Optimization of Transport Routes and Load Planning Using Algorithmic Approaches

Pedro Gomes Bacelar Pires

Master's Dissertation

Supervisor at FEUP: Prof. Gonçalo Figueira

Master's in Mechanical Engineering

2025-06-28

Abstract

The efficient planning of transport routes and load distribution plays a critical role in reducing costs and improving service levels in logistics operations. This dissertation addresses a real-world Rich Vehicle Routing Problem (Rich VRP), considering constraints such as heterogeneous fleet, vehicle load balancing and time windows. The study was developed in collaboration with a logistics company operating in the steel distribution sector, where the goal was to optimize daily delivery routes and reduce the total transportation cost by 10%.

In order to handle this problem, a metaheuristic approach based on the Greedy Randomized Adaptive Search Procedure (GRASP) was implemented, which combines a constructive phase to build solutions and a local search phase to refine them. The algorithm was tested on real operational data and its performance was evaluated for different iteration limits. The results confirmed that increasing the number of iterations improves the quality of the solution, with minimal improvements from a certain point. For the case studied, an iteration limit of 100 offered a good balance between computational efficiency and cost reduction, achieving an estimated savings potential of more than 10% in transportation costs.

Although a direct comparison with the company's current transport planning system could not be performed, the results highlight the practical value of adopting algorithmic optimization methods in this context. Future work includes integrating real-time routing services, validating the algorithm against historical cost data, and developing a dashboard to support ongoing performance monitoring and decision making.

Keywords: Rich VRP, Transportation Cost Reduction, Delivery Routes Optimization, GRASP

Resumo

O planeamento eficiente das rotas de transporte e da distribuição de carga desempenha um papel crucial na redução de custos e na melhoria dos níveis de serviço em operações logísticas. Esta dissertação aborda um problema real do tipo Problema de Roteamento de Veículos Rico (PRVR), considerando restrições como frota heterogénea, equilíbrio de carga nos veículos e janelas de tempo. O estudo foi desenvolvido em colaboração com uma empresa de logística que opera no setor da distribuição de aço, tendo como objetivo otimizar as rotas de entrega diárias e reduzir o custo total de transporte em 10%.

Para lidar com este problema, foi implementada uma abordagem metaheurística baseada no método Greedy Randomized Adaptive Search Procedure (GRASP), que combina uma fase construtiva para gerar soluções e uma fase de pesquisa local para as refinar. O algoritmo foi testado com dados operacionais reais e o seu desempenho foi avaliado para diferentes limites de iterações. Os resultados confirmaram que o aumento do número de iterações melhora a qualidade da solução, embora com melhorias mínimas a partir de determinado ponto. No caso estudado, um limite de 100 iterações revelou-se um bom compromisso entre eficiência computacional e redução de custos, alcançando um potencial estimado de poupança superior a 10% nos custos de transporte.

Embora não tenha sido possível realizar uma comparação direta com o sistema atual de planeamento de transportes da empresa, os resultados evidenciam o valor prático da adoção de métodos algorítmicos de otimização neste contexto. Trabalhos futuros incluem a integração de serviços de roteamento em tempo real, a validação do algoritmo com base em dados históricos de custos, e o desenvolvimento de um painel de controlo para apoiar a monitorização contínua do desempenho e a tomada de decisões.

Palavras-Chave: PRVR, Redução de Custos de Transporte, Otimização de Rotas de Entrega, GRASP

Acknowledgements

To Professor Gonçalo Figueira for his unmatched continuous support, which provided useful insights for the development and writing of this dissertation.

To my supervisor at Kaizen Institute, Engineer Alexandre Araújo, for all the lessons that will certainly guide me through my future. Also, to Engineer Pedro Magalhães, and the whole Kaizen team that participated in the development of the project, for being always available to provide help.

Last but not least, to my family and friends who have supported me unconditionally throughout my academic period and actively contributed to my personal and professional development, making these the best years of my life.

