{1\} \tag{6}$$

$$\mathcal{J}_{ip}^{0} + \sum_{k \in K} q_{ip1k} \leq C_{ip} \qquad \forall i \in V, p \in P \qquad (7)$$

$$\sum_{i,j \in S} x_{ijtk} \leq |S| - 1 \qquad \forall t \in \mathcal{T}, k \in K, S \subseteq V, |S| \geq 2 \qquad (8)$$

$$q_{iptk} \leq C_{ip} y_{iptk} \qquad \forall i \in V, p \in P, t \in \mathcal{T}, k \in K \qquad (9)$$

$$y_{iptk} \leq \sum_{j \in V_0} x_{jitk} \qquad \forall i \in V, p \in P, t \in \mathcal{T}, k \in K \qquad (10)$$

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

$$q_{iptk} \geq 0 \qquad \forall i \in V, p \in P, t \in \mathcal{T}, k \in K \qquad (12)$$

$$x_{ijtk} \in \{0, 1\} \qquad \forall i, j \in V_0, t \in \mathcal{T}, k \in K \qquad (13)$$

$$I_{ipt} \geq 0 \qquad \forall i \in V, p \in P, t \in \mathcal{T} \qquad (14)$$

$$y_{iptk} \in \{0, 1\} \qquad \forall i \in V, p \in P, t \in \mathcal{T}, k \in K \qquad (15)$$

The objective function (1) aims to minimize the total transportation cost. Constraints (2-3) are the inventory flow balance constraints at the customer locations. Constraints (4) do not allow compartments to be loaded with more than their capacities. Constraints (5) restrain the length of the routes by L. Customers' storage capacity for each product is ensured by constraints (6-7). Constraints (8) are the subtour elimination constraints. Constraints (9-10) determine whether a delivery is made to a customer on a route. Constraints (11) ensure that each vehicle visiting a customer should leave that customer. The remaining constraints are the domain constraints specifying the values that decision variables can take.

If we assume that each customer can be visited once for each product on a day, the following constraints should be added to the model.

$$\sum_{k \in K} y_{iptk} \leq 1 \qquad \forall i \in V, p \in P, t \in \mathcal{T} \qquad (16)$$

If we assume that each customer can only be visited by one vehicle on a day, the

following decision variable and constraints need to be added to the model.

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

$$q_{iptk} \leq C_{ip} z_{itk} \qquad \forall i \in V, p \in P, t \in \mathcal{T}, k \in K \qquad (17)$$

$$\sum_{k \in K} z_{itk} \leq 1 \qquad \forall i \in V, t \in \mathcal{T} \qquad (18)$$

$$\sum_{j \in V_0} (x_{ijtk} + x_{jitk}) \leq 2 z_{itk} \qquad \forall i \in V, t \in \mathcal{T}, k \in K \qquad (19)$$

$$z_{itk} \in \{0, 1\} \qquad \forall i \in V, t \in \mathcal{T}, k \in K \qquad (20)$$

Constraints (17) determine whether a delivery is made to a customer on a route. Constraints (18) ensure that each customer can only be visited by one vehicle on a day. Constraints (19) determine whether a customer is visited by a vehicle on a day.

CHAPTER IV

SOLUTION APPROACH

To obtain high quality and efficient solutions to the proposed MC-IRP, we develop a matheuristic approach in which we integrate an Adaptive Large Neighborhood Search metaheuristic with mathematical programming models.

4.1 Finding an initial solution

To find a good initial solution for our matheuristic, we first generate a large number of routes regardless of distribution amounts. Then by using a route-based mathematical model, we determine which distribution routes to be used on each day as well as the distribution amounts of each product to customers in each selected route. Here, if all possible routes are generated and the route-based mathematical model can be solved, the problem can be solved to optimality. However, generating all possible routes is an exponential time operation depending on the number of customers, therefore it is not possible. For this reason, while finding an initial solution, all feasible routes (i.e. routes that are no longer than L) with single and two customers, together with the feasible routes with three to seven customers among 1000 randomly generated routes are included in the route set. When the customers to be visited in each route are known, the most cost-effective route is found by solving a Travelling Salesman Problem (TSP) visiting the depot and the customers in the route. To solve TSP, we use the Lin-Kernighan algorithm that is widely applied in the literature [73].

We use the following notation for the route-based mathematical model (MIP-RB).

$R =$ Set of routes generated

$R_i =$ Set of routes that visit customer $i \quad \forall i \in V$

$L_r =$ Length of the route $r \quad \forall r \in R$

$$\alpha_{ir} = \begin{cases} 1, & \textit{if} \text{ customer } i \text{ is visited by route } r \\ 0, & \textit{otherwise} \end{cases} \qquad \forall i \in V, r \in R$$

We present below the decision variables and the formulation of the MIP-RB assuming that each customer can only be visited by one vehicle on a day.

$$x_{rt} = \begin{cases} 1, & \textit{if} \text{ route } r \text{ is used on day } t \\ 0, & \textit{otherwise} \end{cases}$$

$q_{iprt} =$ Amount of product p delivered to customer i on route r on day t

$I_{ipt} =$ Amount of product p in the inventory at customer i on day t

$$\text{MIP-RB:} \quad \min \quad \sum_{t \in \mathcal{T}} \sum_{r \in R} L_r x_{rt} \tag{21}$$

$$st. \quad \sum_{r \in R^i} x_{rt} \leq 1 \qquad \forall i \in V, t \in \mathcal{T} \tag{22}$$

$$I_{ip,t-1} + \sum_{r \in R} q_{iprt} = I_{ipt} + d_{ipt} \qquad \forall i \in V, p \in P, t \in \mathcal{T} \backslash \{1\} \tag{23}$$

$$\mathcal{J}^0_{ip} + \sum_{r \in R} q_{ipr1} = I_{ip1} + d_{ip1} \qquad \forall i \in V, p \in P \tag{24}$$

$$I_{ip,t-1} + \sum_{r \in R} q_{iprt} \leq C_{ip} \qquad \forall i \in V, p \in P, t \in \mathcal{T} \backslash \{1\} \tag{25}$$

$$\mathcal{J}^0_{ip} + \sum_{r \in R} q_{ipr1} \leq C_{ip} \qquad \forall i \in V, p \in P \tag{26}$$

$$\sum_{i \in V} q_{iprt} \leq Q_p x_{rt} \qquad \forall p \in P, r \in R, t \in \mathcal{T} \tag{27}$$

$$\sum_{i \in V} \sum_{p \in P} \sum_{t \in \mathcal{T}} q_{iprt}(1 - \alpha_{ir}) = 0 \qquad \forall r \in R \tag{28}$$

$$q_{iprt} \geq 0 \qquad \forall i \in V, p \in P, r \in R, t \in \mathcal{T} \quad (29)$$

$$x_{rt} \in \{0,1\} \qquad \forall r \in R, t \in \mathcal{T} \quad (30)$$

$$I_{ipt} \geq 0 \qquad \forall i \in V, p \in P, t \in \mathcal{T} \quad (31)$$

The objective function of the MIP-RB (21) aims to minimize the total length/cost of the routes performed. Constraints (22) ensure that each customer can be visited at most once on each day. Constraints (23-24) are the inventory flow balance constraints for each product at the customer locations. Constraints (25-26) ensure that customers' storage capacities are not exceeded. Constraints (27) ensure that the capacities of compartments dedicated to each product are not exceeded. Constraints (28) guarantee that only visited customers on a tour can be served. The remaining constraints are the domain constraints specifying the values that decision variables can take.

If we assume that each customer can be visited once for each product on a day, we should introduce the y_{iprt} decision variable and add the following constraints to the model. And we also need to remove the constraints (22).

$$y_{iprt} = \begin{cases} 1, & \textit{if} \text{ customer } i \text{ is visited on day } t \text{ on route } r \text{ for product } p \\ 0, & \textit{otherwise} \end{cases}$$

$$q_{iprt} \leq Q_p y_{iprt} \qquad \forall i \in V, p \in P, r \in R, r \in T \quad (32)$$

$$\sum_{r \in R} y_{iprt} \leq 1 \qquad \forall i \in V, p \in P, t \in T \quad (33)$$

To adapt the problem so that each customer can be visited by more than one vehicle for any product on a day, we have to remove constraints (22).

