EXPLORING METAHEURISTIC ALGORITHMS AND SOLVING PHASES FOR ADDRESSING A RICH VRP ENCOUNTERED BY A TURKISH DISTRIBUTOR

A Thesis

by

Ahmad Bassaleh

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 2024

EXPLORING METAHEURISTIC ALGORITHMS AND SOLVING PHASES FOR ADDRESSING A RICH VRP ENCOUNTERED BY A TURKISH DISTRIBUTOR

Advisor: Prof. Dr. Ekrem Duman, Özyeğin University
Coadvisor: Dr. Mehmet Bayram Yıldırım, Wichita State University

Approved by:

Prof. Dr. Ekrem Duman, Advisor
Department of Industrial Engineering
Özyeğin University

Prof. Dr. Burcu Balçık
Department of Industrial Engineering
Özyeğin University

Prof. Dr. Ali Fuat Alkaya
Department of Computer Engineering
Marmara University

Date Approved: May 22, 2024

To my beloved family and my people.

ABSTRACT

Logistical challenges pose a substantial financial burden for distributors worldwide, particularly in light of escalating oil prices and evolving customer demands. This thesis addresses a Rich Vehicle Routing Problem (RVRP) encountered by a distributor in Turkey. The problem involves an objective function that integrates road tariffs, specific customer constraints, and features observed in various VRP variants. Our study explores diverse solving approaches, including an exact method developed through a novel Mixed-Integer Linear Program, as well as non-exact solutions employing heuristic and metaheuristic algorithms. Notably, our findings highlight the efficacy of metaheuristic algorithms, particularly through a two-phase solving strategy: initially dividing and solving the problem into individual sub-problems, then integrating them comprehensively. The study concluded with a 5.13% reduction in operational costs compared to the distributor's existing plan, indicating significant potential for enhanced profitability while optimizing the daily scheduling of the vehicle fleet, and further expanding applications in operations research.

Rich Vehicle Routing problem, VRP variants, Real-life application, Metaheuristics, MILP model

ÖZETÇE

Lojistik zorluklar; özellikle artan petrol fiyatları ve değişen müşteri talepleri ışığında, küresel distribütörler için önemli bir malî yük oluşturmaktadır. Bu tez, Türkiye'deki bir distribütörün karşılaştığı Zengin Araç Rotalama Problemini (Rich Vehicle Routing Problem) ele almaktadır. Problem; otoyol ücretlerini, belirli müşteri kısıtlarını ve çeşitli VRP varyantlarında gözlemlenen özellikleri entegre eden çok amaçlı bir fonksiyon içermektedir. Çalışmamız, yeni bir Karma Tamsayılı Doğrusal Program (Mixed-Integer Linear program) aracılığıyla geliştirilen kesin (exact) bir yöntemin yanı sıra sezgisel (heuristic) ve metasezgisel (metaheuristic) algoritmalar kullanan kesin olmayan çözümler de dahil olmak üzere çeşitli çözüm yaklaşımlarını da sunmaktadır. Bulgularımız, özellikle iki aşamalı bir çözüm stratejisi aracılığıyla metasezgisel algoritmaların etkinliğini vurgulamaktadır: Başlangıçta problemi ayrı ayrı alt problemlere bölmek ve çözmek, ardından da bunları kapsamlı bir şekilde bütünleştirmek. İşbu çalışma, distribütörün mevcut planına kıyasla operasyonel maliyetlerde %5,13 azalma ile sonuçlanmış olup, araç filosunun günlük planlamasını optimize ederken kârlılığı artırmak ve yöneylem araştırmasındaki (Operations Research) uygulamaları daha da genişletmek için önemli bir potansiyele işaret etmektedir.

ACKNOWLEDGEMENTS

Deep gratitude goes to Professor Ekrem Duman, my advisor, whose unwavering guidance and support were instrumental throughout my graduate journey. His profound knowledge and encouragement fueled my academic pursuits. I owe much of my ambition and resilience to his mentorship. His support and wisdom were invaluable, shaping both my academic path and personal growth.

I extend my gratitude to every member of the Industrial Engineering department at Ozyegin university for generously sharing their expertise, both in and out of the classroom, which has contributed significantly to my academic foundation. Among them, I want to express a special thanks to Professor Burcu Balcik, whose passion for teaching ignited my initial interest in pursuing an academic career. Professor Balcik's enthusiastic approach captivates students effortlessly, reflecting her genuine love for her work. Her dedication is truly inspiring, and I aspire to follow in her footsteps as a professor one day.

I am also grateful to my fellow college graduate students, specially, Ahmed Aljabi, Eihab Ahmed, Kian Farajkhah, and Ulvi Findik for fostering a supportive and collaborative environment throughout my studies.

Finally, I extend heartfelt gratitude to my parents and siblings for being my unwavering support system through both triumphs and challenges. I am also deeply appreciative of the steadfast friendship and encouragement from Mustafa Awad, Omar Ahmad, Omar Al-teneiji, and Bethany Anne.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Logistic problems are at the heart of efficient supply chain management, encompassing the planning and control of the goods, services, and related information flows. These problems arise across various industries, including manufacturing, retail, transportation, and healthcare. Logistical planning significantly contributes to the total expenses for distributors, highlighting the importance of opting for well planned routes. A study carried out by Harp [1] reveals that companies with well-organized supply chain planning experience a 38% increase in profitability compared to the industry average. The expenses incurred in transportation are influenced by various factors, including oil prices, vehicle maintenance costs (both fixed and variable), tolls for road usage, expenses related to loading and unloading goods, as well as penalties incurred for failing to meet customers' demands punctually. Given the rapid increase in oil prices, the importance of optimizing route planning becomes even more pronounced.

Logistical challenges and other industry-related problems are often modeled as optimization problems. This approach harnesses the flexibility and efficiency inherent in optimization techniques, resulting in notable achievements [2; 3]. A prime example illustrating a logistical planning scenario translated to an optimization problem is the Traveling Salesman Problem (TSP). In this problem, a salesman must visit a set of cities exactly once and return to the starting city. Different sequencing of visits the travel salesman take leads to different distances that must be crossed. The solution exhibiting a sequence of visits with the minimum distance crossed is recognized as the optimal solution. While the TSP may seem straightforward to comprehend, it is

classified under nondeterministic polynomial-time hard (NP-hard) problems [4]. NP-hard problems like the TSP pose significant computational challenges, as the required amount of time needed to optimally solve them grows exponentially as the problem size increases, making the search for exact solutions impractical for large problem instances. Consequently, addressing such instances often involves employing approximation algorithms or heuristic/Metaheuristic approaches to find *good* solutions within reasonable time frames.

Another optimization problem that deals with logistical problems is the Vehicle Routing Problem (VRP). In the classical VRP, a set of customers located in different regions place orders for delivery from a central depot belonging to a supplier. The delivery of these orders is expected to be executed through the utilization of a fleet of vehicles. Each laden vehicle is anticipated to depart from the central depot, systematically deliver each order to its designated location, and subsequently return to the depot. The total demand of all the customers on a route has to comply with the vehicle capacity. The primary goal is to optimize the routes of the vehicles to minimize the overall cost, which may involve factors such as the distance travelled, fleet size utilized, or total time taken.

The problem was initially introduced by Dantzig and Ramser in 1959, under the name of the 'Truck Dispatching Problem'. Clarke and Wright later expanded the scope of the problem into an optimization problem, thus popularizing its well-known name, the VRP [5]. The VRP arises in numerous practical contexts, understandably making it one of the most well-known, and extensively studied problems within combinatorial optimization [6]. This extensive exploration has also been instrumental in giving rise to numerous variants of the problem [7].

Fundamentally, the VRP extends the principles of the TSP. When the number of vehicles available to serve the set of customers reduces to only one, the problem simplifies into solving a TSP [8]. Moreover, given its acknowledgment as an NP-hard

problem, the TSP serves as a computational benchmark for the VRP along with its variants, recognizing them as NP-hard problems as well [9].

While the current literature presents various variants of the VRP, they typically concentrate on addressing singular or limited aspects. Examples include the Heterogeneous VRP (HVRP) [10], Open VRP (OVRP) [11], and their combination, HOVRP [12]. However, such approaches often fall short in addressing the complexities encountered by distributors in real-world scenarios. Additionally, scholars have observed that variants lacking comprehensiveness have been extensively investigated [13]. Consequently, in recent years, a new category of problems known as real-life or Rich VRP (RVRP) has emerged to bridge the divide between theoretical frameworks and practical applications. In RVRPs, the integration of multiple features enhances their relevance to real-world business challenges.

In this study, we tackle a novel VRP with a special cost structure and a sophisticated set of constraints, encountered by a distribution company in Turkey. The problem combines certain settings and constraints seen in numerous VRP variants, in addition to its unique characteristics. We introduce a unified mathematical model formulated as a Mixed-Integer Linear program, designed to address multiple VRP variants as well as other logistical constraints.

The distributor faces mounting operational costs due to rising oil and vehicle rental prices. Our primary goal is to develop solution methods specifically tailored to overcome the logistical challenges encountered, thereby increasing the distributor's profitability and extending the applicability of RVRP concepts within our research context.

The contributions of this thesis are as follows: 1) A novel logistical problem is introduced under the scope of RVRP, 2) A comprehensive mathematical model that accurately represents the problem by integrating multiple factors and VRP variants is formulated 3) A new metaheuristic algorithm that leverages the strengths of two

existing algorithms developed.

The outline of this thesis report is as follows: Chapter 2 provides an in-depth description of the problem under consideration. Following this, Chapter 3 delves into a comprehensive review of related literature. Chapter 4 outlines our proposed solution approach, encompassing the formulation of the problem as a mathematical model, in addition to a heuristic and three metaheuristic algorithms employed in this study. Moving forward, Chapter 5 offers details of the problem set addressed, along with two merging step undertaken to simplify the problem's complexity. Next, Chapter 6 presents the results obtained from numerical experiments, fostering a discussion and analysis of outcomes. Finally, Chapter 7 serves as the conclusion of the thesis study, summarizing key findings and delineating potential avenues for future research exploration.

CHAPTER II

PROBLEM DESCRIPTION

A distributor is responsible for delivering a diverse range of food products to retail stores located across numerous cities in Turkey. The products consist of small-sized items, packaged in sealed batches. As the products are consumable, some of them are susceptible to perishing and therefore require storage at a cooled temperatures to maintain their quality. Given this consideration, we will refer to the products requiring refrigeration as 'fridge' products, while those not requiring this aspect will be labeled as 'normal'. In addition, each product requested occupies a certain capacity, consisting of weight and volume.

To meet the demands of the customers, the distributor utilizes a fleet of heterogeneous vehicles. This fleet is not privately owned, rather completely outsourced from a common carrier. It is prearranged and made available at the distributor's depot location at the beginning of the day, from where all vehicles commence their routes. Upon completing the delivery to their designated customers, the vehicles are not obligated to return to the depot. At this point, the carrier, as the owner of the fleet, retains control over their next destination or resting place.

The fleet offered by the third-party consists of four types, namely: Truck (T), Refrigerated Truck (RT), Trailer truck (TT), and Refrigerated trailer truck (RTT). The refrigerated type of vehicles offer a temperature-controlled environment which facilitates the delivery of perishable 'fridge' products. Vehicles of the trailer type possess a larger cargo capacity. However, trailers impose limitations on accessing certain customer locations due to narrow roads that are unsuitable for vehicle passage. The carrier is presumed to have an unlimited supply of vehicles for each type at their

disposal.

Each type has a specific upper limit on the total weight and volume it can carry. To put the volume capacity of the vehicles into perspective, products are initially arranged on an set of pallets identical in size, before they are loaded. This systematic approach ensures optimal space utilization and facilitates organized and secure transportation of the goods. Notably, the maximum volume capacity of each vehicle equates to the maximum number of pallets it can carry. To translate the volume requirement of the products into pallet capacity, every single one is assigned an estimated proportion of a pallet it needs to occupy. It is assumed that a single product cannot surpass the volume of a pallet.

Each pallet can only contain products belonging to the same customer. This restriction is established for three reasons. The first one is potential damages (leakage or compression due to stacking) of a customer's products does not impact the requests of other customers. The second rationale is that it streamlines the unloading process upon arrival at customer locations. Moreover, pallets are initially loaded according to the sequence of visits the vehicle will undertake, with the pallets of the final customers positioned at the deepest point of the vehicle storage, and those of the initial customers placed nearest to the unloading doors. Having the products belonging to different customers mixed up in different pallets during loading complicates the unloading process. Lastly, it is founded on the customer's preference for maintaining privacy. In line with customers' privacy policies, a competitive atmosphere exists among certain customers, resulting in a preference for not sharing delivery vehicles with perceived competitors.

Upon a vehicle's visit to a customer, a designated stop is executed. This stopping process encompasses the unloading of products at the corresponding customer location. Given the time-intensive nature of this operation, a limitation is imposed on the maximum number of stops each vehicle can undertake, which is set at six customer

visits for all the vehicle types. This constraint is established to ensure fair working hours for the vehicle's personnel and to ensure timely arrival within the customers' designated receiving window (usually between 8am-6pm).

From the customers' perspective, they expect minimal number of visits from the distributor when delivering their products having the same type (normal or fridge), given the time and effort required by their personnel to help unload and store the products. This preference restricts the splitting of products belonging to the same customer and having the same type among different vehicles, despite the potential cost savings. However, for a subset of customers, the delivery of their products on more than one vehicle is permitted. This allowance applies to customers who have requested multiple products, with their total storage requirement surpassing the weight or/and pallet capacity of the largest vehicle type capable of serving them. Furthermore, upon receiving a customer's product requests, the total capacity requirement for each product type they request is calculated. This total capacity is then divided by the capacity of the largest vehicle capable of delivering to their location. Below is a simple equation outlining the calculations required to determine the number of splits among the vehicles that are permitted to visit each customer based on their product requests.

$$\text{Split allowed} = \max\left(\frac{\sum_i pallets_i}{Palletcapacity}, \frac{\sum_i weight_i}{Weightcapacity}\right)$$

The output is then rounded up to the nearest integer, providing a limitation on the maximum number of vehicles allowed to deliver each type of product to each customer.

Given that a common carrier fulfills the transportation requirements, they establish a distinctive pricing policy, consisting of three component per vehicle utilized. We introduce these components under the name of: 1) Tabulated cost, 2) Extra travel cost, and 3) Extra stops cost. Below, is a description of how each of these costs accumulates.

1. Tabulated cost: Each vehicle incurs a specific cost for servicing each customer. This cost is determined by the locations of the customers and varies depending on the type of vehicle used. Various factors contribute to this expense, including the distance to the customer and the tariff(s) that must be paid along the way. Once the assignment of customers to the vehicle is complete, the cost of serving the customer with the highest expense is identified as the tabulated cost of the vehicle. The tabulated cost represents the fixed cost associated with utilizing each vehicle, as it is incurred every time any vehicle is used.

2. Extra travel cost: The total distance traveled by each vehicle is computed, beginning from the depot and concluding at the location of the last visited customer. An extra cost is incurred based on the additional distance covered by each vehicle. Extra distance is defined as every kilometer beyond the necessary distance to reach the farthest customer from the depot, served by the vehicle. Finally, the extra travel cost is determined by multiplying the extra distance traveled by each vehicle with its specific cost per kilometer. The extra travel cost is considered a variable cost for each vehicle, as it fluctuates based on the distance traveled by each vehicle.

3. Extra stops cost: Whenever a vehicle is requested from the common carrier, a driver and unloading personnel are included as part of the service. The greater the number of stops the vehicle must make for unloading, the more work the personnel must undertake. To address this, the carrier imposes a limit on the number of stops each vehicle can make. If this limit is exceeded, an additional cost is incurred for each extra stop made. The Extra stop cost is regarded as a variable expense for each vehicle, as it varies depending on the number of stops made by each vehicle.