Thank you.

"Improvement usually means doing something that we have never done before."

Masaaki Imai

Contents

Acronyms

ACO	Ant Colony Optimization
ACS	Ant Colony System
BB	Branch-and-Bound
C&P	Cutting and Packing
CVRP	Capacitated Vehicle Routing Problem
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedures
HVRP	Heterogeneous Vehicle Routing Problem
ILS	Iterated Local Search
JIT	Just-In-Time
LIFO	Last-In-First-Out
LS	Local Search
MOGA	Multi-Objective Genetic Algorithm
NP-hard	Non-deterministic Polynomial-Time hard
RCL	Restricted Candidate List
SA	Simulated Annealing
SCM	Supply Chain Management
SDVRP	Site-Dependent Vehicle Routing Problem
TS	Tabu Search
TSP	Traveling Salesman Problem
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
VRPTW	Vehicle Routing Problem with Time Windows

List of Figures

List of Tables

Introduction

The distribution of goods is a critical component of modern supply chains, particularly in sectors characterized by being "extremely time-sensitive" (DispatchTrack (2025)). Operational adaptability can provide a decisive competitive advantage. One such sector is the distribution of construction and industrial materials, where clients usually operate under strict project schedules, and even small delays in material delivery can result in significant project downtime and financial penalties. Companies operating in this environment must manage a set of complex transportation constraints, including limited vehicle availability, varying load capacities, tight delivery time windows, and customer-specific unloading requirements. In addition, the nature of these materials often imposes load balancing and safety requirements during vehicle loading, which can complicate routing and planning activities. In this context, the development of flexible and computationally efficient route optimization techniques becomes essential to improve both service level and operational cost-effectiveness (Deloitte (2024)) .

1.1 Context and Motivation

This thesis approaches a case study in the construction sector. For confidentiality reasons, the company involved in this study will be referred to as Company X. Company X's services are specialized in the purchase, resale and distribution of metal products, with a focus on the construction sector and industry. They do not conduct any manufacturing activities in their facilities, which are dedicated only for the storage of recently acquired products, acting as a critical intermediary in their client's supply chain. At the current moment, its distribution activity takes place only in Portugal, with a network of clients extending mainly across the South, with a small amount of orders coming from the Center and the North. In addition, orders submitted to Company X can be classified as recurring orders, which happen repeatedly over time following a defined cycle, and non-recurring orders, which are unpredictable and require a dynamic management system for delivery and load planning.

At the moment, Company X's considerable lead time, transportation costs and manual setup of delivery routes represent a significant challenge for them. In addition, its logistics planning

team does not have any tool to rely on, which can be helpful to deliver orders in the most optimized routes. Therefore, suboptimal solutions often appear, resulting in a low vehicle occupation and a low number of customers visited per route. This is a clear opportunity for improvement: optimizing efficiency in fleet usage considering delivery, loading sequence, and other important restrictions by developing an interactive tool, which motivated the collaboration between Kaizen Institute and Company X. The project in which this initiative takes place includes other areas, with this focusing on the logistics department, which should deliver and implement a vehicle routing tool, particularly focused on non-recurring orders, specifically in steel rods, which represent 80% of their volume.

1.2 Objectives and Methodology

The transportation and delivery of steel materials to construction sites or industries introduces various challenges, such as the weight of these products, which presents strict limitations on vehicle capacity, requiring compliance with legal weight restrictions, and balance, since the symmetry between both sides of the vehicles has to be ensured. In addition, loading distribution planning becomes a significant difficulty because the available vehicles from Company X are loaded and unloaded from the right and left sides, which implies additional attention in the delivery sequence and consequently in loading sequence, since products loaded on one side cannot be unloaded from the opposite one. Furthermore, the loading of these materials into the vehicle should follow the exact reverse order of unloading, which represents the sequence of clients to deliver, employing the abstract concept of Last-In-First-Out (LIFO).