4.2 Adaptive Large Neighborhood Search

Within the framework of our matheuristic, we use an Adaptive Large Neighborhood Search (ALNS) as the metaheuristic method. This matheuristic uses the solution

obtained by the MIP-RB as an initial solution. ALNS is first described in [74] and is a metaheuristic that employs several destroy, repair/improvement operators within a search. Each destroy operator has a weight assigned to them which affects their probability of being selected by using a roulette-wheel mechanism. These weights are updated at each segment depending on the operators' performance in the search process. In some ALNS applications in the literature [75], as in destroy operators, repair/improvement operators are also selected by using a roulette-wheel mechanism. However, similar to [67] and [76], we apply repair/improvement operators in a given order for every iteration.

4.2.1 Destroy operators

We adopt several neighborhoods that are used to destroy the current solution as solving different IRP variants in the literature [67, 76], which are (i) Randomly remove ρ visits: This operator removes a randomly selected customer from a randomly selected route on a randomly selected period. It is repeated ρ times; (ii) Randomly insert ρ visits: This operator randomly selects a period, a route performed during that period, and a customer that is not visited on that period. It inserts that customer into the selected route with the minimum cost increase. It is repeated ρ times; (iii) Remove worst ρ visits: The customer who will reduce the cost the most when removed from a route is removed from the route it belongs to. It is repeated ρ times; (iv) Insert best ρ visits: This operator calculates the cost increase that occurs as a result of inserting each customer to the routes on the periods they are not visited and the customer who causes the least cost increase among them is inserted to that route. It is repeated ρ times; (v) Shaw removal: Taking its name from [77], this operator randomly selects a period and a customer from a random route. The distance between the selected customer and the nearest customer that belongs to the same route is calculated. Then, the selected customer is taken as the center and all customers who are in an area

twice the diameter of this distance are removed from the route; (vi) Shaw insertion: Following the idea in (v), this operator selects the customers to be inserted based on proximity. It first selects a random route on a random period and a random customer that is not visited on that period by any vehicle. The distance between the selected customer and the nearest customer to it is calculated. Then, the selected customer is taken as the center and the customers who are not visited on that period and lie within an area twice the diameter of this distance are inserted into the selected route, by following the cheapest insertion rule; (vii) Remove ρ customers: This operator removes a randomly selected customer from all the periods it exists. It is repeated ρ times; (viii) Insert ρ customers: This operator inserts a randomly selected customer to all periods that it was not present before by using cheapest insertion. It is repeated ρ times; (ix) Empty one period: This operator randomly selects a period and removes all the routes performed during that period; (x) Swap routes: This operator randomly selects two periods and the routes employed during these periods are exchanged with each other; (xi) Randomly move ρ visits: This operator randomly selects a period, a route performed during that period, and a customer visited on that route. It removes the customer from that route and inserts it to another route performed on a different period with the minimum cost increase. It is repeated ρ times. Additionally, we employ (xii) Remove ρ routes: This operator randomly removes a route and is repeated ρ times; (xiii) Insert ρ routes: This operator randomly selects a route among the randomly generated route set which is used in MIP-RB. It randomly inserts the selected route to one of the periods that the customer/s in the selected route is/are not visited. It is repeated ρ times; (xiv): Divide ρ routes: This operator randomly selects a route and randomly divides it into two. It is repeated ρ times; (xv): Merge 2ρ routes: This operator randomly selects ρ route pairs. The selected route pairs are merged in the most cost-effective way by solving TSP. For the respective destroy operators, ρ takes a random integer value between [1,3].

The destroy operators above are explained under the assumption of each customer can only be visited by one vehicle on a day. To adapt the problem so that each customer can be visited once for each product on a day, we need to modify the application of the insertion operators (ii, iv, vi, viii, xi, xiii). Here, to be able to insert a customer to a route, rather than being sure about the customer to be inserted is not visited on that day, we need to be sure that the customer is visited less than the number of products on that day. Also to adapt the problem so that each customer can be visited by more than one vehicle for any product on a day, as in the previous assumption, we need to modify the application of the insertion operators. For this, while inserting a customer into a route, we need to check whether the customer is already visited on that route.

4.2.2 Repair operators

If the feasibility of the solution is ruined after it is destroyed by one of the operators above, we first apply the (i) Minimum ratio insert operator on the new solution to repair it. If it cannot make the solution feasible, upon we apply the (ii) Break routes operator.

The Minimum ratio insert operator tries to insert customers into the existing routes to make the solution feasible. For each customer with stock-out product(s) and for each route on the days it is not visited we calculate a ratio by dividing the cost increase of inserting the customer to that route, by the decrease in the stock-out amount. Within this operation we also consider inserting a customer to a new route with direct delivery to it. For each product that a customer is out of stock, the decrease in the stock-out amount is equal to the minimum of the following two; (i) the unused capacity in the compartment dedicated for that product and (ii) the stock-out amount in that product. The customer-route pair with the lowest ratio is detected and that customer is inserted into that route. To calculate the distribution

quantities of the solution obtained after this process, we solve a linear programming model (LP-SO) that aims to minimize the total stock-out amount of the customers. If the objective value of this model is zero, it means that the solution is feasible and the operator is terminated; if it takes a value different than zero, the solution is still not feasible, and one of the customers with stock-out product(s) is tried to be inserted to a route by using the same method described above. The operator is terminated when the solution is feasible or if there is no route that we can insert the customer(s) with stock-out product(s) to.

We present below the LP-SO, where R_t denotes the set of routes that are performed on day t, β_{irt} is a binary parameter indicating whether customer i is visited by route r on day t, and S_{ipt} is a decision variable denoting the amount of stock-out customer i has in product p at the end of day t.

$$\text{LP-SO:} \quad \min \quad \sum_{i\in V}\sum_{p\in P}\sum_{t\in\mathcal{T}} S_{ipt} \tag{34}$$

$$I_{ip,t-1} + \sum_{r\in R_t} q_{iprt} + S_{ipt} = I_{ipt} + d_{ipt} \qquad \forall i \in V, p \in P, t \in \mathcal{T}\backslash\{1\} \tag{35}$$

$$\mathcal{J}^0_{ip} + \sum_{r\in R_1} q_{ipr1} + S_{ipt} = I_{ip1} + d_{ip1} \qquad \forall i \in V, p \in P \tag{36}$$

$$I_{ip,t-1} + \sum_{r\in R_t} q_{iprt} \leq C_{ip} \qquad \forall i \in V, p \in P, t \in \mathcal{T}\backslash\{1\} \tag{37}$$

$$\mathcal{J}^0_{ip} + \sum_{r\in R_1} q_{ipr1} \leq C_{ip} \qquad \forall i \in V, p \in P \tag{38}$$

$$\sum_{i\in V} q_{iprt} \leq Q_p \qquad \forall p \in P, r \in R_t, t \in \mathcal{T} \tag{39}$$

$$\sum_{i \in V} \sum_{p \in P} q_{iprt}(1 - \beta_{irt}) = 0 \qquad \forall r \in R_t, t \in \mathcal{T} \tag{40}$$

$$S_{ipt} \geq 0 \qquad \forall i \in V, p \in P, t \in \mathcal{T} \tag{41}$$

$$I_{ipt} \geq 0 \qquad \forall i \in V, p \in P, t \in \mathcal{T} \tag{42}$$

$$q_{iprt} \geq 0 \qquad \forall i \in V, p \in P, r \in R_t, t \in \mathcal{T} \tag{43}$$

The objective function of the LP-SO (34) aims to minimize the total amount of stock-out. Constraints (35-36) are the inventory flow balance constraints for each product at the customer locations. Constraints (37-38) ensure that customers' storage capacities are not exceeded. Constraints (39) ensure that the capacities of compartments dedicated to each product are not exceeded. Constraints (40) guarantee that only visited customers on a route can be served. The remaining constraints are the domain constraints specifying the values that decision variables can take. Since the routes to be performed for each day are known in this model, the problem turns into a simple linear programming model.