Figure 1 includes a basic illustration of how the common carrier charges for each vehicle utilized by the distributor. In this example, a vehicle is assigned to visit four customers. The dark node represents the depot, and the white dots represent the customers. The leftmost frame has a cost (in Turkish lira) of visiting each customer,

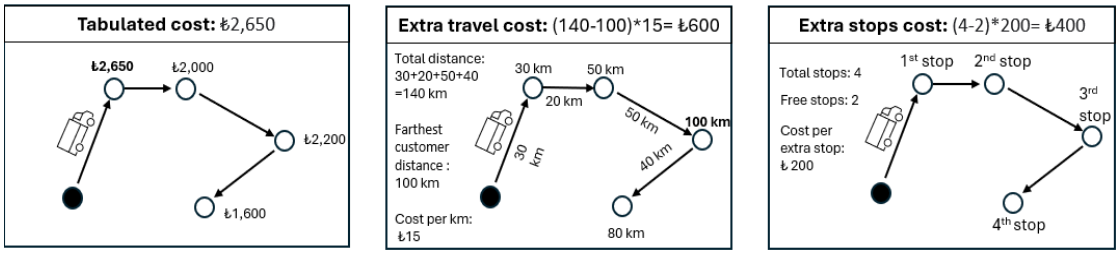

Figure 1: An example of the cost calculation for a single vehicle

noted next to their node. In the middle frame, the length of each arc (in kilometers) is displayed adjacent to it. The total distance traveled by the vehicle is determined by summing all the arc lengths. Each customer node has a distance next to them, representing their direct distance from the depot, and the 'Cost per km' is the cost of traversing each kilometer. In the last frame, the 'Total stops' defines how many customer visits the vehicle has made. The 'Free stops' is how many stops the carrier allows without charging extra, and the cost per extra stop is how much it costs to make extra visits by the vehicle.

To conclude this chapter, we provide a classification of the core elements of the problem, aiming to encompass all their aspects, including products, customers, vehicles, routes, and their associated costs.

- Capacitated Vehicles (**CV**): Each vehicle is restricted by its maximum weight and the number of pallets it can accommodate.
- Heterogeneous Fleet (**HF**): Our fleet comprises four types (T, RT, TT, RTT), each possessing distinct aspects.
- Loading Policy (**LP**): Products must be palletized correctly. Each pallet contains only items belonging to the same customer.
- Unlimited fleet (**UF**): The number of available vehicles of each type is assumed to be unlimited.

- Vehicle Site-Dependence (**SD**): 1) Certain customer sites are inaccessible to trailer-type vehicles due to geographical limitations. 2) Customers with fridge product (perishable) needs cannot be served by vehicles lacking refrigeration capabilities.

- Intra-route regulations (**IR**): 1) Each driver on a route is limited to a maximum number of customer site visitations. 2) Certain customers have preferences to not be served alongside particular others on the same route.

- Multiple Visits (**MV**): Customers with demands exceeding the capacity of a single vehicle are served by permitting multiple vehicles to visit their site.

- Open Routes (**O**): The routes start at the depot but finish on the last customer.

- Multi-component objective (**MO**): The objective function comprises three components that need simultaneous optimization: a fixed cost (Tabulated cost), and two variable costs (Extra travel cost and Extra stops cost).

CHAPTER III

LITERATURE REVIEW

Operational research (OR) has achieved significant success by tackling vehicle routing problems, solidifying its acknowledgement within the field. Since its first appearance, VRPs have been extensively researched, resulting in the production of a considerable body of scientific literature consisting of thousands of publications. Our problem integrates features and constraints extracted from diverse problems documented in VRP literature. Some features have become so prevalent across various problem settings that they have warranted the establishment of distinct variants within the VRP framework.

In this chapter, we describe key variants of the VRP relevant to our context, drawn from a variety of literature studies, and explaining their applicability to our specific problem. Subsequently, we highlight several papers that simultaneously explore multiple VRP variants similar to our problem.

3.1 Related VRP variants

One elementary variant of the VRP is the Capacitated VRP (CVRP), where each vehicle is assigned a capacity constraint, such as weight, volume, or the number of pallets it can accommodate [14]. Similarly, each type of our vehicles posses a limitation on the weight and number of pallets that it can take.

The Heterogeneous VRP (HVRP) differs in the sense that the fleet have distinguishable characteristics, such as varying capacities and distinct fixed or per-unit distance costs [10]. In resemblance to HVRP, the Fleet Size and Mix VRP (FSMVRP) similarly entails having a mixed fleet. However the number of vehicles is assumed to be unlimited [6]. Our problem similarly assumes the availability of an unlimited

heterogeneous fleet of vehicles with varying characteristics, including differences in capacity and cost structure.

The Site-Dependent VRP (SDVRP) introduces a relationship between vehicle types and customers, wherein not all vehicle types can serve all customers due to site-specific limitations such as road infrastructure, or incompatibility between requests needed and vehicle characteristics [15]. This characteristic appears twice in our scenario: firstly, in the limitation on trailer-type vehicles, determining which customer sites they can access; secondly, in the restriction preventing normal-type vehicles from delivering products to customers requiring temperature-controlled (fridge) products.

In the Open VRP (OVRP), vehicles are not required to return to the depot after completing service for the last customer on a route [16]. This scenario generally arise when the supplier does not own some or all of the vehicle fleet, rather they outsource transportation services from a common carrier. likewise, in our problem, routes are exclusively constructed from the depot to the final customer that must be visited on each route.

In the classical VRP, every customer is assigned to a single vehicle, which then fulfills all of that customer's demands. However, in scenarios where customers can be visited multiple times, a variant of the VRP emerges, known as the Split-Delivery VRP (VRPSD) [17]. This characteristic applies to the subset of customers in our problem who request products that, in terms of total capacity, necessitate the use of more than one vehicle, thus more visitations to the customer.

The final variant under consideration involves VRPs that incorporate regulations concerning the vehicle drivers. These regulations may include limits on maximum driving hours or restrictions on the number of unloading stops permitted. This variant is commonly referred to as the Driver and VRP (DVRP) [18], is apparent in our scenario, where the drivers assigned to each route are restricted to a maximum of six customer site visitations.

3.2 Related literature

A survey on the Rich Vehicle Routing Problem conducted by Caceres et al. in 2014 [6] noted the absence of specific defining features that distinguish the RVRP from other variants of the VRP. The RVRP encompasses a wide range of characteristics typically found in real-life vehicle-routing distribution systems, incorporating single or multiple objective functions, stochastic behavior, diverse constraints, and customer policies. Essentially, any VRP that integrates a complex set of constraints drawn from various VRP variants to address real-world challenges is considered an RVRP. It is often viewed as a generalization of other distinct problems documented in the literature, each of which poses significant challenges within the field of operations research. As the complexity of RVRPs necessitates sophisticated solutions, there has been an increasing focus on the advancement of computational methods. In recent years, heightened attention has been directed towards the RVRP variant, largely due to the emergence of methods capable of effectively tackling its intricate challenges [19].

Pellegrini [20] presents one of the earliest instances of the RVRP. The study involves a heterogeneous fleet, multiple time windows, delivery constraints within specified intervals, and a maximum time limit for each tour. The author proposes two heuristic algorithms based on the well-known Nearest Neighbour (NN) heuristic procedure introduced by Solomon [21], along with a swap local search. These algorithms include a Deterministic version of NN (DNN) and a Randomized NN (RNN) version, where randomness is incorporated into the selection of the next customer during route construction. The study demonstrates promising results within a short computational time across instances ranging between 50-200 customers. While the RNN algorithm outperforms the DNN version, its efficiency diminishes with an increasing number of customers. Addressing capacity restrictions, time windows, and a heterogeneous fleet with varying travel times, as well as multiple pickup and delivery

locations, travel costs, different start and end locations for vehicles, and other constraints, Goel and Gruhn [22; 23] propose iterative improvement approaches based on a Large Neighborhood Search (LNS). They develop an instance generator featuring 50-500 orders to showcase the performance of their approach. In addition, Goel [24] consider various real-life constraints, including time window restrictions, a heterogeneous vehicle fleet with different travel times and costs, multidimensional capacity constraints, order/vehicle compatibility constraints, orders with multiple pickup, delivery, and service locations, different start and end locations for vehicles, and route restrictions. They suggest an iterative improvement approach employing a reduced Variable Neighborhood Search (VNS) algorithm for exchanging elements between neighborhoods, alongside an LNS approach for using nested neighborhoods of different sizes, aimed at avoiding local minima. Similarly, Pisinger and Ropke [25] presented an Adaptive LNS framework capable of managing capacitated deliveries, time windows, multiple depots, split deliveries, and open routes constraints. Through evaluations across diverse sets of instances, encompassing up to 1,000 customers, they achieved notable improvements, enhancing 183 out of 486 best-known solutions derived from benchmark tests.

A case study presented by Pellegrini et al. [19] encompasses multiple objectives and constraints, including multiple time windows, a Heterogeneous fleet of vehicles, a maximum duration for sub-tours, and periodic customer visits. They explored two versions of Ant Colony Optimization (ACO): the Multiple Ant Colony System [26], and the MAX-MIN Ant System [27]. The authors conducted a comparative analysis with a Tabu Search (TS) algorithm and a RNN heuristic. Both ACO algorithms exhibited significantly superior performance compared to the TS and RNN methods when tested with an instance generator featuring 70–80 orders.

Several studies have developed methods based on Column Generation (CG). Oppen et al. [28] address a real scenario known as the Livestock Collection Problem

(LCP), which is an extension of the RVRP with inventory constraints. This scenario involves duration, heterogeneous fleets restricted in capacity, time windows, multiple trips, and multi-product considerations. The authors tackled it through an exact solution method based on CG, generating instances with fewer than 30 customer orders inspired by real-world scenarios. While the CG approach has successfully found optimal solutions in various scenarios, the authors noted limitations in finding optimal solutions for LCP instances. A CG method based on a heuristic behavior is proposed by Goel [29] to address a VRP with time windows, a heterogeneous vehicle fleet, multiple depots, and pickup and delivery constraints. Small instances are randomly generated to evaluate the heuristic's performance. Ceselli et al. [30] propose the use of CG combined with a dynamic programming algorithm to address simultaneously a heterogeneous fleet, different depots, time windows, variations in route lengths, optionally opened routes, and pickup and delivery constraints, among others. Their approach is tested with 46 randomly generated instances composed of 100 orders, and the results are compared with valid lower bounds. In a similar context, Ruinelli [31] compared three methods in thesis study: an Ant Colony System (ACS), a CG algorithm, and a general-purpose Mixed-Integer Linear Programming (MILP) solver. Computational results are presented using 14 real instances from a distribution company, where the CG method outperforms the other two methods. Santillán et al. [32] addressed a routing-scheduling-loading problem utilizing a heuristic-based system. Initially, their proposed system employs an ACS for routing and scheduling, followed by the application of a bin packing technique for vehicle loading. Tests are conducted on instances seen in [21], as well as real distribution data from a Mexican company were performed.

The work done by Vidal et al. [33] covers a comprehensive solution framework named Unified Hybrid Genetic Search (UHGS) designed for various RVRP variants.

The framework employs generic local search and genetic operators. The authors provide computational findings based on 39 benchmarks across 26 distinct Rich VRPs. Xia and Fu [34] focus on the capacitated open vehicle routing problem with split deliveries by order. Their research presents a dual-objective mathematical model, integrating split deliveries by order, and devises an adaptive TS algorithm tailored to tackle the problem efficiently. To enhance optimization performance, the study introduces enhancements such as adaptive penalty mechanisms and multiple neighborhood structures. Comparative analysis with other methods in the literature underscores the effectiveness of the algorithm. Similarly, Azadeh and Farrokhi [35] explore the multi-depot Vehicle Routing Problem (MDVRP) with close-open mixed VRP (COMVRP), having a heterogeneous fleet of vehicles and incorporating fleet contractors into customer fulfilment. Their study introduces a novel mixed-integer programming (MIP) model complemented by a hybrid metaheuristic approach. This hybridization involves the integration of a genetic algorithm (GA) with the analytic hierarchical process, aiming to efficiently navigate the optimization space. Computational experiments are conducted to compare the performance of the hybrid GA with other optimization techniques, emphasising its efficacy in solving real-world distribution challenges.

Ferreira et al. [36] delve into a CVRP with Two-Dimensional Loading Constraints, exploring variations such as split deliveries and green requirements. Their approach involves formulating mathematical models for each variation and employing a branch-and-cut (B&C) approach to solve the instances. A customized procedure is developed to tackle the packing sub-problem, integrating lower bounds computation, heuristic-based constructive methods, and constraint programming formulations. Computational experiments conducted on existing literature instances and newly generated ones demonstrate the proposed approach's superiority over previous results. Similarly, Zhang et al. [37] tackle a Two-Dimensional loading open vehicle routing

problem with time windows, proposing a multi-objective learning whale optimization algorithm (MLWOA) to solve it. Their approach integrates routing and loading sub-problems, with the MLWOA devised as a two-phase algorithm to address these components. Simulation analysis demonstrates the superior performance of the MLWOA algorithm over standard WOA and other heuristic algorithms, establishing its absolute advantage.

Lesch et al. [38] address an RVRP, characterized by real-world constraints such as heterogeneous fleets and time windows. Their study introduces a two-stage strategy alongside a Timeline algorithm for managing time windows and pause times efficiently. Furthermore, GA and ACO are separately applied to the problem to identify optimal solutions. Evaluation against well-known algorithms illustrates the effectiveness of their approach in handling specified constraints within a reasonable time-frame. Rajaei et al. [39] focus on a split delivery heterogeneous vehicle routing problem with three-dimensional loading, proposing a CG heuristics (CGH) algorithm tailored to handle routing and loading constraints efficiently. Leveraging computational experiments on literature instances, their approach demonstrates competitive performance, yielding solutions close to existing algorithms in shorter time-frames. Notably, the study presents real-world instances, significantly larger than existing benchmarks, showcasing the proposed algorithm's effectiveness in practical distribution scenarios.

Kabadurmus and Erdogan [40] integrate the Green Vehicle Routing Problem with Multi-Depot, Multi-Tour, Heterogeneous Fleet, and Split Deliveries, formulating it as a mixed-integer linear programming (MILP) model aimed at minimizing overall carbon emissions. They develop a GA with constraint handling techniques to tackle the problem efficiently. Computational experiments conducted on synthetic problems underscore the efficiency of the proposed GA approach in minimizing carbon emissions.

Vieira et al. [41] introduces two metaheuristics tailored for solving the Heterogeneous Site-Dependent Multi-depot Multi-trip Periodic Vehicle Routing Problem. The first metaheuristic adapts the UHGS, while the second introduces a new approach that integrates the VNS with adaptive mechanisms. Computational experiments utilize 398 instances from existing literature and introduce 20 new instances. The results of the metaheuristics outperform or match those obtained by several state-of-the-art algorithms, demonstrating their effectiveness. Among the 398 instances from the literature, the proposed metaheuristics discover 140 new best-known solutions and successfully match 209 of the best-known solutions. For the remaining instances, both approaches yield results closely resembling the best-known solutions.

A study conducted by Garside et al. [42] aims to identify the best fleet composition considering both capacity limitations and route optimization to minimize distance and transportation expenses. Vehicle type allocation is determined based on the proximity of customers to the distributor. The allocation of retail vehicles utilizes the Clarke Wright saving heuristic (C&W), while the GA is utilized to determine the optimal route for each vehicle. Employing a real-world case study from an Indonesian LPG distributor company, it is observed that both methods achieve notable distance savings.

Up to our knowledge, the only study that encompasses all the features of our problem to some extent is the one conducted by Tamke in 2015 [43]. In this study, the focus is on an RVRP incorporating a real-world cost function within the distribution planning context of a major German food retail company. The pricing mechanism relies on transportation tariffs, reflecting the retailer's collaboration with multiple freight carriers. Furthermore, the problem formulation encompasses a combination of constraints and characteristics from various VRP variants, including Capacitated, Heterogeneous, Open VRP, and Split Delivery VRPs. Although the authors refrain

from developing specific solving methodologies, they conclude the study by emphasizing the complexity of the problem and its deviation from its original formulation proposed by Dantzig and Ramsen in 1959 [44]. Given the constraints of short planning horizons, they underscore the importance of employing heuristics and metaheuristics to derive *fair* solutions.