For this reason, and since this problem carries many conditions that must be met, it stands as a Vehicle Routing Problem (VRP), which frequently consists in minimizing total cost, distance, travel time, or even the number of vehicles used. In addition, the problem presents many restrictions that must be integrated into the developed model, which correspond to some of the existing VRP variants. Since dealing with a mixed fleet of vehicles, where all of them present different properties, in terms of capacity and length limits and cost per kilometer, it is important to consider the Heterogeneous Vehicle Routing Problem (HVRP), to select the most appropriate vehicle properties for each route. Secondly, the time window in which each customer can be visited may vary - Vehicle Routing Problem with Time Windows (VRPTW). Furthermore, certain customers cannot be visited by specific vehicles because their delivery sites do not allow it due to their length restrictions, representing the Site-Dependent Vehicle Routing Problem (SDVRP). Lastly, the Capacitated Vehicle Routing Problem (CVRP) surges as extremely relevant in Company X's context since they are dealing with really heavy loads, mainly composed from iron and steel materials, resulting as the primary focus of the model formulation.

Concerning this, the main objective of this project is to build a mathematical model combining different VRP variants related to the restrictions derived from the problem, and bridge this model into an algorithm which can reduce transportation costs, delivery lead time and enhance

vehicle occupation. All of this should be supported by a user-friendly interface, accessible to the logistics and planning team in Company X.

This project follows a methodology grounded on significant approaches focused on optimizing Company X's logistics procedures. Each method represents a distinct phase of the project, designed to follow a structured sequence of developments made throughout the process, providing a clear view and understanding of it.

Firstly, a systematic **literature review** was carried out in order to group and understand all the theoretical information relevant for the project. This procedure started by addressing the main questions related to the problem and clarifying them with the aim of defining the research to be done. After that, using the questions addressed above, a literature search was performed, from which several studies and scientific articles were found. Understanding whether their content would be relevant to the questions previously set was the next step. Finally, a deep and detailed analysis of the selected scientific articles was performed, enabling the assembly of significant data and information, with a focus on methodologies and their consequent results that could be helpful in reaching the main objectives of the project: reduction of delivery lead time and transportation costs.

Following this, **the development of a mathematical model** related to logistics optimization was carried out. First, the model was formulated on the basis of the VRP model, presenting an objective function of minimizing delivery costs. The second step of developing this mathematical model consisted of gathering input data and relevant details from Company X, with the intention of gradually evolving the model, assuring that it represents clearly Company X's delivery logistics reality.

The final step of this work consisted of **the development of an interactive tool** to optimize route and load planning, which could be resolved from three key processes. The first was done through the transformation of the previously described model into a *Python* algorithm, enabling dynamic management of the Client's Company logistics operations. Secondly, the integration of this algorithm into the selected interface (*Excel*), to be efficiently utilized by Company X's delivery planning department. Finally, continuous improvement was obtained by regularly adjusting the provided tool, adapting to possible changes in Company X logistics operations or external factors, guaranteeing an effective model over time.

In Chapter 2, a literature review is followed to give the reader a clear perspective of the theoretical background behind the project, exploring the relevant variants of the VRP. Chapter 3 presents a deep analysis concerning the problem of delivery logistics and load planning in Company X, highlighting all the improvement opportunities in the current process. An exact explanation of the methodology that was carried out is given in Chapter 4, including a detailed description of the theoretical model, the representative objective function, and all related constraints. Chapter 5 exhibits the results obtained from applying the model to Company X's logistics operations procedures. Finally, a conclusion of the dissertation is given in Chapter 7, resuming the major findings

and contributions of the research, providing future work possibilities, and exploring implications that can surge regarding the developed tool.

Literature Review

Companies that include distribution in their operations face unique logistical challenges due to the need to efficiently manage transport operations while adapting to customer demands. In this context, optimizing the delivery process is essential, not only to reduce operational costs but also to improve service levels and customer satisfaction. The VRP, one of the most studied and practically relevant problems in this domain, aims to determine the most efficient routes for a fleet of vehicles to deliver goods to a set of customers under various constraints.