If we assume that each customer can be visited once for each product on a day, while applying the Minimum ratio insert operator, we consider inserting a customer to a day only if that customer is visited less than the number of products on that day. On the other hand, if we assume that customers can be visited by multiple vehicles for any product on a day, while applying the Minimum ratio insert operator, we consider inserting customers with stock-out product(s) to any day. For these two assumptions, there is no need to modify the LP-SO.

The Break routes operator tries to find a feasible solution by removing customers from the existing routes and making direct delivery to the removed customer on the

same day. Similar to the Minimum ratio insert operator, for each customer with stock-out product(s) and for each route it is visited we calculate a ratio by dividing the cost increase occurred by removing the customer from the route and inserting it to a direct delivery route, by the decrease in the amount of stock-out. We solve the LP-SO to calculate the decrease in the amount of stock-out and to see whether the solution is feasible. Among all the customer-route pairs for which the aforementioned rate is calculated, if there are pairs that make the solution feasible, the pair with the lowest ratio is selected and the customer is removed from the route, and direct delivery is made to it. If there is no feasible solution among the customer-route pairs, the pair with the lowest ratio is selected and the customer is removed from the route, and direct delivery is made to it. This process is repeated until a feasible solution is obtained. There is no need to make any changes on this operator while tailoring the problem for the second and the third delivery assumptions.

4.2.3 Improvement operators

When we obtain a feasible solution, either right after implementing a destroy operator or repairing the solution, we apply (i) Remove route and (ii) Remove customer operators respectively to improve the solution.

Remove route operator solves the MIP-RB by using a route set consisting of only the existing routes in the current solution. Considering that the model minimizes the total length of the routes used, it determines which routes to be employed among the current ones, and which routes to be removed without ruining the feasibility of the solution. To adapt this operator to be used for the second and the third delivery assumptions, we only need to make the essential modifications to the MIP-RB model (see §4.1).

Remove customer operator tries to remove customer(s) from the existing routes, thus decrease the transportation cost. Solving the LP-SO for each customer-route

pair to see whether any customer can be removed, would be too costly in terms of the computation time. Therefore, to obtain a list of promising customer-route pairs, we make a calculation for each pair to predict whether the solution will be feasible if the customer is removed from the route. To make this prediction, for each customer-route pair, in case the customer is removed from the route, we calculate the maximum inventory amount that can stem from the days before the route is performed. Besides, we calculate the minimum inventory amount required for the days after the route is performed. We calculate these maximum and minimum inventory amounts over the unused capacities in the compartments. We subtract the minimum inventory amount and the customer's demand on the day it is removed from the maximum inventory amount. If this calculation is non-negative, we add this customer-route pair to the list of promising pairs. After we make this calculation for all the customer-route pairs, we solve the LP-SO for all the promising pairs. If there are any feasible solutions, we select the pair that provides the highest cost decrease and terminate the operator after removing that customer from that route. If there isn't any feasible solution, the operator is terminated without any improvements. There is no need to make any changes on this operator while tailoring the problem for the second and the third delivery assumptions.

4.2.4 Algorithm structure

Within the algorithm, we use the LP-SO model after every destroy operator, within the repair operators, and within an improvement operator (i.e., Remove customer). LP-SO model is a linear programming model, hence using this model frequently does not create a significant computational burden on the algorithm. However, for the second delivery assumption where each customer can be visited once for each product on a day, the LP-SO model may fail to ensure this constraint. In other words, even when a product is distributed to a customer by more than one vehicle on a day, the

objective value of the LP-SO may occur as zero. In LP-SO, to ensure feasibility for the second assumption, we have to add a binary variable to this model which converts it to a mixed-integer programming (MIP) model. We prefer to avoid this conversion, since it is a frequently used model and using a MIP model too often would create a significant burden on the algorithm. Instead, for the second delivery assumption, we create a fixing procedure to use right after the LP-SO model. When the objective value of the LP-SO model occurs as zero, we examine the solution and see whether there exists any delivery that violates the feasibility for the second assumption and if it does, we apply the fixing procedure. This procedure aims to combine all deliveries of a product to a customer in a single vehicle. We first try to combine the deliveries in an occupied vehicle. For all the vehicles visiting that customer, we look at the amount of unused space in the capacity dedicated to the respective product; and if there is enough space in a vehicle, we combine the deliveries in there. If fixing the solution with this method is not possible, we then try to combine the deliveries in a new vehicle (i.e., making direct delivery to the customer). If adding a new route to the solution does not cause the customer to be visited by more than the number of products, we add that route and combine the deliveries in there; however if it does, then we remove that customer from its route and insert it to a new route and combine the deliveries in there. This fixing procedure may not guarantee feasibility in all cases. If we cannot fix the solution with this procedure, we assume that the solution is infeasible and continue to our algorithm.

At each iteration, we use roulette wheel selection to determine which destroy operator to apply. In roulette wheel selection, the probability of destroy operator i being selected in segment j (p_{ij}) is directly proportional to its weight (w_{ij}) and is determined by using the formula (44). Here, I denotes the set of destroy operators.

$$p_{ij} = \frac{w_{ij}}{\sum_{i \in I} w_{ij}} \tag{44}$$

In each segment consisting of a certain number of iterations, the weight of any destroy operator is updated with the formula (45), depending on the status of the solution reached.

$$w_{ij+1} = w_{ij}(1 - r) + r\frac{\pi_i}{\theta_i} \tag{45}$$

Here, π_i denotes the score that the destroy operator i has collected in a segment. At the end of each iteration, if the new solution has a lower objective value than the best solution found, then σ is added to the π value of the destroy operator used. θ_i denotes the number of times the destroy operator i is used in a segment. r is the reaction factor and indicates in what portion the previous weight of the destroy operator i will participate in the new weight ($0 \leq r \leq 1$). All π_i values are set to zero at the start of each segment. Based on the well-performing ALNS applications in the literature [67, 76], we set r to 0.7.

After calculating the weights of all destroy operators at the end of each segment, the probability of operators being selected is calculated based on these weights. Initially, the weights are taken as zero and probabilities are equal for all operators. As the algorithm proceeds, the probability of operators being selected may converge to zero, and in this case they are very unlikely to be used again. We use a second roulette wheel to give these operators another chance. While the first roulette wheel described above is used to select one of all the destroy operators with 90% probability, the second roulette wheel is used with 10% probability to select a destroy operator among the ones whose probability of being selected has dropped below 0.1%.

The steps of our ALNS algorithm are presented in Algorithm 1. It uses the solution obtained by the MIP-RB as an initial solution. Here, $z(\cdot)$ represents the objective function value of a given solution, while R and I represent repair and improvement operators.

Some ALNS studies in the literature use the Simulated Annealing (SA) algorithm to specify the acceptance and stopping criterion [66, 67, 74, 76]. It is usually preferred

Algorithm 1: *Our ALNS algorithm for the MC-IRP*

All weights are set to 1 and all scores are set to 0.
s_{best} = Initial solution
$iterations = 0$
$time = 0$
while $iterations \leq 5000$ and $time \leq 7200$ **do**
  $k = 1$
  while $k \leq K$ **do**
    Select a destroy operator ($d \in D$) using the roulette-wheel mechanism based on the current weights.
    $\theta(d) = \theta(d) + 1$
    $s' = d(s_{best})$
    $s' =$ LP-SO(s')
    if s' is not feasible **then**
      $s'' = R(s')$
      $s''' = I(s'')$
    else
      $s''' = I(s')$
    end if
    if $z(s''') \leq z(s_{best})$ **then**
      $s_{best} = s'''$
      $\pi(d) = \pi(d) + \sigma$
    end if
    $k = k + 1$
    $iteration = iteration + 1$
  end while
  update the weights of all operators and reset their scores.
end while
return s_{best}

since the use of SA provides an additional diversification effect. Depending on the initial temperature and cooling coefficient parameters in the SA, the probability of accepting the current solution in any iteration is known. The stopping criterion can be determined as a certain minimum temperature when using SA, as well as a maximum number of iterations or a time limit. Initial temperature is reduced by the cooling coefficient in each iteration, so the time passed to reach the minimum temperature depends on these parameters. We make experiments in which the acceptance and stopping criterion are applied as in the SA algorithm and examine the results in §5.3. Since the algorithm performs better without the use of SA, we avoid using SA in the ALNS and accept only the solutions which improve the best solution.