Table 1 provides a an overview of common features identified across the top 20 related papers from different years, along with the main solution methods employed. Above each column representing a feature, the abbreviation of the corresponding characteristic is written, which is spelled out in the Chapter 2. A tick in the respective feature column denotes the presence of that feature in the study in the same/similar fashion. The first column represents the number of similar features found in each paper.

Table 1: Papers with mutual characteristics and their solving approaches

*	Author(s)	Year	CV	HF	LP	UF	SD	IR	MV	O	MO	Method(s)
6	Goel & Gruhn [23]	2006	✓	✓	✓		✓	✓		✓		LNS
7	Pellegrini et al. [19]	2007	✓	✓		✓	✓	✓	✓		✓	ACO
5	Pisinger & Ropke [25]	2007	✓	✓		✓	✓			✓		LNS
6	Goel & Gruhn [24]	2008	✓	✓	✓	✓	✓			✓		VNS, LNS
7	Ceselli et al. [30]	2009	✓	✓		✓	✓	✓	✓	✓		CG
5	Oppen et al. [28]	2010	✓	✓	✓	✓			✓			CG
5	Goel [29]	2010	✓	✓		✓	✓			✓		CG
6	Ruinelli [31]	2011	✓	✓			✓	✓		✓	✓	CG
6	Santillán et al. [32]	2012	✓	✓		✓	✓		✓	✓		ACO
7	Vidal et al. [33]	2013	✓	✓		✓	✓	✓		✓	✓	LS, UHGS
9	Tamke [43]	2015	✓	✓	✓	✓	✓	✓	✓	✓	✓	-
4	Xia & Fu [34]	2018	✓						✓	✓	✓	TS
6	Azadeh & Farrokhi [35]	2019	✓	✓		✓		✓		✓	✓	GA
5	Ferreira et al. [36]	2021	✓	✓	✓				✓		✓	B&C
6	Lesch et al. [38]	2022	✓	✓	✓		✓	✓			✓	GA, ACO
7	Rajaei et al. [39]	2022	✓	✓	✓	✓	✓		✓		✓	CG
6	Kabadurmus & Erdogan [40]	2023	✓	✓		✓		✓	✓		✓	GA
4	Vieira et al. [41]	2023	✓	✓			✓	✓				UHGS, VNS
4	Zhang et al. [37]	2024	✓		✓					✓	✓	MLWOA
4	Garside et al. [42]	2024	✓	✓			✓				✓	C&W, GA

In concluding this chapter, it becomes evident that the RVRP has garnered significant attention within the literature. Numerous studies have addressed the challenges posed by integrating different variants of the VRP alongside real-life constraints. Notably, between 2006 and 2013, solution methodologies encompassed both non-exact and exact approaches, with notable contributions from CG. However, since 2013, advancements in technology have facilitated the removal of assumptions and the more effective management of cost-related factors . Furthermore, the emergence of new

challenges in the field has led to a combination of multiple VRP variants [6]. Consequently, this evolution has brought forth challenges associated with addressing larger-scale factors that influence these problems. This marked a significant shift towards the adoption of metaheuristic techniques. Authors increasingly favor metaheuristics for their remarkable effectiveness in addressing large-scale problems that encompass diverse factors.

CHAPTER IV

SOLUTION APPROACHES

In this chapter, we cover the solution methods used while tackling the core problem of this thesis study. Our solution methods include 1) an exact method which used a mathematical model that is then solved using a commercial solver, 2) a heuristic algorithm, and 3) the use of three metaheuristic algorithms.

4.1 Exact method: Commercial solver

As an exact solution approach, we propose utilizing a commercial solver. Given a mathematical model, the solver attempts to find the optimal solution within a set time limit. If the solver identifies the best solution within this time-frame, it presents it. However, if an optimal solution isn't found, it provides the best solution discovered within the time limit, along with an estimated lower bound for the problem's expected cost. We present the terminologies and nomenclature essential to the mathematical model in Table 2, and the mathematical model formulated as an MILP:

Objective function

$$\textbf{min} \quad \sum_{v \in V} stopCost_v ExStops_v + TabulatedCost_v + travelCost_v ExDist_v \tag{1}$$

Constraints

$$\sum_{v \in V} tempSuit_{pv} Z_{pv} = 1 \qquad \forall p \in P \tag{2}$$

$$\sum_{p \in P} w_p Z_{pv} \leq W_v \qquad \forall v \in V \tag{3}$$

$$\sum_{p \in P} belong_{pj} pl_p Z_{pv} \leq PalletsUsed_{jv} \qquad \forall j \in C^*, v \in V \tag{4}$$

Table 2: Nomenclature of the mathematical model

Sets	
$C = \{0, 1, 2, \ldots, n\}$	Set of nodes. Node 0 is the depot, nodes 1-n represent customer sites.
$C^* = C \setminus \{0\}$	Set of customer nodes (excluding the depot).
$P = \{1, 2, 3, \ldots, m\}$	Set of products
$V = \{1, 2, \ldots, k\}$	Set of vehicles.
Parameters	
$normal_p$	1 if product $p \in P$ is normal type, 0 o/w.
$fridge_p$	1 if product $p \in P$ is fridge type, 0 o/w.
$belong_{pj}$	1 if product $p \in P$ belongs to customer $j \in C^*$, 0 o/w.
$tempSuit_{pv}$	1 if vehicle $v \in V$ has temperature suiting product $p \in P$, 0 o/w.
$Nlimit_j$:	Vehicle limit for sharing the normal products of customer $j \in C^*$.
$Flimit_j$:	Vehicle limit for sharing the fridge products of customer $j \in C^*$.
$maxStops$	Maximum number of customer visits any vehicle can make.
$stopCost_v$	Cost of making an extra stop using vehicle $v \in V$.
$freeStops$	Maximum free stops per vehicle.
d_{ij}	Distance between node $i \in C$ and node $j \in C$.
$cost_{jv}$	Cost of visiting customer $j \in C^*$ using vehicle $v \in V$.
$enter_{jv}$	1 if customer $j \in C^*$ can be visited by vehicle $v \in V$.
$tgthr_{jn}$	1 if customers j and $n \in C^*$ can share the same vehicle, 0 o/w.
Pl_v	Pallet capacity of Vehicle $v \in V$.
W_v	Weight capacity of Vehicle $v \in V$.
pl_p	Pallet need of product $p \in P$
w_p	Weight of product $p \in P$.
$travelCost_v$	Traveling cost per km of vehicle $v \in V$.
M	Significantly big number.
Decision variables	
Z_{pv}	1 if product $p \in P$ is assigned to vehicle $v \in V$, 0 o/w.
N_{jv}	1 if customer $j \in C^*$ has normal products on vehicle $v \in V$, 0 o/w.
F_{jv}	1 if customer $j \in C^*$ has fridge products on vehicle $v \in V$, 0 o/w.
x_{ijv}	1 if vehicle $v \in V$ visited node $j \in C$ after node $i \in C$, 0 o/w.
$Furthest_{jv}$	1 if customer $j \in C^*$ is furthest on vehicle $v \in V$, 0 o/w.
$PalletsUsed_{jv}$	Number of pallets for customer $j \in C^*$ on vehicle $v \in V$.
$ExStop_v$	Number of extra stops vehicle $v \in V$ made.
$DistRed_v$	The reduction in distance applied to vehicle $v \in V$.
$TabulatedCost_v$	The cost that must be payed for using vehicle $v \in V$.
$ExDist_v$	The extra distance vehicle $v \in V$ travelled.
u_{jv}	The visiting order of customer $j \in C^*$ on vehicle $v \in V$.

$$\sum_{j \in C^*} PalletsUsed_{jv} \leq Pl_v \qquad \forall v \in V \tag{5}$$

$$\sum_{p \in P} belong_{pj} normal_p Z_{pv} \leq MN_{jv} \qquad \forall j \in C^*, \forall v \in V \tag{6}$$

$$\sum_{p \in P} belong_{pj} fridge_p Z_{pv} \leq MF_{jv} \qquad \forall j \in C^*, \forall v \in V \tag{7}$$

$$\sum_{v \in V} N_{jv} \leq Nlimit_j \qquad \forall j \in C^* \tag{8}$$

$$\sum_{v \in V} F_{jv} \leq Flimit_j \qquad \forall j \in C^* \tag{9}$$

$$F_{jv} + N_{jv} \leq 2enter_{jv} \sum_{i \in C} x_{ijv} \qquad \forall j \in C^*, \forall v \in V \tag{10}$$

$$\sum_{i \in C} x_{ijv} + \sum_{i \in C} x_{inv} \leq 1 + tgthr_{jn} \; \forall j, n \in C^*; j \neq n, v \in V \tag{11}$$

$$\sum_{j \in C} x_{njv} = \sum_{i \in C} x_{inv} \qquad \forall n \in C, \forall v \in V \tag{12}$$

$$u_{iv} - u_{jv} + n x_{ijv} \leq n - 1 \qquad \forall i, j \in C^*, i \neq j, v \in V \tag{13}$$

$$\sum_{i \in C} \sum_{j \in C^*} x_{ijv} \leq freeStops_v + ExStops_v \qquad \forall v \in V \tag{14}$$

$$TabulatedCost_v \geq \sum_{i \in C} cost_j x_{ijv} \qquad \forall j \in C^*, v \in V \tag{15}$$

$$\sum_{j \in C^*} Furthest_{jv} = 1 \qquad \forall v \in V \tag{16}$$

$$DistRed_v \leq \sum_{i \in C} depotDist_j x_{ijv} + M(1 - Furthest_{jv}) \;\; \forall j \in C^*, v \in V \tag{17}$$

$$ExDist_v \geq \sum_{i \in C} \sum_{j \in C^*} d_{ij} x_{ijk} - DistRed_k \qquad \forall v \in V \tag{18}$$

$$1 \leq u_{jv} \leq n, u_{jv} \in Integer \qquad \forall j \in C^*, \forall v \in V \tag{19}$$

$$0 \leq ExStops_v \leq maxStops, ExStops_v \in Integer \qquad \forall v \in V \tag{20}$$

$$Z_{pv}, x_{ijv} \in \{0, 1\} \qquad \forall p \in P, i, j \in C, v \in V \tag{21}$$

$$N_{jv}, F_{jv}, Furthest_{jv} \in \{0, 1\} \qquad \forall j \in C^*, v \in V \tag{22}$$

$$PalletsUsed_{jv} \geq 0, PalletsUsed_{jv} \in Integer \;\; \forall j \in C^*, \forall v \in V \tag{23}$$

$$DistRed_v, TabulatedCost_v, ExDist_v \geq 0 \qquad \forall v \in V \tag{24}$$

The objective function (1) minimizes the sum of the three cost components resulted from each vehicle $v \in V$. Using (2), we load each product on a vehicle, while ensuring that the vehicle's temperature suits the product. (3) regulates the total weight loaded on each vehicle $v \in V$. (4) first checks the pallets for customer $j \in C^*$ loaded on vehicle $v \in V$ and rounds up the result to the nearest integer. Then (5) ensures that the total pallets for each vehicle $v \in V$ do not surpass its maximum capacity. (6) and (7) first checks which vehicles have the normal and fridge products belonging to customer $j \in C^*$ respectively. Afterwards, (8) and (9) limits the number of vehicles that can deliver each type of products belonging to customer $j \in C^*$. (10) ensures that if vehicle $v \in V$ has any products belonging to customer $j \in C^*$, it must stop at their location. (11) restricts customer j and $n \in C^*$ from sharing the same vehicle if any of them do not prefer. (12) ensuring that the balance between entering and leaving nodes is kept, and (13) prevents sub-tours from forming. Through (14), we keep a count of how many extra stops vehicle $v \in V$ made, and (15) assigns the tabulated cost of using vehicle $v \in V$ by comparing all the cost of customers that the vehicle must visit, and setting it as the largest of them all. (16) sets one customer to be furthest away from the depot on vehicle $v \in V$. (17) compares how far from the depot all the customers served by vehicle $v \in V$ are. The further the customer is from the depot, the more reduction in distance vehicle v will get. Consequently, the furthest customer served by the vehicle will be selected. (18) checks the extra distance crossed by vehicle $v \in V$ by comparing its total distance (disregarding returning to the depot), and the distance of the furthest customer served by the vehicle. (19) through (24) govern the behavior of the decision variables.

4.2 Non-exact methods: Heuristic and Metaheuristic algorithms

In this subsection, we delve into a heuristic and three metaheuristic algorithms employed as non-exact solving approaches to tackle the problem under concern. Before that, we go over the aspects of solutions created within the algorithms.

4.2.1 Solution framework

In order to preserve the solutions generated by the algorithms under consideration, a system of bins is employed. Each bin serves as a repository for the information concerning the delivery process. Within these bins, details encompassing the products designated for transport, the corresponding customers associated with each product, the type of vehicle selected for transportation, and the route, outlining the sequence of visits the vehicle must make when delivering to the respective customer sites. Figure 2 illustrates a sample solution for a problem involving 7 products distributed among 4 customers. Notably, all routes originate from node 0, the depot. Note that when implementing any modification moves in the forthcoming algorithms, we target the products within each bin instead of the conventional approach used in the VRP, which typically focuses on customers. This deviation is essential because each customer may have multiple products, making it inadequate to address the customers.

Bin 1			
Products	1	2	3
Customers	1		2
Vehicle Type	Refrigerated Truck		
Route	0	2	1

Bin 2				
Products	4	5	6	7
Customers	3		4	
Vehicle Type	Refrigerated Trailer Truck			
Route	0	3	4	

Figure 2: Solution representation

Within our algorithms, we employ two methods that guide us in making optimal decisions when encountering a bin. The first method, 'Assign Cheapest Vehicle', is employed to determine the most economical vehicle for each bin assignment. Meanwhile, the second method, 'Find Best Route', is utilized to identify the optimal route

assigned to each bin. Below, we delve into the specifics of each of these methods.

Assign Cheapest Vehicle

Given the heterogeneous nature of our vehicle fleet, the service of different types over the same set of products will result in varying costs, and feasibility outcome. The cost elements of the vehicle types follow this hierarchy from least to most expensive: Truck, Refrigerated truck, Trailer truck, then Refrigerated trailer truck. Thus, our aim is to optimize cost by assigning the most economical vehicle to each bin while ensuring it can accommodate all the products, considering various requirements such as total capacity, refrigeration needs, and the accessibility of customer locations.

Our approach attempts to assign the cheapest vehicle type to each bin. If this is not feasible, we proceed to the next available option, and so forth. In cases where none of the vehicle types can accommodate the product set, we deem that bin assignment as infeasible and discard it.

Find Best Route

This method is designed to optimize each route assigned to each bin, which is akin to solving a TSP. The required customer sites to be visited are directly linked to the products contained within each bin. Furthermore, if a product is loaded onto a bin and is requested by customer X, it becomes imperative to visit customer X's site.

Typically, when dealing with a TSP, non-exact algorithms are favored for large problem sizes due to its NP-hard nature [45]. One constraint of the problem in this thesis study restricts each route to a maximum of six stops at customer locations. This constraint streamlines the overall computational process, reducing the time required to determine the optimal sequence of visits for each vehicle. Consequently, we have chosen to utilize an exact method whenever confronted with the routing problem. Here, we consider two methods capable of determining the optimal sequencing for small problem instances, aiming to choose the most suitable one for our needs. The

first method involves enumeration, where we explore all possible visiting combinations and select the one that yields the cheapest sequencing of visits. The second method entails utilizing a typical TSP mathematical model [46], which is then inputted to an exact solver to identify the optimal sequence of visits. The method that achieves optimal routing in the shortest time will be preferred. To evaluate the time complexity of each method, we applied them to TSP problem sets with varying number of nodes that need to be visited. We then measured the time taken by each method to optimize the problem. Table 3 provides a summary of the time required by each method to solve the problem based on its size. The first column denotes the number of nodes tested each problem. The 'Enumeration' and 'Exact Solver' columns display the average time, in milliseconds, taken from three runs to solve each respective problem optimally.