Given the complexity and real-world applicability of the VRP, especially in distribution settings, new approaches such as numerous exact and heuristic methods have been evolving over the years, resulting in a vast literature around this theme. Among these, metaheuristics have emerged as powerful tools to obtain high-quality solutions for complex routing problems that are computationally unsolvable using exact methods. This chapter presents a review of the literature that covers key aspects of supply chain and logistics management, logistics operations in distribution companies, and VRP modeling and resolution through metaheuristic approaches.

2.1 Supply Chain Management, Logistics and Load Planning

Logistics was once defined by Ghiani et al. (2003) as the field that deals with the planning and control of material flows and related information in organizations. It was stated that its mission is to "get the right materials to the right place at the right time", while maximizing a defined performance measure and complying with a given set of constraints. Since dealing with many variants, logistics planning usually results in a key issue, which lies in determining how and when raw materials, semi-finished products, and finished goods should be acquired, transported, and stored along the supply network to optimize value for the entire value chain. The complexity of these choices increases with the number of suppliers, production steps, and demand requirements. However, it is relevant to remember that logistics challenges are not tied to manufacturing contexts but also to firms and public sector organizations, which usually face issues related to allocation of resources, for example. Porter (1985) separated inbound and outbound logistics, which were classified as core primary activities that directly contribute to the creation of value to customers.

Inbound logistics covers the receipt, handling, and storage of incoming materials, while outbound logistics, which will be the focus of this work, involves the processes related to the storage, transportation, and delivery of finished products to customers. With dynamic market demands and high service-level expectations continuously surging, the ability to effectively plan and execute logistics activities becomes a key differentiator. By particularly focusing on outbound flows in storage and distribution networks, this work aims to contribute to the optimization of delivery operations while responding to customer requirements in a timely and cost-effective way.

2.1.1 Load Planning

Load planning can be defined as the process of maximizing the available space of vehicles, consisting of the strategic arrangement and distribution of goods considering constraining factors such as weight, size, delivery demands, and materials specifications (Bortfeldt and Wäscher (2013)). The ultimate goal of this process is to minimize the number of vehicles used while maximizing their occupation, which usually results in complex problems in real organizations, since they deal with a lot of different requests.

Traditionally, the people responsible for load planning in logistics sectors deal with Cutting and Packing (C&P) problems, which were classified by Wäscher et al. (2007). Relying on manual processes to arrange routes and loads is common, which is time-consuming and prone to human error, as illustrated by Company X, which presents the motivation for this work. This approach does not usually consider all the critical factors such as driver workloads, rest times, and optimal routing, since humans do not have the ability to do it, leading to inefficiencies like overworked drivers, delayed deliveries, and non-compliance with regulations. In contrast, software-assisted load planning uses advanced algorithms to optimize load distribution and routing. These systems can automatically consider various constraints, such as vehicle capacities, delivery windows, drivers' work hours, and route sequence to generate efficient loading plans.

The shift from traditional to software-assisted load planning, which represents the objective of this work, addresses the limitations of manual methods, offering increased accuracy and operational efficiency. This transition is crucial for modern logistics operations that need to meet increasing demands and complex delivery requirements.

2.2 Vehicle Routing Problem

The VRP is a fundamental combinatorial optimization problem inserted in operational research that plays a significant role in the field of logistics, transportation, and SCM. Originally introduced by Dantzig and Ramser (1959), it consists of determining the optimal set of routes for a fleet of vehicles to deliver goods to a given set of customers, starting and ending at a central depot. This definition constitutes the baseline of the VRP, the objective of which is usually to minimize total route cost, distance, or time, while satisfying various operational constraints, such as vehicle capacity, delivery time windows, among others.