CHAPTER V

COMPUTATIONAL STUDY

5.1 Generating test instances

We generate our test instances by adapting from [29]. We create our instances on a 100x100 map, where the center of the map (origin point) represents the depot. In every two instances, the number of customers is increased by five up until 50 customers, afterwards it is increased by ten up until 100 customers. We use Euclidean distance to calculate the distance between any nodes. We consider three different product types and different from [29], a planning horizon of 3, 5, 7 days. We generate the demands (d_{ipt}) randomly for each customer, product, and day, so that with 40% probability it falls within the [1,2] range, with 40% probability it falls within [2,3], and with 20% probability it falls within [3,6]. We set the storage capacity (C_{ip}) of the customers as 40 tons for each customer-product pair. In a dataset, the initial inventories meeting the majority of customer demands can simplify the problem. To prevent this, different than the range used in [29], we generate the initial inventories (J^0_{ip}) randomly within the [0,3] range for each customer-product pair. [29] limits the number of customers that can be visited on a route by three. In our problem, such a restriction corresponds to the maximum route length (L), which is set to 300, so that a vehicle dispatched from the depot can at most visit the perimeter of half of the map area. Since more than three customers can be visited within a distance of 300 units, we set a higher value for the capacities of each compartment (Q_p), which is 12. We run the proposed ALNS algorithm in the data sets described above and perform numerical analysis on a system with a 2.80 GHz i7-7700HQ processor and 16GB RAM. We code our algorithm in Python and use CPLEX 12.8 with default

parameter settings.

5.2 Experiments with using Simulated Annealing

As mentioned in §4.2.4, before we bring our ALNS to its final form (Algorithm 1), to comply with the ALNS applications in the literature, we try integrating the SA method, which allows accepting worse solutions to some extent and thus diversifying the searching process. We specifically compare three versions of our algorithm to determine whether to use SA in ALNS, and how. In the following, we first explain the details about these versions and then present the results of this comparison.

Remember that in our ALNS application, π_i denotes the score that the destroy operator i has collected in a segment and each time the best solution is improved after applying a destroy operator, the relevant π_i is increased by σ. When using SA in the algorithm, the acceptance criterion is not just whether the best solution is improved, but also whether the current solution is improved and whether the SA acceptance criterion is met given the current temperature even when no improvement is present.

When using SA in the ALNS, at the end of each iteration, by looking at the quality of the solution, one of the values $\sigma_1 > \sigma_2 > \sigma_3$ can be added to the π value of the destroy operator used. If the new solution at the end of the iteration has a lower objective value than the best solution found, then σ_1; if it has an objective value less than the objective value of the current solution, then σ_2; if it cannot improve the solution in any way but it meets the SA acceptance criterion given the current temperature, then σ_3 is added to the π value. If none of these three conditions are met, no point is added in that iteration. Based on the well-performing ALNS studies in the literature [67, 76], we set the $\sigma_1, \sigma_2, \sigma_3$ values to 10, 5, 2 respectively.

SA employs several parameters such as initial temperature, cooling coefficient, and maximum iteration number. To determine the parameter values to be used in our algorithm, we make a preliminary experiment where we test various parameter

combinations, starting with the values used in [66]. With this parameter setting, we observe that the number of iterations completed falls behind the maximum iteration number within a tolerable run time and this situation affects the performance of the algorithm negatively. Based on our observations, it is due to the trade-off between reaching a better solution and the time required to reach a better solution in a matheuristic method. As a result of our preliminary experiment, we set the maximum iteration number to 5000, segment size to 100, initial temperature to 2726, minimum temperature to 0.01, and cooling coefficient to 0.9975. We take all the runs within a two-hour time limit. During our analysis with the large-scale instances, we observe that in some cases 5000 iterations cannot be completed, therefore the current temperature falls significantly far from the minimum temperature towards the end of the algorithm. This situation yields the algorithm to continue accepting the worse solutions with a high probability towards the end. To prevent this, one method could be increasing the time limit, but in that case we had to increase it more for the larger scale instances and the time required could reach some unreasonable amount. Alternatively, we come up with a method which we name as the Adaptive Cooling Schedule (ACS). In ACS, depending on how fast the algorithm can complete a segment, the value of the cooling coefficient is tuned. At the end of each segment, we average the time passed in all previous iterations. With the average iteration time, we try to predict how long the remaining iterations will take, and accordingly when the algorithm will end. If the estimated time is less than or equal to the remaining time, no change is necessary, if it is greater, a faster cooling is tried to be achieved in the SA by changing the current cooling coefficient.

The first version of our algorithm (ALNS-1) uses SA to determine which solutions to be accepted at each iteration and uses the parameter settings and the ACS method explained above. The psuedocode for the ALNS-1 is provided as Algorithm 2 in Appendix A. The second version (ALNS-2) is created by forcing the ALNS-1 to start

each segment with the best solution found so far. The psuedocode for the ALNS-2 is provided as Algorithm 3 in Appendix A. Finally, the third version (ALNS-3) does not use SA at all, meaning that it only accepts solutions that improve the best solution. We compare these three versions for the 3-day planning horizon and for the first delivery assumption by using seven different instances which vary in size from 20 to 50 customers. Table 1 presents the percentage deviation of the solutions found by each ALNS version from the best solution found for each instance. ALNS-3 finds the best solution among all versions for the six of the instances, and it is the least (only 0.05%) deviant from the best solution on average. Therefore, we conclude that allowing to accept worse solutions with the use of SA does not help us explore valuable neighborhoods, therefore we decide not to use SA within our algorithm.

Instance		**ALNS**		
#	**n**	**ALNS-1**	**ALNS-2**	**ALNS-3**
1	**20**	1.96	0.00	0.00
2	**25**	1.06	0.00	0.32
3	**30**	1.60	4.46	0.00
4	**35**	3.76	2.11	0.00
5	**40**	8.11	1.88	0.00
6	**45**	9.31	1.84	0.00
7	**50**	2.95	2.54	0.00
Average		**4.11**	**1.83**	**0.05**

Table 1: Percentage deviation of each result from the best solution found for each instance

5.3 Finding lower bounds

To find effective lower bounds that we can compare against our solutions obtained in the large-scale instances, we use the flow formulation proposed by [78]. We modify the proposed formulation based on our three assumptions in terms of to what extent split delivery is allowed. There is no vehicle/route index in this formulation. Therefore, the solution found by the flow formulation may not be feasible for our problem as

it cannot cope with the following two situations: (i) Failure to ensure the maximum route length (L) constraint, (ii) Under the second and the third delivery assumption, the amount of product carried on the incoming edge of a customer may be more than the amount of product carried on the outgoing edge of that customer. Such an incident infers that the customer is loading some product on the vehicle and it is not possible within the scope of our problem. We use the following sets and decision variables within the flow formulation we refer to find tight lower bounds.

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

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

$$x_{ijt} = \begin{cases} 1, & \textit{if} \text{ edge } (i,j) \text{ is used to travel from node } i \text{ to node } j \text{ on day } t \\ 0, & \textit{otherwise} \end{cases}$$

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

$y_{ijt} =$ Number of times the (i,j) edge is used on day $t \quad \forall (i,j) \in E, t \in \mathcal{T}$

$$z_{it} = \begin{cases} 1, & \textit{if} \text{ node } i \text{ is visited on day } t \\ 0, & \textit{otherwise} \end{cases} \qquad \forall i \in V, t \in \mathcal{T}$$

$l_{ijpt} =$ The amount of product p loaded on the vehicle that traverses from node i to node j on day $t \quad \forall (i,j) \in V_0, p \in P, t \in \mathcal{T}$

$q_{ipt} =$ The amount of product p delivered to customer i on day t

$\forall i \in V, p \in P, t \in \mathcal{T}$

$I_{ipt} =$ The amount of inventory at customer i in product p at the end of day t

$\forall i \in V, p \in P, t \in \mathcal{T}$

We model the flow formulation as follows under the first delivery assumption, where each customer can only be visited by one vehicle on a day.