Table 3: Time complexity study on Enumeration & Exact solver (in milliseconds)

# nodes	**Enumeration**	**Exact solver**
2	0.002	2.105
3	0.002	3.245
4	0.003	5.979
5	0.012	10.700
6	0.069	40.102

It's evident that for all the problems with node sizes up to six, the enumeration method consistently discovers optimal solutions significantly faster than the exact solver. Consequently, we have decided to consistently utilize the enumeration method for optimizing the routing problem within each bin.

When faced with a new combination of products grouped together within the same bin, this prompts the selection of the cheapest vehicle type and the establishment of a new optimal sequence for visiting their respective customers. As a result, the two

methods described above are activated whenever our algorithms generate a new solution that includes a recently discovered combination of products. Figure 3 provides an example demonstrating the application of these two methods on a bin.

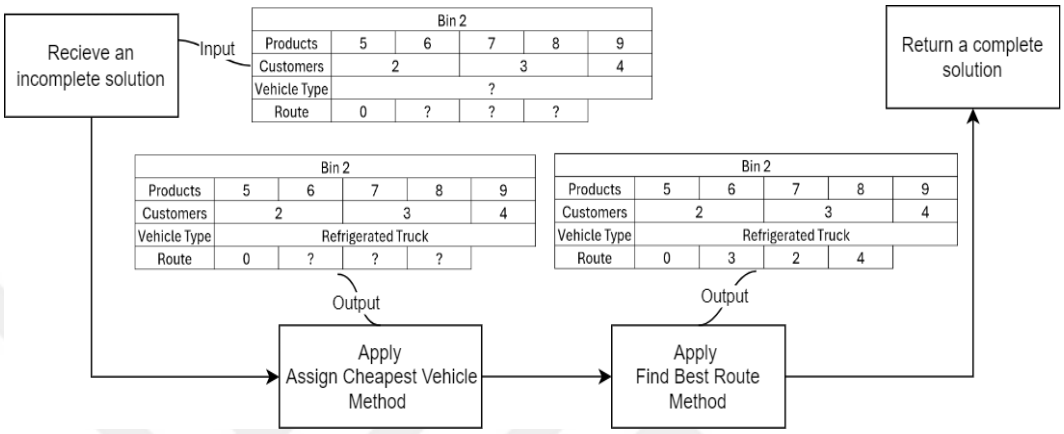

Figure 3: Example of the the solution framework

4.2.2 Heuristic algorithm: Clarke and Wright Savings Algorithm

Heuristic algorithms are frequently utilized for their ease of implementation and capability of providing solutions in a short time. Among the heuristic algorithms commonly used for vehicle routing problems, the Clarke and Wright saving algorithm [47] stands out. This algorithm explores combining two routes into one, identifying potential savings iteratively.

In our context, we have also adapted the Clarke and Wright's for swiftly obtaining solutions, which may be optimal or sub-optimal. These solutions also serve in providing valuable initial solutions when employing metaheuristic algorithms. Below, we outline the implementation of the algorithm to address our specific problem:

The algorithms commence by initially allocating products to individual bins, resulting in an initial solution and its associated cost. Subsequently, an iterative process is initiated with the primary objective of minimizing the total number of bins and consequently, the number of vehicles used. In each iteration, we evaluate the feasibility of transferring all products from each bin to all others individually, and assess

the potential cost savings. The transfer that is both feasible and yields the highest reduction in cost is selected for implementation. This iterative framework continues until reaching a point where no further feasible and cost-reducing elimination of bins can be made.

This implementation is aimed at continually enhancing the efficiency of our solution through the optimization of product allocation across bins. As a result, we anticipate a more streamlined product assignment, requiring fewer vehicles to serve customers and ultimately leading to a cost-effective transportation plan.

4.2.3 Metaheuristic algorithms: SA, MBO, and Hybrid MBO-SA

In this subsection, we provide an in-depth overview of the framework underlying the metaheuristic algorithms utilized in this study. This encompasses two methods employed by the algorithms throughout their iterations, along with a detailed explanation of the implementation of each metaheuristic algorithm.

The Neighborhood search method

Metaheuristic algorithms harness randomness to navigate solution spaces in search for optimal solutions. They achieve this by introducing random modifications to existing solutions, thus uncovering new potential solutions. These random adjustments are known as neighborhood search moves. Below, we explore our neighborhood search method in detail.

When selecting a neighborhood search method for a local search-based algorithm, it's crucial to understand its main cost elements. Our problem's cost structure is contingent upon the products assigned to each bin, the customer locations that must be visited, the vehicle that will serve the products, and the route it must traverse for delivery. The neighborhood search moves commonly employed in solving VRP and its variants typically involve adjustments within each bin, known as intra-route moves, or transitions between different bins, referred to as inter-route moves. As previously

stated, the optimal routing for each vehicle is consistently determined using the 'Find Best Route' method. Consequently, our neighborhood search exclusively encompasses inter-route moves.

Maintaining a diverse set of tools within search spaces is crucial for effective exploration of solution neighborhoods [48]. Therefore, we introduce six distinct inter-route methods utilized for exploring these solution neighborhoods. It is important to note that after each move is executed, the 'Assign Cheapest Vehicle' and 'Find Best Route' methods are applied, as a new set of products and customers that need to be visited is realized, thus a new problem that must be optimized.

Figure 4 visually showcases the outcomes of each method after being applied to the product assignment of each bin, Under each method are the changes that are made after each move is applied. Notably, variations in vehicle type and route may occur to illustrate how a new assignment of products and customers can reveal different optimal vehicle types and routes. In addition to a comprehensive explanations detailing their individual implementations below:

1. Insertion: Select a product randomly from one bin and transfer it to another bin chosen arbitrarily.

2. Swap: Select a product randomly from a pair of bins, and swap their assignments.

3. Swap 2 for 1: Randomly select two products assigned on the same bin and swap their assignments with a product from another bin chosen randomly.

4. Double Swap: Select two bins arbitrarily, then randomly select two products from each bin and interchange them with the products from the other bin.

5. Large Swap: This method covers all swap moves between two bins that have not been covered by the preceding four methods. We randomly choose two bins. From each selected bin, we then randomly pick a subset of products of varying sizes. Subsequently, we transfer each subset from one bin to the other.

6. Cycle: Choose three bins, say 1, 2, and 3. From each bin, select a product. Then, move the first product to bin 2, the second product to bin 3, and the third product to bin 1.

All the methods mentioned above either maintain the same fleet size or result in its reduction upon implementation. In a study conducted by Derigs et al. [49], a neighborhood search move was introduced, which increases the size of the fleet being utilized. The authors suggest that enlarging the fleet size can potentially reduce the overall distance traveled by all vehicles. After consulting with our collaborating distributor, they asserted that, on average, 80-90% of the total cost incurred from their solutions arises from the tabulated cost of the vehicles. Moreover, literature commonly holds the belief that focusing the reduction of necessary vehicles is more critical than minimizing travel expenses from a cost perspective [50]. These insights suggest that efforts should be concentrated on reducing the overall fleet size.

The selection of the mentioned methods was informed by observing them in related papers that employed inter-route neighborhood search moves. Additionally, our decision was influenced by observations specific to our problem domain. The 'Insertion' and 'Swap' methods are widely recognized across various contexts for solving optimization problems. Their popularity stems from their straightforward implementation, minimal time requirement, and effectiveness in shuffling solutions to explore new areas in a neighborhood. The 'Swap 2 for 1' method proves valuable in scenarios where a vehicle is fully occupied and an insertion or a swap may be infeasible or cause the need of a bigger and more expensive vehicle type due to capacity limitations. However, by removing two products simultaneously, more room becomes available, facilitating the other of a product. Given that multiple products belonging to the same customer can be loaded on a single vehicle, it can be more economical to move both products together during a neighborhood search move. This consideration reduces the necessity for two separate vehicle visits to the same customer location,

Figure 4: The Neighborhood search strategies

hence the inclusion of the 'Double Swap' method. To account for methods that are infrequently mentioned in the literature or have an unforeseen advantage, we have also incorporated the 'Large Swap' method. This method considers all potential swap combinations that have not been covered by the previous methods. Moreover, algorithms may occasionally become stranded in a local optimum. To escape from this situation, a move that substantially alters the solution's structure becomes imperative [51]. The 'Cycle' method was incorporated to accommodate combinations involving three bins, as it can also aid in overcoming local optima.

After selecting the neighborhood search moves to employ, it is important to determine their frequency of occurrence within the algorithms. We evaluated the performance of each method by measuring their capacity to generate feasible improving solutions and the time required for their execution. To assess these aspects, we independently executed each method iteratively and recorded the number of improving solutions found by each method within the same time limit. The effectiveness of each method in exploring fruitful solutions within a neighborhood correlates with its outcome. Table 4 summarizes the average number of improving solutions discovered by each method over three runs within the same runtime. To determine the final probability for each method, we calculated the total number of solutions found by all methods and then calculated the probability for each method based on the number of solutions it discovered out of the total.

Table 4: Probability assignment of the Neighborhood Search methods

Method	**Improving Moves**	**Probability**
Insertion	13781	22.3%
Swap	16521	26.8%
Swap 2 for 1	9619	15.6%
Double Swap	11643	18.9%
Large Swap	4298	7.0%
Cycle	5807	9.4%
Total	61668	100%

The Neighborhood search method is activated whenever our metaheuristic algorithms, which we will discuss in more detail next, seek to discover a new solution. It plays a key role in exploring solution spaces, aiding in the iterative refinement process to uncover potentially optimal solutions.

Enhance Fleet assignment

As previously discussed, the cost of the solution is greatly influenced by the number of vehicles utilized. A problem requiring fewer vehicles typically yields a solution with lower costs. To address this factor, we integrated a method into the metaheuristics. When this method is called, it assesses the average utilization of each bin in terms of its vehicle's pallets and weight. Subsequently, it selects the bin with the least utilization. Following this selection, an attempt is made to either remove the bin entirely or switch its assigned vehicle type to a cheaper one, such as transitioning from a Trailer truck to a Truck.

This process involves attempting to relocate all products loaded in the selected bin to the another bin chosen randomly. If feasible, it reduces the total number of bins used and the fleet size. If only the relocation of a portion of the products is feasible, the process proceeds to the 'Assign Cheapest Vehicle' method. In this method, the algorithm evaluates whether changing the bin's vehicle type from a larger, more expensive trailer truck to a smaller, cheaper truck reduces the overall solution cost. If so, the modified solution is incorporated back into the metaheuristic algorithm.

This approach is implemented by selecting the top-performing solution preserved by each metaheuristic at the time the method is called. The method is triggered every time 5% of the algorithm's overall runtime has passed. However, it only becomes active after 50% of the algorithm's execution is completed. This decision arises from the observation that algorithms become more vulnerable to being trapped at local optima during the initial stages, a scenario typically undesirable as it limits the

algorithm's ability to explore other areas in the neighborhood.

4.2.3.1 Migrating birds optimization algorithm (MBO)

The MBO algorithm draws inspiration from the communal behavior, resembling a V-shape, observed in bird flocks during migration. This algorithm stems from the recognition that birds during migration employ efficient navigation and communication strategies. Below, we outline the general flow of the MBO algorithm.

1. Initialization (set the algorithm's parameters and construct a V flight shape):
   - n: The number of birds (preferably an odd number). Each bird represents a solution.
   - k: The number of neighboring solutions to generate for the leader.
   - x: The number of neighbors to pass downwards.
   - m: The number of iterations maintaining the same leader solution.
   - K: The iteration limit where each neighbor solution generated counts as an iteration.
   - V flight formation: Arrange n many initial birds in a V-shaped structure, with one bird (the leader bird) positioned at the tip, and others of equal size positioned symmetrically on both the left and right sides.
2. Main loop:

   Generate k neighboring solutions from the leader bird. If the best neighbor is superior (cheaper in our case), it replaces the leader bird. Pass x unused neighbors to both the left and right sides. For each bird on each side individually, generate k-x neighboring solutions. If the best neighbor is superior, adopt it. Otherwise, evaluate the passed solutions and adopt the best one if it surpasses the current bird. Subsequently, update the passed solutions with the unused neighbors of the current birds only and pass them down to the preceding bird. Repeat this process until the last bird on each side is reached.

Once this scheme is completed m times, pass the leader bird to the bottom of either side, with the first bird from the same side becoming the new leader. The algorithm terminates after K many leader bird changes.

3. Report best bird found

Approximating the best parameter values for a metaheuristic algorithm is of paramount importance as it significantly impacts its performance [52]. Regarding the MBO algorithm, when initiating a solution for the initial birds, a straightforward approach is to set them based on a randomized approach. However, valuable potential can be gained from the solution derived by Clarke and Wright's algorithm. Hence, we opted to incorporate that solution. Yet, initializing all the birds with this solution isn't advisable, as it heightens the risk of early convergence, given that all birds commence from the same direction. Consequently, we resolved to designate each bird in the flock as the Clarke and Wright solution, subjecting it to two neighborhood search moves to introduce deviations from the original solution and foster diversity among the birds.

To determine the rest of the parameters specific to the MBO algorithm, we utilized a grid search-based approach. We systematically tested different values on the same problem for each parameter while keeping the other parameters constant throughout the algorithm. The values that produced the best cost according to our problem on average, were identified as the optimal. The experimented values are suggested within a range of values used in studies that employed the algorithm [53; 54].

Table 5 displays the grid outlining the exploration of each parameter value, with the best resulting value highlighted in bold font.

Table 5: MBO algorithm parameter grid & values determination

n	k	x	m
5	3	**1**	5
11	**5**	2	**10**
15	7	3	15
21	9	4	20

4.2.3.2 Simulated annealing (SA)

Simulated annealing is a metaheuristic algorithm inspired by the annealing process in metallurgy. The algorithm operates based on the Metropolis criterion, which guides the acceptance or rejection of new solutions during the exploration process. This criterion allows the algorithm to probabilistically accept worse solutions, particularly during the early stages when the temperature is high. As the algorithm progresses and the temperature decreases, the likelihood of accepting worse solutions diminishes, promoting a more focused search for better solutions. This phase, known as the exploitation phase, allows simulated annealing to refine its search and converge towards optimal or near-optimal solutions. Below is a detailed implementation guide for simulated annealing.

1. Initialization (set the algorithm's parameters):
    - T_i (initial temperature): The temperature the algorithm starts with
    - T_f (final temperature): The temperature during the last iteration
    - L (epoch length): Iterations to be performed at every temperature level
    - α (cooling rate): A factor that takes a value between 0 and 1, used to reduce the temperature
    - S_c (current solution): Initialize the current solution using a heuristic algorithm or randomly.
    - S_b (best solution so far): Initialize the best solution as S_c.
2. Main loop:

Generate a new solution neighboring S_c. If the new solution is superior (cheaper in our case), we adopt it as S_c. It is also adopted as S_b if it is the best solution ever found. If it has a lower quality, we decide whether to adopt it based on the Metropolis criterion.

After this scheme is done L many times, T_i is reduced by multiplying it with α. The algorithm terminates when T_i becomes smaller than or equal to T_f.

3. Report best solution (S_b) found

To obtain an initial solution to set S_c with, we employ the outcome generated by Clarke and Wright's algorithm. This solution also guides the establishment of the initial temperature T_i systematically, leveraging a methodology outlined in [54]. The authors propose setting T_i to a level where solutions deviating by x% in cost possess a y% probability of acceptance during the algorithm's early phases. This approach facilitates control over the avoidance of unfavorable search spaces at the beginning of the algorithm. Based on our experimentation, we have found that using higher percentages works well for addressing small-scale problems. However, when it comes to larger problems, employing such percentages tends to yield poor results. This discrepancy is primarily because smaller problems inherently have a more constrained solution space. Thus, it's easier to navigate through *bad* search spaces in smaller problems, since finding a path that leads to *good* search spaces eventually can be found. Whereas in larger problems, there's a greater risk of getting trapped in unfavourable areas upon visiting them, without a clear route back to more promising ones. Consequently, during Phase I, we configure the acceptance probability for solutions deviating by 25% in cost to be 5%. On the other hand, in Phase II, solutions deviating by 1% in cost are accepted with a probability of 5% (refer to Chapter 6 for Phase I and Phase II).