Since its appearance, it has been extremely developed and expanded. With research focused on exact algorithms and heuristics capable of solving relatively small instances, such as Branch-and-Bound (BB) and savings algorithms developed by Clarke and Wright (1964), this led to an improved ability to deal with more complex problems. Later in the last decade of the 20^{th} century, the introduction of metaheuristics including Genetic Algorithms (GA) introduced by Holland (1975) and Tabu Search (TS) studied by Glover (1986) brought a turning point to operational research methods, offering powerful tools to address rich VRP problems with a large number of constraints.

Following this, an overview of all the VRP variants that will be part of the solution is provided, which means that each of them represents a separate requirement that must be satisfied from the original problem.

2.2.1 Capacity

The first variant is the CVRP, which is considered one of the most fundamental approaches of the classical VRP. It was first introduced by Dantzig and Ramser (1959) and states that a fleet of vehicles with identical capacity is located in a central depot and must service a set of customers with known demands, without exceeding the capacity limit of any vehicle, with the common objective of minimizing the total distance traveled or the total cost of transportation. This formulation highlights one of the essential and most relevant constraints in real-world logistics scenarios, known as vehicle load capacity, which is critical when transporting products such as construction materials, retail goods, among others.

Since its appearance in the literature, CVRP has been extensively studied and has served as the basis for many more complex VRP extensions, constituting a more generalized approach. A wide range of heuristic or metaheuristic approaches have been proposed to tackle the CVRP, such as the one from Laporte (1992), especially for larger instances where exact methods become computationally impractical. In most real cases and in the context of the present project, capacity constraints are often the primary operational limitation. This is particularly true in industries that deal with high-density products, eliminating restrictions that could possibly surge with respect to vehicle volume limits, since the limit of loads is always related to their weight.

2.2.2 Heterogeneous Fleet

This variant, known as the HVRP, considers a fleet composed of vehicles with different capacities, costs, dimensions, and other operational constraints, providing a more realistic framework for logistic operations modeling. Its formal introduction was made by Golden et al. (1984), in which the study went beyond structuring routing decisions by addressing the optimal selection of the fleet mix. The objective of this approach is to minimize the total operational cost, which includes both the fixed cost of using specific vehicles and the variable costs related to the selected routes.

Concerning HVRP, additional research has been done over the years, highlighting the contribution of Taillard (1999) by developing and proposing a column generation method. The results showed that it is highly relevant in optimizing either the fleet composition or the total distribution costs, specifically for medium and large-size problem instances.

Furthermore, an extensive review of HVRP over three decades is presented by Çağrı Koç et al. (2016), in which a categorization of the existing literature around the variant is made, emphasizing the evolution of variants based on HVRP and their solution methodologies. In addition, the article provides a comparative analysis of metaheuristic algorithms developed for these problems and an overview of their performance and applicability along the variant framework, showcasing the developments over the past decades.

In general, the HVRP has established itself as a fundamental extension of the classical VRP, by allowing for the consideration of mixed fleets. It introduces a more realistic scenario where route planning and fleet assignment are combined, making it a more versatile approach for advanced routing problems.

2.2.3 Site-Dependency

The SDVRP rises as a classical variant of the VRP, which was first introduced by Chao et al. (1998) and explains that each customer can only be served by a subset of vehicles available from the heterogeneous fleet due to some real-world scenarios such as road accessibility and compatibility between the vehicle and the delivery site of the customer. This main condition makes the differentiation between the SDVRP and the VRP since in this last one, it is assumed that all vehicles can visit all customers without distinction and specific length requirements, for example.

This variant consists of some additional complexity in the formulation of the VRP model, and some studies have already shown that these supplementary conditions generally have a huge impact and complicate the routing and vehicle assignment process. Following the foundational work cited above, the SDVRP has attracted considerable research interest, where various exact methods have been developed for smaller problem instances (Baldacci et al. (2012)). Furthermore, and since real logistics problems usually carry some additional operational difficulties, metaheuristics and hybrid heuristic approaches have become the favorite method to solve larger instances (Chao and Liou, 2005), due to the increased computational difficulty of SDVRP.