MIP-F: $$\min \quad \sum_{(i,j)\in E} \sum_{t\in\mathcal{T}} b_{ij} x_{ijt} \tag{46}$$

$$\mathcal{J}^0_{ip} + q_{ip1} = I_{ip1} + d_{ip1} \qquad \forall i \in V, p \in P \tag{47}$$

$$I_{ip,t-1} + q_{ipt} = I_{ipt} + d_{ipt} \qquad \forall i \in V, p \in P, t \in \mathcal{T}\backslash\{1\} \tag{48}$$

$$q_{ip1} \leq C_{ip} - \mathcal{J}^0_{ip} \qquad \forall i \in V, p \in P \tag{49}$$

$$q_{ipt} \leq C_{ip} - I_{ip,t-1} \qquad \forall i \in V, p \in P, t \in \mathcal{T}\backslash\{1\} \tag{50}$$

$$q_{ipt} \leq C_{ip} z_{it} \qquad \forall i \in V, p \in P, t \in \mathcal{T} \tag{51}$$

$$\sum_{j:(i,j)\in E} y_{ijt} + \sum_{j:(j,i)\in E} y_{jit} = 2z_{it} \qquad \forall i \in V, t \in \mathcal{T} \tag{52}$$

$$\sum_{j|(i,j)\in A} l_{ijpt} - \sum_{j|(j,i)\in A} l_{jipt} = \begin{cases} -q_{ipt} & \text{if } i \in V \\ \sum\limits_{i\in V} q_{ipt} & \text{if } i = 0 \end{cases} \qquad \forall i \in V_0, p \in P, t \in \mathcal{T} \tag{53}$$

$$l_{ijpt} \leq Q_p x_{ijt} \qquad \forall (i,j) \in A, p \in P, t \in \mathcal{T} \tag{54}$$

$$\sum_{j|(j,i)\in A} x_{jit} - \sum_{j|(i,j)\in A} x_{ijt} = 0 \qquad \forall i \in V_0, t \in \mathcal{T} \tag{55}$$

$$x_{ijt} + x_{jit} = y_{ijt} \qquad \forall (i,j) \in E, t \in \mathcal{T} \tag{56}$$

$$y_{ijt} \in \{0,1\} \qquad \forall (i,j) \in E \backslash \{i,j=0\}, t \in \mathcal{T} \tag{57}$$

$$y_{0jt} \in \{0,1,2\} \qquad \forall j \in V, t \in \mathcal{T} \tag{58}$$

$$x_{ijt} \in \{0,1\} \qquad \forall (i,j) \in A, t \in \mathcal{T} \tag{59}$$

$$z_{it} \in \{0,1\} \qquad \forall i \in V, t \in \mathcal{T} \tag{60}$$

$$l_{ijpt} \geq 0 \qquad \forall i,j \in V_0, p \in P, t \in \mathcal{T} \tag{61}$$

$$q_{ipt} \geq 0 \qquad \forall i \in V_0, p \in P, t \in \mathcal{T} \tag{62}$$

$$I_{ipt} \geq 0 \qquad \forall i \in V_0, p \in P, t \in \mathcal{T} \tag{63}$$

The objective function (46) minimizes the total cost of the edges used. Constraints (47- 48) are the inventory flow balance constraints at customers for all products. Constraints (49- 50) ensure that storage capacities at customers for each product are not exceeded. Constraints (51) determine whether a distribution is made to a customer. In the first assumption, since a customer can be visited once a day, constraints (52) and (55-56) ensure that when a customer is visited, there must be a total of two edges (inbound and outbound) that pass through that node. Constraints (53) ensure that each product carried in a vehicle within a day is reduced by the amount distributed as customers are visited. Constraints (54) ensure that if a customer is visited, the amount of each product carried in the vehicle visiting that customer can at most be as much as the capacity dedicated to those products. Constraints (57-63) determine the values that decision variables can take.

For the second delivery assumption, where each customer can be visited once for each product on a day, the domains of the variables z_{it}, y_{ijt}, x_{ijt} are modified so that they can take any integer value between 0 and m. And, the domain of the y_{0jt} variables is modified so that they can take any integer value between 0 and $2m$.

For the third delivery assumption, where a product is allowed to be distributed to a customer by more than one vehicle on a day, the domains of the variables z_{it}, y_{ijt}, x_{ijt} and y_{0jt} in the model are modified so that they can take any non-negative integer value.

5.4 Numerical results

We evaluate the performance of our matheuristic approach by comparing it with the lower bounds obtained by the MIP-F. We run both our algorithm and the MIP-F model for two hours for every instance. Besides the time limit, we also set a maximum iteration number as our termination criteria, which is 5000. We compare the objective values found by our matheuristic against the lower bounds reported in CPLEX after solving the MIP-F model. We report the percentage deviations from the lower bounds found by solving MIP-F with CPLEX in separate tables (Tables 2, 3, 4) for each delivery assumption. We run each test instance within a 3-, 5-, and 7-day planning horizon for each delivery assumption. In Tables 2, 3 and 4, n represents the number of customers in an instance. The first of the two results under each planning horizon is the average of the runs taken with two different seeds (Avg.), whereas the second means the best of these two runs (Best).

While the average of the "Avg." columns is 13%, the average of the "Best" columns is 11.7%. In other words, our algorithm finds solutions that are 13% higher in cost on average in respect to the lower bound found by solving MIP-F using CPLEX. If we calculate the best performances among the different seeds, this difference drops off to 11.7%.

Instance		ALNS					
		First Delivery Assumption					
		T = 3		T = 5		T = 7	
#	n	Avg.	Best	Avg.	Best	Avg.	Best
1	**20**	7.51	7.51	2.29	2.29	7.14	6.88
2	**20**	11.55	10.02	6.98	6.98	7.67	7.04
3	**25**	10.10	9.26	8.87	8.07	7.13	6.94
4	**25**	12.41	12.21	4.55	4.54	7.04	6.23
5	**30**	9.64	9.11	8.94	8.12	7.57	7.26
6	**30**	14.27	13.41	6.54	5.89	5.03	4.21
7	**35**	11.20	10.69	8.29	7.94	6.94	6.39
8	**35**	13.90	13.86	8.19	7.54	8.75	8.24
9	**40**	11.68	11.08	10.49	8.99	7.91	7.57
10	**40**	16.51	16.24	7.61	7.54	8.56	7.89
11	**45**	12.20	11.71	6.82	6.78	11.49	11.46
12	**45**	11.60	9.49	7.36	7.17	8.48	7.12
13	**50**	11.44	9.10	10.26	8.79	7.66	7.36
14	**50**	19.03	17.73	10.15	10.03	7.97	7.74
15	**60**	20.48	19.86	15.62	15.54	9.84	9.41
16	**60**	16.87	16.76	13.22	11.69	11.19	10.76
17	**70**	21.01	20.83	14.66	14.61	12.79	11.69
18	**70**	20.03	19.83	16.45	15.44	13.61	12.94
19	**80**	21.89	19.07	15.92	15.46	13.71	13.07
20	**80**	24.30	24.24	13.49	12.31	16.00	14.16
21	**90**	26.73	23.55	20.30	19.15	17.75	17.60
22	**90**	23.82	19.71	19.33	17.99	15.97	15.82
23	**100**	27.78	25.31	19.36	15.36	14.56	14.11
24	**100**	29.50	28.12	17.96	16.66	18.94	15.51
Avg.		**16.89**	**15.78**	**11.40**	**10.62**	**10.57**	**9.89**

Table 2: Under the first delivery assumption, percentage deviation of the solutions found by our algorithm from the lower bound found by MIP-F using CPLEX