Assigning a specific value to L would not facilitate stopping the algorithm at a

desired time limit. Subsequently we adapt a different approach, similarly carried out by Bassaleh and Duman [55], where we initiate the process by calculating the necessary reductions to T_i by α required to attain the final temperature T_f. This calculation can be facilitated using a formula commonly employed in economics for compound interest calculation [56]. The formula is as follows:

$$F_v = P_v + (1 + r)^n$$

In the formula, F_v stands for the future value, P_v is the present value, r is the interest rate, and n is the number of compounding periods. The formula addresses the question of what the future value would be for an investment made now, accruing interest at a rate of r after n periods. Similar to our problem, we inquire about the number of periods required for T_i to reach T_f, undergoing a reduction of α per period. Upon substituting our values into the formula, determining the value of n can be accomplished using the following equation:

$$n = \frac{\ln(T_f/T_i)}{\ln(\alpha)}$$

Following this, we divide the resulting value of n by the total runtime of the algorithm. This computation provides the duration per temperature, signifying the allotted time for each temperature level of the algorithm to operate. Through this approach, T_i will converge to T_f within our desired time limit.

The rest of SA's parameters are found in a manner similar to MBO algorithm's parameters. The grid is outlined in Table 6, and its values were derived from [57; 58].

Table 6: SA algorithm parameter grid & values determination

$\boldsymbol{T_f}$	$\boldsymbol{\alpha}$
0.01	0.95
0.1	0.975
1	**0.98**
10	0.99

Duman and Duman [54] introduced a noteworthy enhancement to Simulated Annealing aimed at rescuing the algorithm from sub-optimal search spaces. By allowing

the algorithm to incorporate non-improving moves at certain stages, it may inadvertently enter areas where the optimal solution does not reside. To address this issue, the authors propose a mechanism wherein, once the algorithm reaches, for instance, 90% of its iteration limit, it reverts to the best solution encountered up to that point. Subsequently, it exclusively it converges to a steepest decent algorithm, and only accepts solutions that yield improvement. We similarly applied this strategy to enhance the performance of the algorithm in navigating the search spaces.

4.2.3.3 Hybrid MBO-SA

The MBO algorithm relies on a steepest descent approach, leveraging a flock of birds (solutions) with collaborative searching. It exhibits a steadfast nature, refusing to undertake a direction unless improvements are guaranteed. The SA on the other hand accepts worse solutions based on a temperature scheme that balances between the exploration and exploitation stages. Although, it traverses a single solution path across the iterations.

Simultaneous navigation through multiple solution spaces with a temperature controlled frame-work may foster an enhanced search capability. To harness the strengths of both algorithms, we propose a hybrid algorithm. This includes integrating MBO's navigational attributes through a flock of birds, with SA's capacity for exploration and exploitation-driven solution acceptance.

While using the same parameters derived from MBO and SA algorithms, we outline an implementation guide for the proposed algorithm below:

1. Initialization (set the algorithm's parameters and construct a V flight shape):

    - n: The number of initial solutions (must be an odd number)
    - k: The number of neighboring solutions to generate for the leader
    - x: The number of neighbors to pass downwards
    - m: The number of iterations maintaining the same leader solution

- T_i (initial temperature): The temperature the algorithm starts with
- T_f (final temperature): The temperature during the last iteration
- α (cooling rate): A factor that takes a value between 0 and 1, used to reduce the temperature.
- V flight formation: Arrange n many initial birds in a V-shaped structure, with one bird (the leader bird) positioned at the tip, and others of equal size positioned symmetrically on both the left and right sides. The initial birds are obtained randomly or through a heuristic algorithm.

2. Main loop:

   Generate k neighboring solutions from the leader bird. If the best neighbor is superior (cheaper in our case), it is replaces the leader bird. Otherwise, it overtakes the leader bird based on the Metropolis criterion. Pass x unused neighbors to both the left and right sides. For each bird on each side individually, generate $k - x$ neighboring solutions. If the best neighbor is superior, adopt it. Otherwise, evaluate whether it should be adapted based on the Metropolis criterion. If the best neighbor is not superior and fails the criterion, assess the best passed solution. If the passed solution surpasses the current bird, adapt it; otherwise, reevaluate its adaptation based on the Metropolis criterion. Subsequently, update the passed solutions with the unused neighbors of the current birds and pass them down to the preceding bird. Repeat this process until the last bird on each side is reached.

   Once this scheme is completed m times, pass the leader bird to the bottom of either side, with the first bird from the same side becoming the new leader. Additionally, reduce T_i by α. The whole scheme mentioned is repeated until T_i becomes smaller than or equal to T_f.

3. Report best bird found

The initial birds of the algorithm are generated in the same manner as the MBO algorithm. The proposed algorithm is similar to the SA in the sense that it keeps

iterating until its initial temperature T_i reaches the final temperature T_f. Again here, we cannot determine an m value that would allow T_i to reach T_f in the desired run time limit. Thus similar to SA, m is composed of a run time per leader bird that is problem based, instead of having a numerical limit. To decide on the initial temperature T_i of the algorithm, we followed the same methodology applied in the SA algorithm, although we set lower percentages. In Phase II, we set the acceptance probability for solutions deviating by 25% in cost to be 1%, and in Phase III, solutions deviating by 1% in cost are accepted with a probability of 1%. These reduced probabilities stem from the hybrid algorithm employing multiple agents for searching through solution spaces. Consequently, tightening their acceptance criteria was deemed more beneficial than in comparison with SA, where only a single agent is utilized for the search process. The rest of the parameters are derived from the findings in the original MBO and SA algorithms mentioned.

Given that in this algorithm, like SA, occasionally accepts non-improving moves, it may also explore solution spaces where the optimal solution does not exist. Just as we discussed a modification in SA to improve its navigation through search spaces, we've implemented a similar strategy here. When the algorithm reaches 90% of its total iteration limit, all birds adopt the best solution each of them encountered so far and only accept improving solutions thereafter.

Figure 5 demonstrates the behaviors of the three metaheuristics discussed, when applied to a minimization problem, where they converge at an objective value of 30. Simulated annealing exhibits flexibility as it transitions between solutions, utilizing a single solution as a guide, while tolerating some deterioration in solutions. On the other hand, the Migrating Birds Optimization algorithm employs multiple solutions to navigate the solution space, strictly descending towards improving solutions. The Hybrid metaheuristic proposed incorporates both multiple solutions navigating the solution space and the acceptance of weaker solutions, thereby modifying its stubborn

behavior.

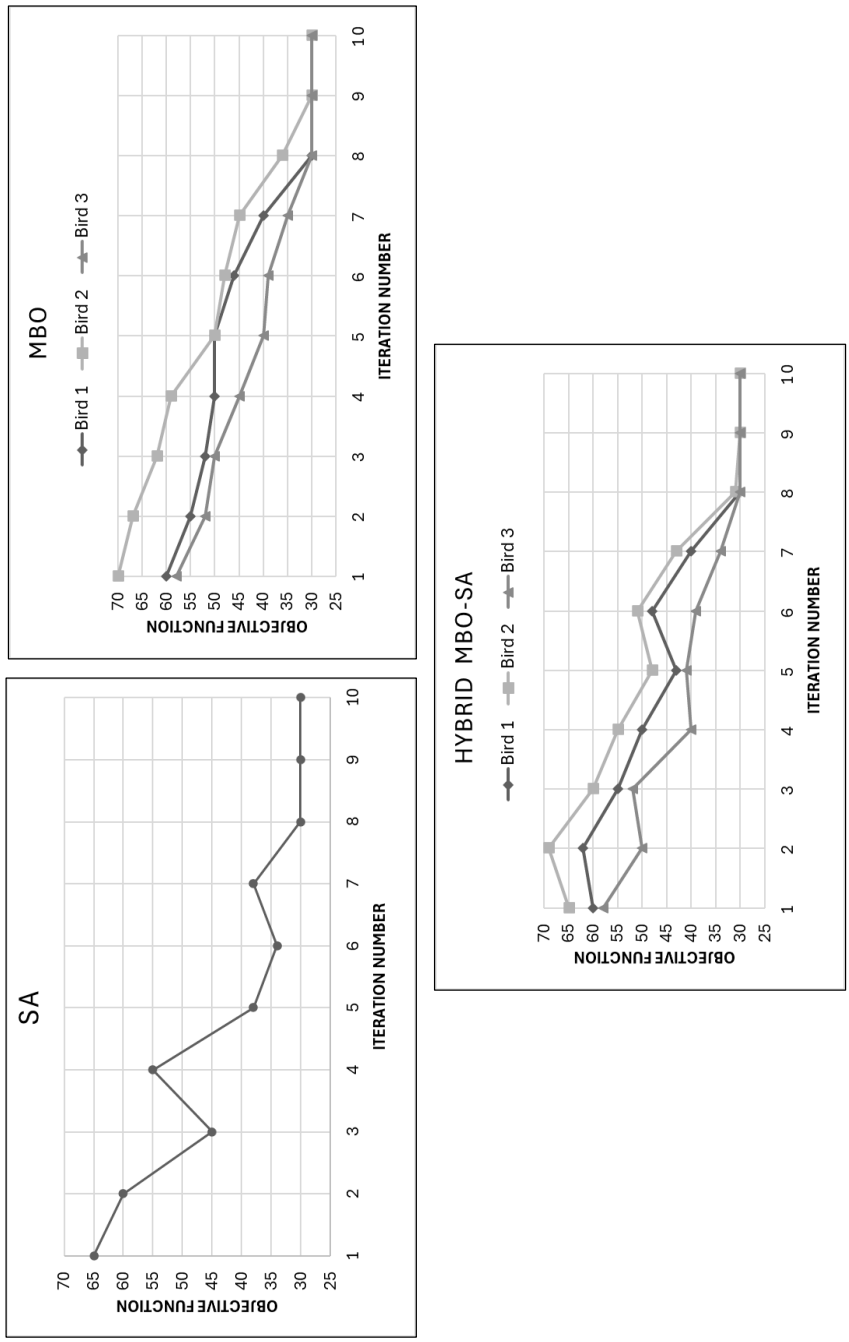

Figure 5: The behavior of the metaheuristics through the iterations

CHAPTER V

DATA DETAILS AND MERGING STEPS

In this chapter, we provide an overview of the problem sets we addressed. Additionally, we examine two merging steps done to the customers' products, each implemented with particular objectives in mind.

The distributor provided historical data concerning problems encountered over three distinct days. These problems encompassed daily orders, comprising multiple product requests from customers located across various cities in Turkey. Each day's problem set must be solved within the same day at the morning.

To streamline problem-solving, the distributor partitioned the map of Turkey into 17 regions and subsequently assigned each customer to one of these regions based on their location. Each region is then addressed as an independent problem requiring resolution. Below, we outline two merging steps applied to customer products, integral to the solving approaches that will be discussed in the following chapter.

5.1 Merge 1

The first merging step is used to reduce the number of entities being saved for the products each customer requested. This merging step is related with the constraint limiting the number of vehicles that can deliver to each customer site.

If the total storage requirements can be met by a single vehicle, all products are merged and treated as a single entity. This consolidation involves summing up the collective weight and pallet requirements of the products. The implementation of this step helps in reducing the number of entities to work with, while not losing any optimal solution since all the customer's products were expected to be delivered on one vehicle, given that they can fit.

This strategy cannot be adopted for customers ordering products that in total exceed the capacity of a single vehicle, as such products would be not fit in any vehicle, including the largest one.

5.2 Merge 2

In this merging step, we concentrate on customers who have placed an order for products of the same type (normal/fridge) that collectively exceed the capacity of a single vehicle. This allows us to distribute their products across multiple vehicles. The number of vehicles capable of sharing these customers' products depends on the Split allowed parameter discussed earlier in the problem definition.

Consider a scenario where we focus on a single customer with m products and an allowed split between different vehicles of k. In such instances, using fewer than m vehicles is not feasible due to capacity limitations, and using more than m vehicles is not permitted from the customer's side.

Finding the number of combinations to distribute m distinct items into exactly k bins mirrors the calculation of the Stirling numbers of the second kind. These numbers represent the count of distinct ways to partition a set of m elements into k non-empty subsets. The notation for Stirling numbers of the second kind, with n elements and m subsets, is denoted as $S(m, k)$. The number of possible combinations can be calculated using the following formula [59]:

$$S(m,k) = \frac{(-1)^k}{k!} \sum_{i=1}^{k} (-1)^i \binom{k}{i} i^m$$

This also leads us to explore potential combinations to distribute m products among exactly k vehicles. Some combinations may prove infeasible due to the capacity limitations of the vehicles. For instance, a combination that includes two large products together.

If we devise a method to allocate m products into k bins, each accommodating a portion of the products, we can effectively reduce the number of products assigned to

Table 7: Nomenclature of Merge 2 mathematical model

Sets	
$P = \{1, 2, \ldots, m\}$	Set of products that belong to the same customer and of the same type.
$B = \{1, 2, \ldots, k\}$	Set of bins.
Parameters	
Pl	Maximum number of pallets any bin can have.
W	Maximum weight any bin can have.
pl_p	Pallet need for product $p \in P$.
w_p	Weight need for product $p \in P$.
Decision variables	
Z_{pb}	1 if product $p \in P$ is assigned to bin $b \in B$, 0 o/w.

each customer from m to k. More importantly, we wouldn't have to concern ourselves with the maximum number of vehicles permitted to stop at a customer's location anymore, as we prearrange the products in a manner that guarantees feasibility at all times. The stated reasons reduce the size of the problems and streamline the implementation during the developed algorithms, ultimately reducing the complexity of the problem.

Out of all the combinations, we decided to choose the one that maximizes the utilization of all $k-1$ bins while keeping one bin utilized the least. The rationale behind this strategy is to fill up $k-1$ vehicles to directly assign them to the customer location, leaving the last vehicle with room for additional products belonging to other customers.

To find the combination in question, we utilize a simple and efficient mathematical model that is solved using an exact solver. This model organizes the products of each customer whose cumulative capacity for the same product type exceeds that of a single vehicle. The model's terminologies are outlined in Table 7, with its formulation provided below:

Objective Function

$$\textbf{max} \qquad \sum_{b \in B} \left(\frac{\sum_{p \in P} pl_p Z_{pb}}{Pl} \right)^2 + \left(\frac{\sum_{p \in P} w_p Z_{pb}}{W} \right)^2 \tag{1}$$

Constraints

$$\sum_{b \in B} Z_{pb} = 1 \qquad \forall p \in P \tag{2}$$

$$\sum_{p \in P} w_p Z_{pb} \leq W \qquad \forall b \in B \tag{3}$$

$$\sum_{p \in P} pl_p Z_{pb} \leq Pl \qquad \forall b \in B \tag{4}$$

$$Z_{pb} \in \{0, 1\} \qquad \forall p \in P, b \in B \tag{5}$$

The objective function (1) aims to maximize the utilization of all the vehicles in terms on weight and pallets. Since the utilization has an exponential term, the model will arrange the products in a way that fills up more vehicles than the rest. Through constraint (2), we ensure that each product is assigned to exactly one bin. Constraints (3) and (4) are used to limit the total weight and pallets loaded on each bin $b \in B$ respectively. Constraints (5) ensures the binary behavior of the decision variables.

The *k-1* bins that are utilized the most resulted out of the exact solver's solution are disintegrated out of the problem with the products assigned on them for direct delivery to the associated customer location, and the products on the least utilized bin undergo the Merge 1 step and are treated as a single product. This product is then addressed alongside products from other customers within the problem.

While this approach aids in simplifying the problem, it risks compromising optimal solutions. By analyzing the products of the customers separately before merging them with others, we may inadvertently overlook more efficient combinations. A potentially more cost-effective strategy could involve maintaining some unused capacity across all m bins. This surplus capacity enables the loading of products from different customers, potentially reducing the overall number of vehicles required, with the k many bins serving more customers and making more stops.