The SDVRP is included in a broader class of problems where multiple conditions of real-world complexities need to be integrated into the model structure. Research on this problem continues to be highly relevant for logistics business frameworks, which is the case of Company X, since dealing with heavy and long materials and delivering them in construction sites requires dynamic adaptation and high computational performance.

2.2.4 Time Windows

One of the most studied extensions of the main problem is VRPTW, which was first introduced by Solomon (1987). It states that each customer has to be served within a specific time interval and acts in not only the route optimization to minimize costs, but also must guarantee that the service at each customer must start within the predefined time window. This approach also deals with events outside the defined time windows, as it allows early arrivals but considering the following waiting time, but strictly forbids late arrivals.

The VRPTW can be considered as a much more complex problem than the traditional VRP due to the addition of scheduling conditions, which often requires advanced solution methods. Regarding this, Desrosiers et al. (1995) applied branch-and-price algorithms, exploring the structure of time windows through a column generation method, adopting exact algorithms. For larger instances, Ombuki et al. (2006) developed a Multi-Objective Genetic Algorithm (MOGA) in which they focused on minimizing the total travel distance and the number of vehicles used, able to match real-world requirements, but at the cost of a computationally challenging problem.

In common scenarios, VRPTW models cover critical operational conditions that add a layer of complexity to the model formulation, related to scheduling and overall service time. Failing to comply with customers' time requirements usually results in general dissatisfaction, but also in relevant financial penalties. From that, VRPTW has become a serious topic in the field of combinatorial optimization and logistics innovation, particularly with the increasing demand for one-day deliveries and Just-In-Time (JIT) logistics, studied by Bräysy and Gendreau (2005a).

2.3 Rich Vehicle Routing Problems

With the complexity of real-world logistics scenarios continuously increasing, people in the field began to recognize that most problems could not be adequately represented by a single VRP variant, which is the case represented in this work. This event led to the rise of the Rich Vehicle Routing Problems (Rich VRPs) concept, which includes problems that deal with the need to incorporate multiple previously studied VRP variants, such as time windows, heterogeneous fleets, multiple depots, split deliveries, site dependence, and many others. These problems are defined by a single mathematical formulation, but have combinatorial nature, combining parts from the adequate VRP variants, reflecting the need to model industry requirements carefully, in order to meet with customers strict delivery schedules and complex requirements, not forgetting the availability of limited resources as well.

Researchers have responded to this by constructing flexible algorithmic frameworks, specifically adopting metaheuristics and hybrid approaches as main solutions, as shown by Vidal et al. (2013a), since they appear as methods capable of handling multiple intertwined constraints in an efficient way. According to the work of Cordeau et al. (2007), the rise of Rich VRPs states a shift in vehicle routing research from academic developments to solving problems closer to the

challenges that companies and industries, including logistics in their operations, face in reality.

Given the complexity of the challenge faced in this work, involving time windows, heterogeneous fleet, site dependence, and load balance constraints, exact methods become computationally inadequate for realistically sized instances. Therefore, metaheuristic approaches have emerged as a promising method since they present a good balance between solution quality and computational efficiency. In the following section, a review of these methods is presented with a focus on their application to VRP variants relevant to this study.

2.4 Metaheuristic Approaches

Solving problems that involve combinatorial optimization in an efficient way is one of the main challenges in operational research. The VRP and its many variants, in particular Rich VRPs, are included in the category of Non-deterministic Polynomial-Time hard problems, known as NP-hard (Hartmanis (1982)). For these problems, no known algorithm can find an optimal solution in polynomial time as the input size grows, meaning that the computational effort and time required to solve instances grows exponentially with the problem size and complexity. This limits the use of exact methods, even though capable of finding optimal solutions.