Instance		ALNS					
		Second Delivery Assumption					
		T = 3		T = 5		T = 7	
#	n	Avg.	Best	Avg.	Best	Avg.	Best
1	**20**	14.15	14.15	4.21	3.72	7.20	7.10
2	**20**	11.50	10.13	9.60	8.99	6.37	6.18
3	**25**	22.12	21.41	10.00	9.83	7.21	6.69
4	**25**	16.10	11.14	7.34	6.80	7.08	6.40
5	**30**	12.44	10.61	11.49	11.45	7.88	7.14
6	**30**	18.20	17.73	7.75	7.22	6.66	6.29
7	**35**	15.55	12.51	11.73	9.70	7.03	6.85
8	**35**	20.37	18.82	10.33	8.89	11.35	7.74
9	**40**	12.61	10.89	11.10	10.33	9.25	8.10
10	**40**	20.59	16.28	9.32	8.55	9.50	8.46
11	**45**	13.78	12.33	8.96	8.11	12.56	10.19
12	**45**	11.40	9.96	10.20	7.99	9.39	7.46
13	**50**	15.23	14.52	11.96	11.05	9.92	8.52
14	**50**	19.55	19.13	13.41	12.38	14.46	10.94
15	**60**	24.26	22.28	15.49	13.94	15.04	11.73
16	**60**	20.29	15.74	15.97	12.36	12.97	10.16
17	**70**	26.81	24.02	17.57	15.27	18.95	16.85
18	**70**	25.29	24.60	18.49	15.18	18.55	13.43
19	**80**	32.35	29.19	16.24	14.72	26.79	19.72
20	**80**	27.67	21.38	19.33	12.69	18.07	15.94
21	**90**	35.71	30.90	20.64	20.21	19.94	16.46
22	**90**	26.79	22.76	27.45	26.52	19.70	18.73
23	**100**	36.75	30.17	18.81	17.04	21.56	15.38
24	**100**	33.34	27.95	26.45	18.66	23.78	18.23
Avg.		**21.37**	**18.69**	**13.91**	**12.15**	**13.39**	**11.03**

Table 3: Under the second delivery assumption, percentage deviation of the solutions found by our algorithm from the lower bound found by MIP-F using CPLEX

Instance		ALNS					
		Third Delivery Assumption					
		T = 3		T = 5		T = 7	
#	n	Avg.	Best	Avg.	Best	Avg.	Best
1	**20**	5.65	5.65	2.16	2.16	6.10	6.10
2	**20**	6.43	5.81	4.26	4.02	6.21	5.89
3	**25**	8.08	7.35	7.28	7.10	5.92	5.90
4	**25**	5.85	5.79	5.98	5.82	5.75	4.79
5	**30**	6.09	5.62	6.92	5.92	5.40	5.35
6	**30**	6.22	6.01	6.52	5.76	4.46	4.15
7	**35**	8.10	7.19	8.95	8.84	5.87	5.44
8	**35**	10.81	9.64	8.41	7.63	6.38	6.27
9	**40**	6.15	5.60	6.96	6.53	7.52	7.48
10	**40**	8.94	7.17	4.75	4.60	6.12	5.35
11	**45**	9.91	6.71	7.64	7.63	8.66	8.39
12	**45**	10.31	8.02	7.27	7.18	6.45	6.14
13	**50**	7.89	7.51	8.00	7.81	6.90	6.42
14	**50**	10.15	9.17	9.13	9.08	7.61	7.16
15	**60**	16.67	13.44	10.31	9.65	13.88	13.26
16	**60**	12.07	11.93	9.17	8.48	10.03	9.01
17	**70**	15.79	13.78	12.43	11.33	12.95	11.85
18	**70**	18.35	16.87	11.39	11.18	11.52	10.69
19	**80**	16.03	15.89	12.68	11.60	9.30	8.58
20	**80**	16.77	13.56	13.88	12.20	12.23	10.35
21	**90**	15.53	13.49	15.12	14.05	13.57	12.89
22	**90**	16.43	15.82	11.99	11.48	13.93	13.66
23	**100**	18.81	14.19	14.70	12.89	16.22	11.37
24	**100**	15.80	15.07	12.98	11.22	9.68	9.35
Avg.		**11.37**	**10.05**	**9.12**	**8.51**	**8.86**	**8.16**

Table 4: Under the third delivery assumption, percentage deviation of the solutions found by our algorithm from the lower bound found by MIP-F using CPLEX

As for computation times, for the instances with up to 30 customers our algorithm can terminate before the two-hour time limit. For the first delivery assumption, our algorithm terminates in 4,304 seconds at the earliest (with 20 customers). For the second delivery assumption, our algorithm terminates in 6,718 seconds at the earliest (with 20 customers). And for the third delivery assumption, our algorithm terminates in 3,910 seconds at the earliest (with 25 customers).

We observe that for all the delivery assumptions, as the size of the instance increases, the deviation of the results obtained by ALNS from the lower bounds found by solving MIP-F increases as well. However, it should be noted that the reason for these high values is mainly that the lower bounds obtained by MIP-F are rather loose in large-scale instances. Although the optimal solution of the MIP-F theoretically gives a lower bound for the MC-IRP, we were able to use the lower bounds obtained at the end of 2 hours to evaluate the performance of the ALNS, since no optimal solution could be found for the MIP-F at the end of 2 hours. And this lower bound gets looser as the size of the instance increases. To express it numerically, for the first delivery assumption, while the average optimality gap of the MIP-F is 10.4% for the instances with ≤ 50 customers, for the instances with > 50 customers, the average optimality gap is 24.6%. For the second delivery assumption, while the average optimality gap of the MIP-F is 7.2% for the instances with ≤ 50 customers, for the instances with > 50 customers, the average optimality gap is 48.9%. For the third delivery assumption, while the average optimality gap of the MIP-F is 9.4% for the instances with ≤ 50 customers, for the instances with > 50 customers, the average optimality gap is 44.5%.

We also measure how much we improve our initial solution by using our matheuristic approach. For the first delivery assumption, when we take the average of two seeds we see that we improve the solution by 5.4% on average; whereas if we consider the

best of two seeds, we improve the solution by 6.1%. For the second delivery assumption, when we take the average of two seeds we see that we improve the solution by 10.1% on average; whereas if we consider the best of two seeds, we improve the solution by 16.1%. And for the third delivery assumption, when we take the average of two seeds we see that we improve the solution by 4.3% on average; whereas if we consider the best of two seeds, we improve the solution by 5.4%.

CHAPTER VI

CONCLUSION

In this study, we analyze the distribution of multiple products to customers on a multi-period basis by using compartmentalized vehicles. This problem is known as the Multi-Compartment Inventory Routing Problem (MC-IRP) in the literature. Using separate compartments for each product type allows the supplier to distribute assorted products to its customers on a delivery route. Thus, the supplier can better coordinate the distribution and inventory management to avoid additional routes and achieve lower transportation costs. On the other hand, supplier's distribution options vary considerably, hence a high number of feasible solutions emerge and it makes the problem more difficult than the classical inventory routing problem.

We assume that compartments are dedicated to products and the capacities of compartments are fixed, as in real-life operations such as transportation of foods under different storage conditions, collection of recyclable goods, and feed distribution to livestock farms. Customers have certain storage capacities for each product. We limit the maximum route length by a certain distance to mimic the restriction on the working times of the drivers. Our aim is to determine the delivery amounts of each product to each customer per route per day, the amount of inventory to be held at each customer for each product per day, and the vehicle routes navigating between customers and the depot to minimize the total transportation cost of the routes performed over the planning horizon.

We develop a matheuristic approach to address the difficulties of the problem in an effective and efficient way. Here we use an Adaptive Large Neighborhood Search algorithm that systematically incorporates mathematical programming models. To

find an initial feasible solution to our algorithm we formulate a mathematical model which determines the routes to be performed on each day among a set of routes that we generate. We create test instances with up to 100 customers and a 7-day planning horizon and run our algorithm under three assumptions in terms of to what extent split delivery to customers is allowed. To measure the performance of our algorithm, we adapt a mathematical model (MIP-F) that uses flow formulation for routing and distribution purposes. We solve MIP-F using the CPLEX solver and report the lower bound found by the solver. If we compare the lower bound found by solving MIP-F with our algorithm, we observe that our algorithm finds solutions with 13% higher cost on average; and only with 11.7% higher cost if we compare the best solutions found in different runs.