Table 8 details the order requests received over the three days. Under each day, we represent how the customers and their products are distributed among the 17

regions. For each region's row, we present the number of customers that must be served under the 'C' notation, and the quantity of the products they requested under the 'P' column. We also show the the size of the products after applying the first merging step under 'M1', and the size after applying the first and second merging steps under the 'M1&2' column. When the product sizes in 'M1' and 'M1&2' match, it means none of the customers in that problem required more than one vehicle for service, making the second merging step unnecessary.

In Chapter 4.2.3, all experiments — from assigning neighborhood search probabilities to fine-tuning parameters in the metaheuristics — were performed on the largest problem in terms of customers and products to be served (problem 17 on Day 1). This choice stems from the recognition that metaheuristic algorithms excel when tackling problems with the greatest computational complexities [60]. Hence, we trained our algorithms on the largest problem available.

In the following chapter, we provide numerical results for each solution approach employed across the problem sets, along with details specifying the merging applied in each case.

Table 8: The Three days problem sets

	Day 1				**Day 2**				**Day 3**			
#	**C**	**P**	**M1**	**M1&2**	**C**	**P**	**M1**	**M1&2**	**C**	**P**	**M1**	**M1&2**
1	6	22	6	6	7	36	9	9	6	59	9	9
2	6	54	9	9	7	37	9	9	6	64	8	8
3	8	36	10	10	7	49	10	10	6	111	9	9
4	10	25	10	10	9	37	12	12	8	96	13	13
5	11	64	14	14	11	34	13	13	11	114	18	18
6	12	79	17	17	11	120	18	18	11	213	18	18
7	12	70	16	16	11	55	17	17	12	71	18	18
8	13	95	57	16	12	46	17	17	12	243	20	20
9	13	151	20	20	14	104	20	20	12	123	20	20
10	14	158	19	19	14	73	21	21	15	188	24	24
11	15	89	19	19	14	105	18	18	15	85	24	24
12	15	60	19	19	16	113	42	27	15	104	24	22
13	16	85	35	26	16	88	24	24	15	224	25	25
14	16	117	21	21	17	94	40	29	16	151	30	25
15	16	143	37	24	18	127	29	29	18	107	58	24
16	25	119	42	34	23	100	33	29	24	325	42	42
17	28	232	51	43	27	226	39	39	26	153	40	40
Total	236	1599	402	323	234	1444	371	341	228	2431	400	359

CHAPTER VI

STUDY OF COMPLEXITY, SOLVING PHASES, RESULTS, AND COMPARATIVE ANALYSIS

In this chapter, we begin by studying the complexity of the problem with varying sizes using the exact method. We then proceed to outline different solving phases, detailing the solution method used, the type of merging applied to the problem data, and the allocation of runtime for each phase. Finally, the chapter concludes with a comparative analysis across the different solving phases and methods used, aiming to determine the most effective approach. All the solution approaches outlined in this thesis report were implemented using Java programming on a laptop equipped with an Intel Core i5-12500H processor running at 2.50 GHz, and 16GB of RAM. For exact solutions, the Gurobi optimization software (version 10.0.1) commercial solver was employed.

6.1 Problem study

To have a cohesive understanding of the problem in hand, we evaluate the performance of the mathematical model developed in this study across various problem sizes using an exact solver. We address the problems in each region over the three-day dataset provided in the previous chapter, with a runtime limit of 30 minutes allocated for each region.

In this subsection, we exclusively utilize the data resulting from the first merging step. As previously mentioned, the first merging step effectively reduces the problem size while preserving optimal solutions. However, implementing the second step may lead to the loss of optimal solutions, hindering our ability to establish a realistic lower

bound for the problems.

Table 9 displays the minimum costs determined by the exact solver across the 17 problems on each of the three days. The first column designates the problem number for each day. The 'C' column indicates the count of customers awaiting delivery, while 'M1' represents the number of products they requested, specifically after undergoing the first merging step. The 'Ex1' column indicates the cost of the solution provided by the exact solver. Additionally, 'LB' represents the lower bound estimated by the exact solver. When the lower bound and the cost are the same, it signifies that the optimal solution has been found. Finally, the last column illustrates the percentage difference between the cost found and the lower bound.

Table 9: Results of the Exact solver with a runtime limit of 30 minutes

	Day 1					**Day 2**					**Day 3**				
#	**C**	**M1**	**Ex1**	**LB**	**Dev**	**C**	**M1**	**Ex1**	**LB**	**Dev**	**C**	**M1**	**Ex1**	**LB**	**Dev**
1	6	6	4030	4030	0%	7	9	7190	7190	0%	6	9	7537	7537	0%
2	6	9	2401	2401	0%	7	9	6551	6551	0%	6	8	4357	4357	0%
3	8	10	3546	3546	0%	7	10	2745	2745	0%	6	9	3720	3720	0%
4	10	10	2888	2888	0%	9	12	2734	2734	0%	8	13	3177	3177	0%
5	11	14	4033	3852	5%	11	13	3896	3699	5%	11	18	6203	5873	6%
6	12	17	3780	3411	11%	11	18	9543	8135	17%	11	18	9245	8255	12%
7	12	16	7890	7681	3%	11	17	1262	1262	0%	12	18	12533	12170	3%
8	13	57	12015	12015	0%	12	17	12673	12526	1%	12	20	17884	16926	6%
9	13	20	16374	12541	31%	14	20	4514	4514	0%	12	20	1983	1983	0%
10	14	19	3766	2730	38%	14	21	14463	14462	0%	15	24	6114	6092	0%
11	15	19	7236	6708	8%	14	18	20117	17471	15%	15	24	12871	12867	0%
12	15	19	4560	4546	0%	16	42	17689	15315	16%	15	24	8670	7829	11%
13	16	35	15005	12218	23%	16	24	2843	2441	16%	15	25	15805	15635	1%
14	16	21	6217	6149	1%	17	40	9449	7902	20%	16	30	22710	21227	7%
15	16	37	12974	11910	9%	18	29	19798	15345	29%	18	58	6060	5684	7%
16	25	42	11810	5949	99%	23	33	5678	4070	40%	24	42	17581	8615	104%
17	28	51	20829	5065	311%	27	39	18354	7365	149%	26	40	4695	2448	92%
Total	236	402	139354	107640	539%	234	371	159499	133727	308%	228	400	161145	144395	249%

In implementing this solving, three primary goals were considered. Firstly, it

established lower bound values for all problems. By extending the runtime, our goal was to refine and tighten the lower bounds found. This lower bound is used for comparison in the next solving phases.

Another advantage of these runs is that they facilitated our comprehension of the factors contributing most significantly to problem difficulty. For each problem, we know two metrics: the number of customers and the number of products. The aim was to ascertain which of these metrics has a greater impact on the difficulty level of a particular problem.

The Pearson correlation coefficient measures the linear relationship between two variables, ranging from -1 to 1. Values closer to 1 indicate a strong positive correlation, closer to -1 indicate a strong negative correlation, and around 0 suggest little to no correlation [61]. The following formula is used to find the correlation coefficient between two variables, say X and Y:

$$Correl(X,Y) = \frac{\sum(X_i - \bar{X}) \times (Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \times \sum(Y_i - \bar{Y})^2}}$$

Utilizing this metric, we examined the correlation between the percentage deviation of each problem and its associated number of customers and products. Notably, the analysis unveiled a positive correlation of 73% between the number of customers and the percentage deviation. On the other hand, the correlation between the number of products and the percentage deviation was comparatively weaker, standing at 53%. This implies that the number of customers serves as a superior indicator of a problem's difficulty. Moreover, as the number of customers increases, we anticipate a greater challenge in finding the optimal solution for the problem. Having established that the problem's number of customers is the primary indicator of complexity, we will prioritize its consideration when making comparisons.

Finally, this solving provided valuable insights into the solver's capacity to solve

the proposed mathematical model across problems of varying sizes. Figure 6 illustrates the average percentage deviation between the cost and the lower bound, categorized by the encountered customer sizes within our problem set.

The observations show that the solver effectively handled problems with 10 or fewer customers, finding optimal solutions within the provided time-frame. However, for problems with 11 to 18 customers, we begin to see a gap between the found costs and the lower bound, ranging between 3% and 20%. The gap became notably wider and began to escalate exponentially for problem sets involving more than 18 customers. Notably, with problem sets of 28 customers, the gap expanded drastically, reaching a significant 311%. This pronounced increase underscores the importance of implementing strategies to reduce problem size or employing non-exact solving approaches.

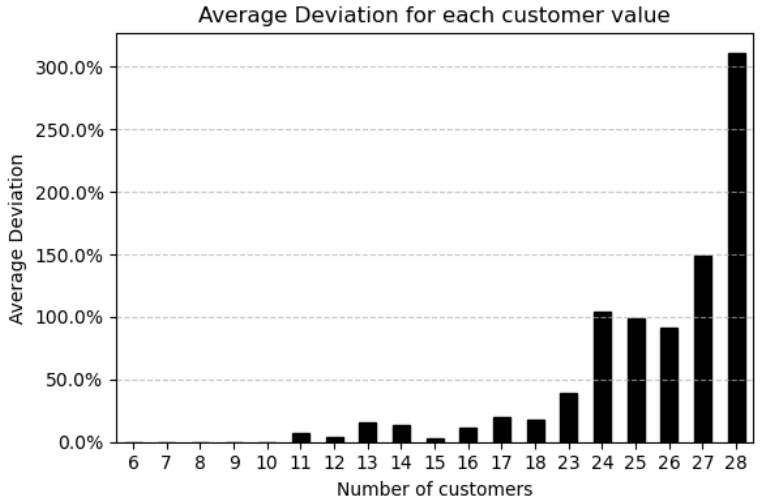

Figure 6: Percentage deviation per problem's customer size

6.2 Solving Phases

In this subsection, we explore different solving phases through the manipulation of the problem set, over our solving methodologies. Metaheuristic algorithms demonstrate non-deterministic behavior due to the random decisions they make, resulting

in outcome variations. To address this, we conducted three independent runs for each problem and computed the average outcome as the final result in each of the following solving phases.

Upon receiving orders from customers in the morning of each day, the distributor allocates them across the 17 regions. They then have a one-hour window before commencing their operations, during which they plan for all the regions. To align with the distributor's time frame, the following solving phases are all bound to one hour of run-time for each day's problem.

6.2.1 Phase I: Solve the problem based on the 17 regions

The first solving approach entails evaluating the exact and non-exact solution tools developed across the 17 regions for each day. Additionally, we compare these results with those obtained from the distributor's current solving approach.

In this solving phase, our non-exact algorithms are applied to the data resulting from both the first and second merging steps. These merging steps are crucial to our heuristic/metaheuristic algorithms. Without them, managing the distribution of each customer's products among multiple vehicles would be necessary and very challenging. This task becomes particularly complex during the execution of Clarke and Wright's algorithm and the neighborhood search moves within the metaheuristics, since ensuring that a customer's products are not distributed over the maximum allowable number of vehicles becomes crucial. Thanks to the merging steps, monitoring the distribution of products for each customer among vehicles is no longer necessary as it ensures feasibility in a manner that adheres to the maximum number of vehicles permitted for distributing each customer's products.

Since the two merging steps result in problems with reduced sizes and a relaxation of the constraint governing the number of vehicles serving each customer, we also assess the performance of the exact method in this solving phase. Finally, we employed

the exact method to solve the problem set, having solely the first merging step. We anticipate observing a drawback in the results compared to the previous subsection, due to the altered allocated run time.

The total allocated runtime to solve all 17 regional problems for each day is one hour as previously discussed. The question arises regarding how to distribute this time limit among the 17 regions. In the problem complexity discussions, we concluded that the number of customers is the most reliable indicator of the problem's complexity. Hence, one potential strategy is to allocate available time linearly, considering the number of customers associated with each problem. Although this would not be a wise allocation of time considering the exponential increase in the number of possible solution combinations and the expanding solution space as problem sizes grow. As the solution space expands, the amount of time needed to explore it also increases. As a result, we opted to allocate time based on a function with exponential growth. Initially, we assess the number of customers in each region. Then, we amplify this count by squaring it, whereby the square function's impact grows in proportion to the number of customers present. Subsequently, we sum up all the squared customer values and divide the one hour by the result. This yields the time allocated per customer. Consequently, each region will have a total run time calculated as its number of customers squared multiplied by the time allocated per customer. Table 10 shows the allocation of the available time over the 17 regional problems. In the 'C' column, we denote the number of customers for each problem, while the 'C^2' term represents the square of this value. Additionally, in the 'Secs' column, we provide the runtime for each problem, measured in seconds.

Table 10: Time allocated for each region within each day's problem (in seconds)

	Day 1			Day 2			Day 3		
#	C	C^2	Secs	C	C^2	Secs	C	C^2	Secs
1	6	36	34.05	7	49	47.60	6	36	36.22
2	6	36	34.05	7	49	47.60	6	36	36.22
3	8	64	60.54	7	49	47.60	6	36	36.22
4	10	100	94.59	9	81	78.68	8	64	64.39
5	11	121	114.45	11	121	117.54	11	121	121.74
6	12	144	136.21	11	121	117.54	11	121	121.74
7	12	144	136.21	11	121	117.54	12	144	144.89
8	13	169	159.85	12	144	139.88	12	144	144.89
9	13	169	159.85	14	196	190.39	12	144	144.89
10	14	196	185.39	14	196	190.39	15	225	226.38
11	15	225	212.82	14	196	190.39	15	225	226.38
12	15	225	212.82	16	256	248.68	15	225	226.38
13	16	256	242.14	16	256	248.68	15	225	226.38
14	16	256	242.14	17	289	280.73	16	256	257.57
15	16	256	242.14	18	324	314.73	18	324	325.99
16	25	625	591.17	23	529	513.87	24	576	579.54
17	28	784	741.57	27	729	708.15	26	676	680.16
Total	236	3806	3600	234	3706	3600	228	3578	3600

Table 11 outlines the results of each method employed, along with its percentage deviation from the lower bound obtained in the previous subsection. The table displays results over the three days, with each row corresponding to a regional problem number, denoted by '#'. 'C' indicates the number of customers to be served for each problem, 'M1' signifies the size of requested products after the first merging step, and 'M1&2' represents the size of products after both merging steps. 'Secs' denotes the permissible runtime in seconds, derived through the previously mentioned method. As discussed, during this solving phase, the exact solver is employed across the problem set. Two executions of the exact solver occur: the first after the first merging step, labeled 'Ex1' and the second following both merging steps, labeled 'Ex2' The 'Dist' column illustrates the cost determined by the distributor, while 'C&W' showcases results obtained from Clarke and Wright's algorithm. 'MBO', 'SA', and 'MBOSA' denote outcomes from the Migrating Birds Optimization algorithm, Simulated Annealing, and the hybrid algorithm that integrates both metaheuristics, respectively. The 'LB' column presents the lower bound acquired in previous subsection. After that

column, we illustrate the percentage difference between the cost identified by each method and the lower bound. In Table 12, we summarized the results by grouping the problems according to their customer sizes and indicating the average percentage deviation of each method for the same customer size.

From Table 12, it's evident that when product sizes remain unchanged after the first merging step M1 and the first and second merging steps M1&2, the results from Ex1 and Ex2 are identical as the problems are identical. However, when the second merging step reduces product sizes, it simplifies the problem's complexity, leading to Ex2 yielding better results with smaller percentage deviations. Furthermore, the second merging step's operation of more product prearrangement results in smaller products to be assigned in each problem, contributing to better Ex2 cost outcomes. For example, in Day 3 problem 12, where the product size changes only slightly from 24 to 22 products, the deviation between Ex1 and Ex2 is just 2%. Conversely, in problem 15 on the same day, where the product size decreases significantly from 58 to 24 products, the deviation difference is 8%. Although there's concern that the second merging step might compromise optimal solutions due to its prearrangement of products, it's evident that Ex2 results consistently equal or outperform Ex1 results. This underscores the success of the merging step in simplifying problems and yielding better outcomes, while also demonstrating that the logic behind the second merging step effectively preserves optimal/sub-optimal solutions.