In this context, metaheuristics have become a powerful solution for NP-hard problems, such as the VRP. Metaheuristics are designed to guide heuristics in efficiently exploring large solution spaces. They offer a balance between intensification, which means developing the best solutions found until the moment, and diversification, which consists of exploring unvisited regions of the search space, helping to improve solution quality. Furthermore, while heuristic methods like the savings algorithm introduced by Clarke and Wright (1964) or nearest neighbor constructions offer fast and problem-specific solutions, they often lack the capacity required to solve large and rich routing problems. On the other hand, metaheuristics have been showing high- level performance across problems that involve a wide range of VRP variants, like the one for this work. Popular methods include GAs, TS, Simulated Annealing (SA), Ant Colony Optimization (ACO) and Variable Neighborhood Search (VNS), with many of them being successfully applied to Rich VRPs with multiple constraints, such as in the work of Vidal et al. (2013a).

The problem addressed in this dissertation includes multiple real-world constraints, which characterize it as a Rich VRP. Given its scale and structure, exact methods can be inadequate, and therefore metaheuristic approaches provide a practical and effective alternative, offering flexibility and good solution quality within realistic computational time. Following this, the next sections present an overview of the most relevant metaheuristic techniques applied to VRPs, with a particular focus on their possible adaptation and incorporation to complex and realistic problems such as the one studied in this work.

2.4.1 Genetic Algortihms

Known as one of the most well-known metaheuristics approaches, GAs belong to the group of population-based metaheuristics, which are inspired by the principles of natural selection and evolution, were first introduced by Holland (1975). Regarding the optimization topic, GAs consist of iteratively evolving a set of candidate solutions (known as individuals or chromosomes) in order to obtain better quality by introducing genetic operators such as selection, crossover, and mutation. Each solution is then evaluated by an objective function that guides the evolutionary process toward promising regions of the solution space.

GAs are particularly useful for solving large and complex combinatorial optimization problems such as Rich VRPs with its many variants. Their strength lies in their ability to create a vast and diverse population of solutions, allowing a global search that helps escape local optima, which is a common issue in routing problems. Moreover, GAs offer flexibility to include constraints and integrate problem-specific knowledge through customized operators, repair mechanisms, or the adoption of hybrid methods by combining themselves with local search heuristics (Prins (2004)).

In VRP's framework, GAs have been widely studied and in constant adaptation. For example, Berger and Barkaoui (2003) developed a hybrid GA for the VRPTW, strategically combining evolutionary methods with local search to improve promising solutions. Ombuki et al. (2006) applied a MOGA, also to the VRPTW, with which he was able not only to minimize the total distance but also the number of vehicles needed, resulting in a relevant work for real-world scenarios where the trade-off between cost and resources is important. Vidal et al. (2013b) proposed a hybrid GA that was capable of handling multiple variants of the VRP, including those with heterogeneous fleet, time windows, and split deliveries, which is similar to the problem addressed in this work, but different since it does not include the load planning aspect. Given these advantages and similarities, GAs can be considered one of the most effective metaheuristic approaches to solve Rich VRPs, offering efficiency and adaptability. Their modular structure allows them to be customized to complex routing scenarios such as the one addressed in this work, where multiple constraints interact in a complex way, which translates as a suitable approach for this work.

2.4.2 Ant Colony Optimization

Dorigo et al. (1996) developed a metaheuristic approach known by ACO that, as GA, belongs to the class of population-based metaheuristics, which solves combinatorial optimization problems such as the Traveling Salesman Problem (TSP) (originally surging in the 19^{th} century), by getting inspiration from the common activity of actual ants. Artificial ants are guided by pheromone trails and problem-specific information to navigate the solution space, the former is updated over time to fade less promising courses and improve the most desirable ones, ensuring a process of collective learning that leads to high-quality solutions.