This problem can be further elaborated by considering different compartment structures, such as allowing to configure the capacity of each compartment discretely (e.g. using separators to increase/decrease the capacities in terms of unit capacity). To the best of our knowledge there exists no study in the literature that studies MC-IRP with this assumption. Moreover, one can also examine this problem by incorporating the assignment decision of products to compartments, which have fixed capacities. Although there exist such studies in the literature, the performance of the solution approach may be tested on more compelling test instances and the assumptions/constraints that reduce the difficulty of the problem may be avoided.

APPENDIX A

PSEUDOCODES OF ALNS-1 AND ALNS-2

Algorithm 2: *ALNS-1*

All weights are set to 1 and all scores are set to 0.
s, s_{best} = Initial solution
$iterations = 0$
$time = 0$
while $iterations \leq 5000$ and $time \leq 7200$ **do**
  $k = 1$
  while $k \leq K$ **do**
    Select a destroy operator ($d \in D$) using the roulette-wheel mechanism based on the current weights.
    $\theta(d) = \theta(d) + 1$
    $s' = d(s)$
    $s' =$ LP-SO(s')
    if s' is not feasible **then**
      $s'' = R(s')$
      $s''' = I(s'')$
    else
      $s''' = I(s')$
    end if
    if $z(s''') \leq z(s_{best})$ **then**
      $s_{best} = s'''$
      $\pi(d) = \pi(d) + \sigma_1$
    else if $z(s''') \leq z(s)$ **then**
      $s = s'''$
      $\pi(d) = \pi(d) + \sigma_2$
    else if s''' is accepted by the SA criterion **then**
      $s = s'''$
      $\pi(d) = \pi(d) + \sigma_3$
    end if
    $k = k + 1$
    $iteration = iteration + 1$
  end while
  update the weights of all operators and reset their scores.
end while
return s_{best}

Algorithm 3: *ALNS-2*

All weights are set to 1 and all scores are set to 0.
s, s_{best} = Initial solution
$iterations = 0$
$time = 0$
while $iterations \leq 5000$ and $time \leq 7200$ **do**
  $k = 1$
  while $k \leq K$ **do**
    Select a destroy operator ($d \in D$) using the roulette-wheel mechanism based on the current weights.
    $\theta(d) = \theta(d) + 1$
    $s' = d(s)$
    s' = LP-SO(s')
    if s' is not feasible **then**
      $s'' = R(s')$
      $s''' = I(s'')$
    else
      $s''' = I(s')$
    end if
    if $z(s''') \leq z(s_{best})$ **then**
      $s_{best} = s'''$
      $\pi(d) = \pi(d) + \sigma_1$
    else if $z(s''') \leq z(s)$ **then**
      $s = s'''$
      $\pi(d) = \pi(d) + \sigma_2$
    else if s''' is accepted by the SA criterion **then**
      $s = s'''$
      $\pi(d) = \pi(d) + \sigma_3$
    end if
    $k = k + 1$
    $iteration = iteration + 1$
  end while
  $s = s*$
  update the weights of all operators and reset their scores.
end while
return s_{best}

Bibliography

[1] J. Bramel and D. Simchi-Levi, "On the effectiveness of set covering formulations for the vehicle routing problem with time windows," *Operations Research*, vol. 45, no. 2, pp. 295–301, 1997.

[2] A. Campbell, L. Clarke, A. Kleywegt, and M. Savelsbergh, *The inventory routing problem.*, 1998. *Fleet Management and Logistics*, T.G. Crainic and G. Laporte, Eds., Kluwer Academic Publishers 95–112.

[3] A. Campbell and M. Savelsbergh, "A decomposition approach for the inventory-routing problem," *Transportation Science*, vol. 38, no. 4, pp. 488–502, 2004a.

[4] A. Campbell and M. Savelsbergh, "Efficient insertion heuristics for vehicle routing and scheduling problems," *Transportation Science*, vol. 38, no. 3, pp. 369–378, 2004b.

[5] V. Gaur and M. Fisher, "A periodic inventory routing problem at a supermarket chain," *Operations Research*, vol. 52, no. 6, pp. 813–822, 2004.

[6] C. Archetti, L. Bertazzi, G. Laporte, and M. Speranza, "A branch-and-cut algorithm for a vendor-managed inventory routing problem," *Transportation Science*, vol. 41, no. 3, pp. 382–391, 2007.

[7] C. Archetti, L. Bertazzi, A. Hertz, and M. Speranza, "A hybrid heuristic for an inventory routing problem," *INFORMS Journal on Computing*, vol. 24, no. 1, pp. 101–116, 2012.

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

[9] L. Coelho and G. Laporte, "The exact solution of several classes of inventory-routing problems," *Computers & Operations Research*, vol. 40, no. 2, pp. 558–565, 2013.

[10] O. Solyali and H. Sural, "A branch-and-cut algorithm using a strong formulation and an a priori tour-based heuristic for an inventory-routing problem," *Transportation Science*, vol. 45, no. 3, pp. 335–345, 2011.

[11] L. Bertazzi, L. Chan, and M. Speranza, "Analysis of practical policies for a single link distribution system," *Naval Research Logistics*, vol. 54, no. 5, pp. 497–509, 2007.

[12] L. Chan, A. Federgruen, and D. Simchi-Levi, "Probabilistic analyses and practical algorithms for inventory-routing models," *Operations Research*, vol. 46, no. 1, pp. 96–106, 1998.

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

[14] M. Reed, A. Yiannakou, and R. Evering, "An ant colony algorithm for the multi-compartment vehicle routing problem," *Applied Soft Computing*, vol. 15, pp. 169–176, 2014.

[15] A. El-Fallahi, C. Prins, and R. Calvo, "A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem," *Computers and Operations Research*, vol. 35, no. 5, pp. 1725–1741, 2008.

[16] J. Chen, B. Dan, and J. Shi, "A variable neighborhood search approach for the multi-compartment vehicle routing problem with time windows considering carbon emission," *Journal of Cleaner Production*, vol. 277, 2020.

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

[18] 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.

[19] M. Muyldermans and G. Pang, "On the benefits of co-collection: Experiments with a multi-compartment vehicle routing algorithm," *European Journal of Operational Research*, vol. 206, no. 1, pp. 93–103, 2010.

[20] 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.

[21] J. Mendoza, B. Castanier, C. Guéret, A. 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.

[22] E. Chajakis and M. Guignard, "Scheduling deliveries in vehicles with multiple compartments," *Journal of Global Optimization*, vol. 26, no. 1, pp. 43–78, 2003.

[23] 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.

[24] L. 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.

[25] F. Cornillier, F. Boctor, G. Laporte, and J. Renaud, "An exact algorithm for the petrol station replenishment problem," *Journal of the Operational Research Society*, vol. 59, no. 5, pp. 607–615, 2008a.

[26] F. Cornillier, 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. 2295–2305, 2008b.

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

[28] 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.

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

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

[31] F. Cornillier, F. Boctor, and J. Renaud, "Petrol delivery tanker assignment and routing: a case study in hong kong," *Journal of the Operational Research Society*, vol. 59, no. 9, pp. 1991–1200, 2008.

[32] W. Chowmali and S. Sukto, "A novel two-phase approach for solving the multi-compartment vehicle routing problem with a heterogeneous fleet of vehicles: a case study on fuel delivery," *Decision Science Letters*, vol. 9, no. 1, pp. 77–90, 2020.

[33] T. Henke, G. Speranza, and G. Wäscher, "The multi-compartment vehicle routing problem with flexible compartment sizes," *European Journal of Operational Research*, vol. 246, no. 3, pp. 730–743, 2015.

[34] H. Koch, T. Henke, and G. Wascher, "A genetic algorithm for the multicompartment vehicle routing problem with flexible compartment sizes," *Working Paper*, Otto-von-Guericke-University Magdeburg, 2016.

[35] D. Jansen, G. V. D. Vorst, and A. V. Weert, "Multi compartment distribution in the catering supply chain," *International Transaction in Operations Research*, vol. 5, no. 6, pp. 509–517, 1998.