Through observing Table 12, it's evident that the percentage deviation of all methods significantly increases on average as the customer size of the problems increases. Furthermore, we note that both methods employing an exact solver (Ex1 and Ex2) successfully find optimal solutions for problems with customer sizes up to 10, outperforming the other methods slightly. This establishes the exact solver as the optimal approach for small problems. Beyond that, metaheuristics consistently deliver the

Table 11: Phase I results

Day 1																			
#	C	M1	M1&2	Secs	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA	LB	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA
1	6	6	6	34	4030	4030	4030	4030	4030	4030	4030	4030	0%	0%	0%	0%	0%	0%	0%
2	6	9	9	34	2401	2401	2401	2401	2401	2401	2401	2401	0%	0%	0%	0%	0%	0%	0%
3	8	10	10	61	3546	3546	3587	3678	3546	3546	3546	3546	0%	0%	1%	4%	0%	0%	0%
4	10	10	10	95	2888	2888	2893	2937	2888	2893	2888	2888	0%	0%	0%	2%	0%	0%	0%
5	11	14	14	114	4033	4033	4029	4657	4033	4033	4033	3852	5%	5%	5%	21%	5%	5%	5%
6	12	17	17	136	3780	3780	3822	3789	3744	3744	3744	3411	11%	11%	12%	11%	10%	10%	10%
7	12	16	16	136	7890	7890	7979	8369	7902	7902	7902	7681	3%	3%	4%	9%	3%	3%	3%
8	13	57	16	160	13200	12442	13261	13349	12231	12231	12231	12015	10%	4%	10%	11%	2%	2%	2%
9	13	20	20	160	16421	16421	17090	17545	16374	16374	16374	12541	31%	31%	36%	40%	31%	31%	31%
10	14	19	19	185	3766	3766	4485	3584	3510	3510	3510	2730	38%	38%	64%	31%	29%	29%	29%
11	15	19	19	213	7879	7879	8021	8226	7197	7197	7197	6708	17%	17%	20%	23%	7%	7%	7%
12	15	19	19	213	4560	4560	5044	5180	5180	4754	4754	4546	0%	0%	11%	14%	14%	5%	5%
13	16	35	26	242	15026	15005	14844	15012	14875	14591	14591	12218	23%	23%	21%	23%	22%	19%	19%
14	16	21	21	242	6607	6607	7009	6660	6608	6329	6329	6149	7%	7%	14%	8%	7%	3%	3%
15	16	37	24	242	14490	13402	12963	14999	13383	13227	13227	11910	22%	13%	9%	26%	12%	11%	11%
16	25	42	34	591	11871	11680	10649	11002	10460	10457	10429	5949	100%	96%	79%	85%	76%	76%	75%
17	28	51	43	742	23214	22807	19665	19902	18955	18654	18783	5065	358%	350%	288%	293%	274%	268%	271%
Total	236	402	323	3600	145603	143137	141772	145318	137317	135873	135970	107639	625%	598%	575%	600%	491%	468%	470%
Day 2																			
#	C	M1	M1&2	Secs	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA	LB	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA
1	7	9	9	48	7190	7190	7190	7190	7190	7190	7190	7190	0%	0%	0%	0%	0%	0%	0%
2	7	9	9	48	6551	6551	6731	6603	6551	6551	6551	6551	0%	0%	3%	1%	0%	0%	0%
3	7	10	10	48	2745	2745	3196	3196	3196	3196	3196	2745	0%	0%	16%	16%	16%	16%	16%
4	9	12	12	79	2734	2734	2744	2744	2744	2744	2744	2734	0%	0%	0%	0%	0%	0%	0%
5	11	13	13	118	3896	3896	3904	3922	3896	3896	3896	3699	5%	5%	6%	6%	5%	5%	5%
6	11	18	18	118	9543	9543	9513	9672	9543	9543	9543	8135	17%	17%	17%	19%	17%	17%	17%
7	11	17	17	118	1262	1262	1509	1335	1262	1262	1262	1262	0%	0%	20%	6%	0%	0%	0%
8	12	17	17	140	13028	13028	12853	13137	12864	12864	12864	12526	4%	4%	3%	5%	3%	3%	3%
9	14	20	20	190	4872	4872	4514	4692	4514	4514	4514	4514	8%	8%	0%	4%	0%	0%	0%
10	14	21	21	190	14465	14465	14959	14764	14465	14465	14465	14462	0%	0%	3%	2%	0%	0%	0%
11	14	18	18	190	20117	20117	19060	20008	18202	18202	18202	17471	15%	15%	9%	15%	4%	4%	4%
12	16	42	27	249	19353	17718	17437	19383	17545	17480	17480	15315	26%	16%	14%	27%	15%	14%	14%
13	16	24	24	249	2843	2843	2788	3024	2764	2764	2764	2441	16%	16%	14%	24%	13%	13%	13%
14	17	40	29	281	10258	10209	9309	10000	9402	9402	9402	7902	30%	29%	18%	27%	19%	19%	19%
15	18	29	29	315	19798	19798	17771	19676	18100	18100	18100	15345	29%	29%	16%	28%	18%	18%	18%
16	23	33	29	514	5973	5678	7226	6064	5475	5468	5468	4070	47%	40%	78%	49%	35%	34%	34%
17	27	39	39	708	18447	18447	17047	17040	16415	16287	16287	7365	150%	150%	131%	131%	123%	121%	121%
Total	234	371	341	3600	162779	161095	157752	162451	154128	153928	153928	133726	349%	330%	347%	359%	268%	266%	266%
Day 3																			
#	C	M1	M1&2	Secs	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA	LB	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA
1	6	9	9	36	7537	7537	7537	7537	7537	7537	7537	7537	0%	0%	0%	0%	0%	0%	0%
2	6	8	8	36	4357	4357	4357	4357	4357	4357	4357	4357	0%	0%	0%	0%	0%	0%	0%
3	6	9	9	36	3720	3720	3720	3720	3720	3720	3720	3720	0%	0%	0%	0%	0%	0%	0%
4	8	13	13	64	3177	3177	3177	3177	3177	3177	3177	3177	0%	0%	0%	0%	0%	0%	0%
5	11	18	18	122	6258	6258	6054	7175	6155	6155	6155	5873	7%	7%	3%	22%	5%	5%	5%
6	11	18	18	122	9245	9245	9140	10438	9256	9256	9256	8255	12%	12%	11%	26%	12%	12%	12%
7	12	18	18	145	12533	12533	12417	12570	12570	12570	12570	12170	3%	3%	2%	3%	3%	3%	3%
8	12	20	20	145	19255	19255	17594	17988	17884	17884	17884	16926	14%	14%	4%	6%	6%	6%	6%
9	12	20	20	145	1983	1983	2707	2567	2174	2320	2320	1983	0%	0%	36%	29%	10%	17%	17%
10	15	24	24	226	6496	6496	6877	7348	6793	6786	6786	6092	7%	7%	13%	21%	12%	11%	11%
11	15	24	24	226	12871	12871	13229	15782	12921	12921	12921	12867	0%	0%	3%	23%	0%	0%	0%
12	15	24	22	226	8816	8670	8212	9348	9061	9061	9061	7829	13%	11%	5%	19%	16%	16%	16%
13	15	25	25	226	16837	16837	15725	17389	16933	15892	15892	15635	8%	8%	1%	11%	8%	2%	2%
14	16	30	25	258	22766	22710	23126	25543	23140	22718	22718	21227	7%	7%	9%	20%	9%	7%	7%
15	18	58	24	326	6522	6069	6025	6524	6154	6112	6112	5684	15%	7%	6%	15%	8%	8%	8%
16	24	42	42	580	18675	18675	15225	17871	16051	15736	15736	8615	117%	117%	77%	107%	86%	83%	83%
17	26	40	40	680	4695	4695	5616	4723	4403	4243	4243	2448	92%	92%	129%	93%	80%	73%	73%
Total	228	400	359	3600	165745	165090	160737	174057	162284	160444	160444	144395	293%	283%	298%	397%	255%	243%	243%

lowest average percentage deviation, with the SA and the hybridized MBOSA algorithms leading, closely followed by the MBO algorithm. On another note, the Clarke and Wright algorithm yielded solutions with costs that do not significantly deviate from those produced by other methods employed. This indicates its success in constructing effective solutions, which serve as initial solutions for the metaheuristics.

Table 12: Average percentage deviation based on the customer size

C	Ex1	Ex2	Dist	C&W	MBO	SA	MBOSA
6	0%	0%	0%	0%	0%	0%	0%
7	0%	0%	6%	6%	6%	6%	6%
8	0%	0%	1%	2%	0%	0%	0%
9	0%	0%	0%	0%	0%	0%	0%
10	0%	0%	0%	2%	0%	0%	0%
11	8%	8%	10%	17%	7%	7%	7%
12	6%	6%	10%	11%	6%	7%	7%
13	20%	17%	23%	26%	16%	16%	16%
14	15%	15%	19%	13%	8%	8%	8%
15	7%	7%	9%	18%	10%	7%	7%
16	17%	14%	14%	21%	13%	11%	11%
17	30%	29%	18%	27%	19%	19%	19%
18	22%	18%	11%	22%	13%	13%	13%
23	47%	40%	78%	49%	35%	34%	34%
24	117%	117%	77%	107%	86%	83%	83%
25	100%	96%	79%	85%	76%	76%	75%
26	92%	92%	129%	93%	80%	73%	73%
27	151%	151%	132%	131%	123%	121%	121%
28	358%	350%	288%	293%	274%	268%	271%

6.2.2 Phase II: Solve the problem based on the days

While the distributor's plan to divide customers among the 17 regions based on their locations makes logistical sense, as it groups customers with proximity to each other for efficiency, it may inadvertently exclude potentially more cost-efficient solutions. This is because there could be instances where it's economically advantageous to have customers from different regions served together. In this solving phase, we address each day's problem without segregating the customers among different sub-problems, rather tackling them as one whole problem. Employing this methodology restricts us to utilizing non-exact solutions as a solving tool, as the exact method surpassed the memory limits of the testing machine when confronted with the enormous problem instance at hand. Once more, in this solution phase, we apply the first and second merging steps. The allotted run time for each day's problem remains one hour.

Table 13 illustrates the costs obtained through the non-exact approaches to address the encountered problems over the three days. As expected, the Clarke and Wright algorithm produced results divergent from the metaheuristics, aligning with its

heuristic nature that provides swift solution derivation at the cost of an exploitative-based search. Notably, in coherence with Phase I, SA and the Hybrid MBO-SA stand out as the superior metaheuristics, with SA occasionally demonstrating stronger performance, and MBO-SA outperforming in other instances, while MBO shows a slight lag in performance.

Table 13: Phase II results

‘	**C**	**M1&2**	**C&W**	**MBO**	**SA**	**MBOSA**
Day 1	236	323	142,649	136,223	134,985	135,146
Day 2	234	341	161,071	152,005	151,327	150,988
Day 3	228	359	169,165	158,762	157,444	158,557

6.2.3 Phase III: Solve Phase I then Phase II

Solving a problem divided into 17 sub-problems can lead to optimally loss outcomes. Conversely, tackling the entire problem without any regional divisions results in dealing with an exceedingly large and complex problem, especially given the limited runtime. Therefore, we advocate for a middle-ground approach that tries balance between the two previous phases. In this solving approach, we divide the problem into two stages: Phase I and Phase II. This can be done by allocating the one-hour time limit between these stages. First, Phase I addresses the problem by breaking it down into its 17 regional sub-problems. The solution generated in this phase may have a higher overall cost compared to Phase I results alone, due to the shortened runtime. The solution obtained serves as a starting point for Phase II to refine further. Phase II introduces the customers from the different regions to each other, potentially finding a plan that serve customers that were in different sub-problems using the same vehicles, which happens to be more economical. Phase II operates during the remaining time limit in search for enhancements.

In order to approximate the optimal time allocation for each phase, we experimented with three different combinations of time ratios between the two phases on a single identical problem: 25%-75%, 50%-50%, and 75%-25%. The allocation of time

with a ratio of 25%-75% yielded the most effective overall cost, and thus was chosen for implementation.

Table 14 displays the costs associated with each method employed across the three days. Throughout this phase, it becomes evident that the Hybrid algorithm consistently outperforms the other two across all tested problem sets, with SA ranking second and MBO trailing behind.

Table 14: Phase III

	C	**M1&2**	**MBO**	**SA**	**MBOSA**
Day 1	236	323	134,639	134,082	133,454
Day 2	234	341	150,409	149,179	148,717
Day 3	228	359	155,158	154,563	154,465

6.2.4 Phase IV: Implement Phase III with a smart search

In Phase III, we first solve and form a solution for each region within a day's problem, by allocating a portion of run time for this process. The remainder is dedicated to facilitating inter-regional exchanges within the metaheuristic algorithms, aimed at identifying potential customer assignment swaps that could lead to a more cost-effective overall solution for the day. However, not all exchanges are worth exploring, as they may likely result in less economical outcomes. For instance, attempting to relocate a customer from Istanbul (located in the northwest of Turkey) to a vehicle serving customers in Mardin (located in the southeast of Turkey) may not be practical.

To enhance our strategy in this phase, we construct an information table for each region, outlining the viable regions for neighborhood exchanges. This implementation enables us to utilize runtime more effectively, focusing only on exchanges that align with geographical proximity of the regions. After consulting with a logistics expert from the distributor's side, they identified potentially related regions by evaluating the costs associated with reaching each one from another. These costs encompass bridge tariffs and the distance required to traverse.

Table 15: Phase IV results

	C	M1&2	MBO	SA	MBOSA
Day 1	236	323	135,281	134,152	133,775
Day 2	234	341	150,484	149,323	149,234
Day 3	228	359	154,985	154,916	154,731

Table 15 displays the costs obtained for each method across the three-day problems. The outcomes aligns with those of the previous phase, showcasing the hybrid metaheuristic as the top performer across all three days, trailed by SA and then MBO.

6.3 Phases and Metaheuristics comparison

Expanding on the efficacy and applicability of metaheuristics in addressing problem sets with varying sizes to establish optimal routing plans in preceding solving phases, this subsection presents a comparative analysis of their performance across the three solving phases and in relation to each other. Table 16 provides a summary of the solution costs generated by each metaheuristic across the solving phases. The cost indicated in each cell in the table is computed by aggregating the result of each metaheuristic throughout the three-day problem sets for their respective solving phase. The 'MH*' row displays the algorithm with the lowest overall plan cost for each solving phase, while the 'Phase*' column identifies the optimal solving phase to employ for each metaheuristic, based on the plan cost it produces.

Table 16: Comparison of the metaheuristics over the solving phases

	Phase I	Phase II	Phase III	Phase IV	Phase*
MBO	453,729	446,990	440,207	440,750	Phase III
SA	450,245	443,756	437,825	438,392	Phase III
MBOSA	450,342	444,691	436,638	437,739	Phase III
MH*	SA	SA	MBOSA	MBOSA	

Throughout all solving phases, it is evident that the MBO algorithm demonstrates weaker performance compared to the other two algorithms. This observation suggests that the MBO algorithm may be more prone to getting trapped in local optima due to its rigid nature, which only accepts solutions demonstrating immediate performance

improvements. Such limitations hinder its exploration ability and necessitate immediate enhancements to traverse the solution space effectively. Notably, the success of the hybrid MBOSA algorithm is evident as it consistently produces solutions superior to those of the classic MBO algorithm. The hybrid MBOSA algorithm delivered competitive results comparable to Simulated Annealing in the initial two phases, ultimately emerging as the top-performing algorithm for implementation the last two solution phases.