[36] U. Derigs, J. Gottlieb, J. Kalkoff, M. Piesche, F. Rothlauf, and U. Vogel, "Vehicle routing with compartments: Applications, modelling and heuristics," *Operations Research Spectrum*, vol. 33, no. 4, pp. 885–914, 2011.

[37] M. Caramia and F. Guerriero, "A milk collection problem with incompatibility constraints," *Interfaces*, vol. 40, no. 2, pp. 130–143, 2010.

[38] K. Sethanan and R. Pitakaso, "Differential evolution algorithms for scheduling raw milk transportation," *Computers and Electronics in Agriculture*, vol. 121, pp. 245–259, 2016.

[39] J. Melechovsky, "A variable neighborhood search for the selective multi-compartment vehicle routing problem with time windows," *Lecture notes in management science*, vol. 5, pp. 159–166, 2013.

[40] M. Christiansen, K. Fagerholt, T. Flatberg, O. Haugen, O. Kloster, and E. 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.

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

[42] D. Ronen, "Marine inventory routing: Shipments planning," *Journal of the Operational Research Society*, vol. 53, no. 1, pp. 108–114, 2002.

[43] S. Hwang, "Inventory constrained maritime routing and scheduling for multi-commodity liquid bulk," *Ph.D. Thesis*, Georgia Institute of Technology, Atlanta, 2005.

[44] K. Fagerholt and M. Christiansen, "A combined ship scheduling and allocation problem," *Journal of the Operational Research Society*, vol. 51, no. 7, pp. 834–842, 2000.

[45] A. Agra, M. Christiansen, and A. Delgado, "Mixed integer formulations for a short sea fuel oil distribution problem," *Transportation Science*, vol. 47, no. 1, pp. 108–124, 2013.

[46] A. Agra, M. Christiansen, A. Delgado, and L. Simonetti, "Hybrid heuristics for a short sea inventory routing problem," *European Journal of Operational Research*, vol. 236, no. 3, pp. 924–935, 2014.

[47] A. Agra, M. Christiansen, A. Delgado, and L. Hvattum, "A maritime inventory routing problem with stochastic sailing and port times," *Computers and Operations Research*, vol. 61, pp. 18–30, 2015.

[48] B. Santosa, R. Damayanti, and B. Sarkar, "Solving multi-product inventory ship routing with a heterogeneous fleet model using a hybrid cross entropy-genetic algorithm: a case study in indonesia," *Production & Manufacturing Research*, vol. 4, no. 1, pp. 90–113, 2016.

[49] M. Christiansen and K. Fagerholt, *Maritime inventory routing problems.*, 2009. *Encyclopedia of Optimization*, C.A. Floudas and P.M. Pardalos, Eds., second ed. Springer-Verlag 1947–1955.

[50] S. Misra, M. Kapadi, and R. Gudi, "A multi grid discrete time based framework for maritime distribution logistics & inventory planning for refinery products," *Computers & Industrial Engineering*, vol. 146, 2020.

[51] J. Goodson, "A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands," *European Journal of Operational Research*, vol. 241, no. 2, pp. 361–369, 2015.

[52] P. Kabcome and T. Mouktonglang, "Vehicle routing problem for multiple product types, compartments, and trips with soft time windows," *International Journal of Mathematics and Mathematical Sciences*, vol. 2015, 2015.

[53] M. Ostermeier and A. Hübner, "Vehicle selection for a multi-compartment vehicle routing problem," *European Journal of Operational Research*, vol. 269, no. 2, pp. 682–694, 2018.

[54] 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.

[55] G. Brown and G. Graves, "Real-time dispatch of petroleum tank trucks," *Management Science*, vol. 27, no. 1, pp. 19–32, 1981.

[56] G. Brown, G. Graves, and D. Ronen, "Scheduling ocean transportation of crude oil," *Management Science*, vol. 33, no. 3, pp. 335–346, 1987.

[57] D. Bausch, G. Brown, and D. Ronen, "Scheduling short-term marine transport of bulk products," *Maritime Policy and Management*, vol. 25, no. 4, pp. 335–348, 1998.

[58] M. Coccola, C. Mendez, and R. Dondo, "A two-stage procedure for efficiently solving the integrated problem of production, inventory, and distribution of industrial products," *Computers & Chemical Engineering*, vol. 134, 2020.

[59] T. Henke, G. Speranza, and G. Wäscher, "A branch-and-cut algorithm for the multi-compartment vehicle routing problem with flexible compartment sizes," *Annals of Operations Research*, vol. 275, no. 2, pp. 321–338, 2019.

[60] 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.

[61] K. Hessler, *Exact algorithms for the multi-compartment vehicle routing problem with flexible compartment sizes*, 2020. Gutenberg School of Management and Economics & Research Unit "Interdisciplinary Public Policy" Discussion Paper Series, Discussion paper number 2007.

[62] R. Lahyani, L. Coelho, M. Khemakhem, G. Laporte, and F. Semet, "A multicompartment vehicle routing problem arising in the collection of olive oil in tunisia," *Omega*, vol. 51, no. 1, pp. 1–10, 2015.

[63] I. Surjandari, A. Rachman, F. Dianawati, and R. Wibowo, "Petrol delivery assignment with multi-product, multi depot, split deliveries and time windows," *International Journal of Modeling and Optimization*, vol. 1, no. 1, pp. 375–379, 2011.

[64] Y. Hsu, J. Walteros, and R. Batta, "Solving the petroleum replenishment and routing problem with variable demands and time windows," *Annals of Operations Research*, vol. 1, pp. 1–38, 2018.

[65] 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.

[66] L. Coelho, J. Cordeau, and G. Laporte, "Consistency in multivehicle inventory-routing," *Transportation Research Part C: Emerging Technologies*, vol. 24, no. 1, pp. 270–287, 2012a.

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

[68] K. Doerner and V. Schmid, *Matheuristics for rich vehicle routing problems*, 2010. Hybrid metaheuristics. Lecture notes in computer science. *Springer* **6373** 206–221.

[69] T. Guimaraes, L. Coelho, C. Schenekemberg, and C. Scarpin, "The two echelon multi-depot inventory-routing problem," *Computers & Operations Research*, vol. 101, pp. 220–233, 2019.

[70] J. Lysgaard, A. Lopez-Sanchez, and A. Hernandez-Diaz, "A matheuristic for the minmax capacitated open vehicle routing problem," *International Transactions in Operational Research*, vol. 27, no. 1, pp. 394—-417, 2020.

[71] Q. Yu, K. Fang, N. Zhu, and S. Ma, "A matheuristic approach to the orienteering problem with service time dependent profits," *European Journal of Operational Research*, vol. 273, no. 2, pp. 488–503, 2019.

[72] J. Mendoza, B. Castanier, C. Guéret, A. Medaglia, and N. Velasco, "Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands," *Transportation Science*, vol. 45, no. 3, pp. 346–363, 2011.

[73] K. Helsgaun, "General k-opt submoves for the lin-kernighan tsp heuristic," *Mathematical Programming Computation*, vol. 1, no. 2–3, pp. 119–163, 2009.

[74] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.

[75] D. Pisinger and S. Ropke, "A general heuristic for vehicle routing problems," *Computers & Operations Research*, vol. 34, no. 8, pp. 2403–2435, 2007.

[76] D. Aksen, O. Kaya, F. Salman, and O. Tuncel, "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.

[77] P. Shaw, "A new local search algorithm providing high quality solutions to vehicle routing problems," technical report, University of Strathclyde, Glasgow, 1997.

[78] C. Archetti, N. Bianchessi, S. Irnich, and M. Speranza, "Formulations for an inventory routing problem," *International Transactions in Operational Research*, vol. 21, pp. 353–374, 2014.

VITA

Ceren Gültekin graduated from Adem Tolunay Anatolian High School in 2013. She earned her B.Sc. degree in Industrial Engineering from Özyeğin University in June 2018. She has been pursuing her M.Sc. degree in Industrial Engineering at Özyeğin University since September 2018, under the supervision of Dr. Okan Örsan Özener and Dr. Ali Ekici. Her research interests include supply chain management and logistics and she is eager to explore various branches of operations research and operations management.