In analyzing the solving phases, its evident that the initial phase consistently exhibited the weakest performance. This is attributed to the segregation of customers, resulting in the loss of crucial network information essential for optimal solutions. Contrastingly, in Phase II, where efforts were made to retain all network relationships, performance across all algorithms improved noticeably. This underscores how the segregation in Phase I inadvertently erased cost-effective insights. Although, we operate an immediate engagement with a highly complex problem. Transitioning to Phase III, an emphasis is placed on establishing strong relationships among proximate customer groups. This step prioritizes gathering essential network information. Subsequently, through inter-group interactions, previously unseen insights emerge, enabling cost-effective adjustments within the network. During this phase, all metaheuristic algorithms presented an enhanced performance in comparison with the previous two phases. In the final phase, a drawback was observed for the three metaheuristics. This indicates that the implementation introduced in Phase IV did not enhance the exploration capability of the algorithms. The likely reason for this scenario is the inefficiency of the developed regional connections, as human judgment may have overlooked crucial network links that were omitted.

CHAPTER VII

SUMMARY, CONCLUSION, AND FUTURE WORK

In this thesis study, we investigate a novel Vehicle Routing Problem encountered by a distributor in Turkey. Classified as the Rich VRP variant, this problem is distinguished by its collection of sophisticated constraints and a multifaceted objective function. Within this realm, we encounter several established variants of the VRP documented in the literature.

Our problem-solving methodologies involves merging steps used to successfully reduce the complexity of the problem, followed by exact and non-exact solving solution approaches. The exact solution utilizes a commercial solver which uses a uniquely formulated Mixed-Integer Linear Programming mathematical model that comprehensively captures features seen in various VRP variants, in addition to its unique elements. Among the non-exact solving approaches, we utilize the Clarke and Wright saving algorithm. Additionally, we utilize Simulated Annealing and Migrating Birds Optimization algorithms, along with a novel algorithm that integrates the strengths of both metaheuristics. With the non-exact solving approach, we introduce methods that swiftly yet optimally provides the best vehicle type to use for each delivery, in addition to the optimal route that must be taken to reach customers' locations.

The distributor provided us with data detailing the problems encountered over three distinct days. Their approach to problem-solving entails partitioning each day's customers into 17 different sub-problems based on individual locations. We commenced by studying the problem's complexity through its developed mathematical formulation, and a commercial solver's capability in solving it. Following that, we delved into three solving phases, each manipulates the problem set distinctly.

The first phase solves the problems similar to the distributors approach, where each day's problem is divided into different sub-problems, the second phase tackles each day's problem as a whole. The third solving phase involves creating a successful initial solution via first solving each day's problem based on the 17 different sub-problems, then the sub-problems are introduced to each other and solved as one whole problem. In the final solving phase, we approach the problem akin to Phase III but strive to enhance the efficiency of the metaheuristics' search capabilities in the solution space.

The optimal solution approach entails tackling the problem through Phase III, which incorporates a hybrid metaheuristic algorithm combining Simulated Annealing with the Migrating Birds Optimization algorithms, and through a two-phased solving approach. The resulting solutions provide distributors with a comprehensive logistical plan, totaling 436,638 Turkish liras in cost over three days. In contrast, the distributor's current plan incurs a cost of 460,261 Turkish liras over the same period. This represents a 5.13% decrease in the distributor's overall operational cost, highlighting considerable potential savings, especially when managing large operational expenses.

Our problem featured a subset of customers that must be served using more than one vehicle, given that their total order exceed the total capacity of the largest vehicle type. This necessitated splitting their order over multiple vehicles. As a solution, in our second merging step, we pre-process their products, allocating a portion directly to their respective customer locations and distributing the remainder among vehicles alongside other customers' orders.

While this approach has shown success, it may inadvertently lead to a loss in optimality due to the predetermined decisions we make. As a potential avenue for future research, we propose the development of methodologies capable of addressing large-scale problems while maintaining control over the allocation of vehicles to each customer, without relying on product prearrangement. Such implementations could

reveal solution spaces with costs lower than those observed in this study. Exploring alternative metaheuristic or matheuristic algorithms can present a promising avenue for future research. Additionally, investigating the manipulation of the problem set across solving phases different from those conducted in this thesis report, opens up further opportunities for result enhancement.

REFERENCES

[1] L. H. HARPS, "Supply chain best practices: hitting the mark," *Inbound Logistics*, vol. 23, pp. 48–57, 2003.

[2] L. T. Biegler and I. E. Grossmann, "Retrospective on optimization," *Computers & Chemical Engineering*, vol. 28, no. 8, pp. 1169–1192, 2004.

[3] J. Kallrath, "Planning and scheduling in the process industry," *OR spectrum*, vol. 24, pp. 219–250, 2002.

[4] B. H. Korte and J. Vygen, *Combinatorial optimization*, vol. 1, pp. 405–406. Springer, 2011.

[5] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Computers & industrial engineering*, vol. 99, pp. 300–313, 2016.

[6] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera, and A. A. Juan, "Rich vehicle routing problem: Survey," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, pp. 1–28, 2014.

[7] W. Ho, G. T. Ho, P. Ji, and H. C. Lau, "A hybrid genetic algorithm for the multi-depot vehicle routing problem," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 4, pp. 548–557, 2008.

[8] J. Zhang and T. V. Woensel, "Dynamic vehicle routing with random requests: A literature review," *International Journal of Production Economics*, vol. 256, p. 108751, 2023.

[9] J. K. Lenstra and A. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[10] J. Li, Y. Ma, R. Gao, Z. Cao, A. Lim, W. Song, and J. Zhang, "Deep reinforcement learning for solving the heterogeneous capacitated vehicle routing problem," *IEEE Transactions on Cybernetics*, vol. 52, no. 12, pp. 13572–13585, 2021.

[11] S. MirHassani and N. Abolghasemi, "A particle swarm optimization algorithm for open vehicle routing problem," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11547–11551, 2011.

[12] C. Y. Ren, "Research on improved genetic algorithm for heterogeneous open vehicle routing problem," *Applied Mechanics and Materials*, vol. 55, pp. 859–862, 2011.

[13] G. Hasle and O. Kloster, "Industrial vehicle routing," *Geometric modelling, numerical simulation, and optimization: applied mathematics at SINTEF*, pp. 397–435, 2007.

[14] G. Laporte, Y. Nobert, and S. Taillefer, "A branch-and-bound algorithm for the asymmetrical distance-constrained vehicle routing problem," *Mathematical Modelling*, vol. 9, no. 12, pp. 857–868, 1987.

[15] R. Baldacci, M. Battarra, and D. Vigo, "Routing a heterogeneous fleet of vehicles," *The vehicle routing problem: latest advances and new challenges*, vol. 43, pp. 3–27, 2008.

[16] F. Li, B. Golden, and E. Wasil, "The open vehicle routing problem: Algorithms, large-scale test problems, and computational results," *Computers Operations Research*, vol. 34, no. 10, pp. 2918–2930, 2007.

[17] M. Dror and P. Trudeau, "Savings by split delivery routing," *Transportation Science*, vol. 23, no. 2, pp. 141–145, 1989.

[18] B. Domínguez-Martín, I. Rodríguez-Martín, and J.-J. Salazar-González, "The driver and vehicle routing problem," *Computers & Operations Research*, vol. 92, pp. 56–64, 2018.

[19] P. Pellegrini, D. Favaretto, and E. Moretti, "Multiple ant colony optimization for a rich vehicle routing problem: a case study," in *Knowledge-Based Intelligent Information and Engineering Systems: 11th International Conference, KES 2007, XVII Italian Workshop on Neural Networks, Vietri sul Mare, Italy, September 12-14, 2007. Proceedings, Part II 11*, pp. 627–634, Springer, 2007.

[20] P. Pellegrini, "Application of two nearest neighbor approaches to a rich vehicle routing problem," in *Technical report, TR/IRIDIA/2005-15*, p. 2005, IRIDIA, Université Libre de Bruxelles, 2005.

[21] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time wndow constraints," *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.

[22] A. Goel and V. Gruhn, "Large neighborhood search for rich vrp with multiple pickup and delivery locations," 2005.

[23] A. Goel and V. Gruhn, "Solving a dynamic real-life vehicle routing problem," in *Operations Research Proceedings 2005: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), Bremen, September 7–9, 2005*, pp. 367–372, Springer, 2006.

[24] A. Goel and V. Gruhn, "A general vehicle routing problem," *European Journal of Operational Research*, vol. 191, no. 3, pp. 650–660, 2008.

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

[26] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *biosystems*, vol. 43, no. 2, pp. 73–81, 1997.

[27] T. Stutzle and H. Hoos, "Max-min ant system and local search for the traveling salesman problem," in *Proceedings of 1997 IEEE international conference on evolutionary computation (ICEC'97)*, pp. 309–314, IEEE, 1997.

[28] 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.

[29] A. Goel, "A column generation heuristic for the general vehicle routing problem," in *International Conference on Learning and Intelligent Optimization*, pp. 1–9, Springer, 2010.

[30] A. Ceselli, G. Righini, and M. Salani, "A column generation algorithm for a rich vehicle-routing problem," *Transportation Science*, vol. 43, no. 1, pp. 56–69, 2009.

[31] L. Ruinelli, *Column generation for a rich vrp*. PhD thesis, Scuola universitaria professionale della Svizzera italiana, Manno, Switzerland, February 2011. Available: https://repository.supsi.ch/5148/1/IDSIA-08-11.pdf.

[32] C. G. Santillán, L. C. Reyes, M. L. M. Rodríguez, J. J. G. Barbosa, O. C. López, G. R. Zarate, and P. Hernández, "Variants of vrp to optimize logistics management problems," in *Logistics Management and Optimization through Hybrid Artificial Intelligence Systems*, pp. 207–237, IGI Global, 2012.

[33] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins, "Heuristics for multi-attribute vehicle routing problems: A survey and synthesis," *European Journal of Operational Research*, vol. 231, no. 1, pp. 1–21, 2013.

[34] Y. Xia and Z. Fu, "An adaptive tabu search algorithm for the open vehicle routing problem with split deliveries by order," *Wireless Personal Communications*, vol. 103, pp. 595–609, 2018.

[35] A. Azadeh and H. Farrokhi-Asl, "The close–open mixed multi depot vehicle routing problem considering internal and external fleet of vehicles," *Transportation Letters*, vol. 11, no. 2, pp. 78–92, 2019.

[36] K. M. Ferreira, T. A. de Queiroz, and F. M. B. Toledo, "An exact approach for the green vehicle routing problem with two-dimensional loading constraints and split delivery," *Computers & Operations Research*, vol. 136, p. 105452, 2021.

[37] Y. Zhang, H. Li, Z. Wang, and H. Wang, "A multi-objective learning whale optimization algorithm for open vehicle routing problem with two-dimensional loading constraints," *Mathematics*, vol. 12, no. 5, p. 731, 2024.

[38] V. Lesch, M. König, S. Kounev, A. Stein, and C. Krupitzer, "Tackling the rich vehicle routing problem with nature-inspired algorithms," *Applied Intelligence*, vol. 52, no. 8, pp. 9476–9500, 2022.

[39] M. Rajaei, G. Moslehi, and M. Reisi-Nafchi, "The split heterogeneous vehicle routing problem with three-dimensional loading constraints on a large scale," *European Journal of Operational Research*, vol. 299, no. 2, pp. 706–721, 2022.

[40] O. Kabadurmus and M. S. Erdogan, "A green vehicle routing problem with multi-depot, multi-tour, heterogeneous fleet and split deliveries: a mathematical model and heuristic approach," *Journal of Combinatorial Optimization*, vol. 45, no. 3, p. 89, 2023.

[41] B. S. Vieira, G. M. Ribeiro, and L. Bahiense, "Metaheuristics with variable diversity control and neighborhood search for the heterogeneous site-dependent multi-depot multi-trip periodic vehicle routing problem," *Computers & Operations Research*, vol. 153, p. 106189, 2023.

[42] A. K. Garside, L. Erlinda, and I. Amallynda, "Solving heterogeneous fleet vehicle routing problem with clarke wright saving heuristic and genetic algorithm," in *AIP Conference Proceedings*, vol. 2927, AIP Publishing, 2024.

[43] F. Tamke, "A real-world cost function based on tariffs for vehicle routing in the food retailing industry," in *Logistics Management: Contributions of the Section Logistics of the German Academic Association for Business Research, 2015, Braunschweig, Germany*, pp. 229–239, Springer, 2015.

[44] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[45] S. Deb, S. Fong, Z. Tian, R. K. Wong, S. Mohammed, and J. Fiaidhi, "Finding approximate solutions of np-hard optimization and tsp problems using elephant search algorithm," *The Journal of Supercomputing*, vol. 72, pp. 3960–3992, 2016.

[46] R. Roberti and P. Toth, "Models and algorithms for the asymmetric traveling salesman problem: an experimental comparison," *EURO Journal on Transportation and Logistics*, vol. 1, no. 1-2, pp. 113–133, 2012.

[47] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations research*, vol. 12, no. 4, pp. 568–581, 1964.

[48] V. Tongur and E. Ülker, "The analysis of migrating birds optimization algorithm with neighborhood operator on traveling salesman problem," in *Intelligent and Evolutionary Systems: The 19th Asia Pacific Symposium, IES 2015, Bangkok, Thailand, November 2015, Proceedings*, pp. 227–237, Springer, 2016.

[49] U. Derigs, B. Li, and U. Vogel, "Local search-based metaheuristics for the split delivery vehicle routing problem," *Journal of the Operational Research Society*, vol. 61, no. 9, pp. 1356–1364, 2010.

[50] Z. Fu, R. Eglese, and L. Y. Li, "A new tabu search heuristic for the open vehicle routing problem," *Journal of the operational Research Society*, vol. 56, no. 3, pp. 267–274, 2005.

[51] C.-T. Cheng, W.-C. Wang, D.-M. Xu, and K. W. Chau, "Optimizing hydropower reservoir operation using hybrid genetic algorithm and chaos," *Water Resources Management*, vol. 22, pp. 895–909, 2008.

[52] E.-G. Talbi, *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.

[53] E. Duman, M. Uysal, and A. F. Alkaya, "Migrating birds optimization: a new metaheuristic approach and its performance on quadratic assignment problem," *Information Sciences*, vol. 217, pp. 65–77, 2012.

[54] T. Duman and E. Duman, "Solving a new application of asymmetric tsp by modified migrating birds optimization algorithm," *Evolutionary Intelligence*, vol. 17, pp. 1–17, 2023.

[55] A. Bassaleh and E. Duman, "Turkish cashier problem with time windows and its solution by matheuristic algorithms," *RAIRO-Operations Research*, 2024. Forthcoming article. Received: 07 October 2023 / Revised: 12 February 2024 / Accepted: 04 April 2024.

[56] "Compound interest calculator UK." `https://occaminvesting.co.uk/compound-interest-calculator-uk/`. Accessed: March 2024.

[57] E. Duman and I. Or, "The quadratic assignment problem in the context of the printed circuit board assembly process," *Computers & Operations Research*, vol. 34, no. 1, pp. 163–179, 2007.

[58] R. Tavakkoli-Moghaddam, N. Safaei, M. Kah, and M. Rabbani, "A new capacitated vehicle routing problem with split service for minimizing fleet cost by simulated annealing," *Journal of the Franklin Institute*, vol. 344, no. 5, pp. 406–425, 2007.

[59] L. Moser and M. Wyman, "Stirling numbers of the second kind," vol. 25, 1958.

[60] A. Bassaleh and E. Duman, "Turkish cashier problem with time windows and its solution by migrating bird optimization algorithm," in *2023 9th International Conference on Optimization and Applications (ICOA)*, pp. 1–5, 2023.

[61] I. Cohen, Y. Huang, J. Chen, J. Benesty, J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," *Noise reduction in speech processing*, vol. 2, pp. 1–4, 2009.

VITA

Ahmad Bassaleh graduated with a Bachelor of Science degree in Industrial Engineering from Ozyegin University in 2022, subsequently pursuing a Master of Science degree in the same field at the same institution. His research focus primarily revolved around combinatorial optimization problems within the realm of transportation under Operations Research. Notably, he contributed to the academic literature through collaborative work with his advisor, Professor Ekrem Duman. Throughout his graduate studies, Ahmad also served as a teaching assistant, conducting recitation classes for Applied Statistics, Operations Research II, and Mathematical Modeling and Exact Methods courses.