This constructive and adaptable behavior has made it especially successful for routing issues such as the VRP. The algorithm is ideal for situations with dynamic restrictions, for example, the

ones presented in this work, because it creates routes in an incremental way. In addition, ACO can navigate complex search environments without converging to inferior solutions, which is a major difficulty in VRP variations, due to its ability to strike a balance between exploration and exploitation. ACO has suffered modification in a number of ways for VRP applications in the literature. In order to increase the speed of convergence and the quality of the solution, Gambardella et al. (1999) introduced the Ant Colony System (ACS) for the VRPTW, which integrates specialized pheromone updates with local search strategies. ACO was more recently applied to Rich VRPs with numerous constraints by Yu et al. (2009), who showed that the technique can be successfully modified to include characteristics such as heterogeneous fleets and customer time preferences.

Through parameter adjustment and the constant attempt to mix with other methods, ACO's structure enables adaptation that can improve intensification and diversification. The flexibility and performance of this approach in complex routing scenarios translate it into a suitable metaheuristic in the context of this dissertation, where the problem involves various coexisting restrictions.

2.4.3 Tabu Search

Known as a capable local search-based metaheuristic, TS was first developed by Glover (1986), to overcome the disadvantages of conventional local optimization methods. By allowing non-improving moves under specific circumstances and utilizing a memory structure, to prevent cycling and promote exploration of new regions in the solution space, it improves on basic hill-climbing algorithms that might become stuck in local optima.

In an iterative way, the algorithm moves from one solution to a nearby one, which is usually produced by minor adjustments such as customer reassignments, reverse route order, or node swaps. TS keeps a short-term memory of recent moves or characteristics of solutions in order to prevent repeating recently examined solutions, classified as "tabu", which may also be overridden if it results in an especially good solution, offering an equilibrium between exploring new locations and exploiting promising areas. Because of that, TS has demonstrated exceptional performance in a wide range of VRP variants, for example, when Éric Taillard et al. (1997) developed a TS heuristic for VRPs with soft time windows that included customer insertion or removal and route improvement techniques, making it one of the first successful applications of TS to VRPs. Gendreau et al. (1994) used specific neighborhood structures to create a variation for the VRPTW. For instance, in their comprehensive review of metaheuristics for the VRPTW, Bräysy and Gendreau (2005b) highlighted the versatility of TS, pointing out how easily it may be used with other techniques such as constraint programming or Local Search (LS).

Due to its resilience and versatility, TS can be especially suited to issues that include real-world limitations, because of its modular structure, since it is simple to adapt memory methods, punishment mechanisms, and neighborhood operators to the unique features of dynamic and Rich VRP instances.

2.4.4 Variable Neighborhood Search

Mladenović and Hansen (1997) developed the metaheuristic framework known as VNS, which is based on the methodical change of neighborhood structures throughout the search process. VNS specifically overcomes this constraint by repeatedly alternating between the neighborhoods, in contrast to conventional LS techniques that investigate a single neighborhood and usually become stuck in local optima. Due to this diversification, the algorithm can break out of local optima and investigate larger areas of the solution space. In summary, the core idea of VNS is simple, but powerful and scalable, since it starts with an initial solution, applies a shaking procedure to generate a new solution from a different neighborhood, and then applies LS to find a local optimum in the new region. After that, through an evaluation of the new solution, it can become the current one if it is better than the previous one. If not, the algorithm continues exploring in another neighborhood. The process repeats until a stopping condition is met, such as a maximum number of iterations or a time limit.

This heuristic has been successfully applied to various VRP variants due to its effectiveness. For example, when Polacek et al. (2004) applied it to the Capacitated Arc Routing Problem with intermediate facilities, showing strong performance on complex logistics problems. Furthermore, VNS has been adapted to handle VRPs with constraints similar to this work's, such as time windows, heterogeneous fleets, site dependency, and loading constraints, often by designing custom neighborhood structures that respect these features. Hemmelmayr et al. (2012) work resulted in a hybrid VNS for Rich VRPs involving time windows and pickups and deliveries, demonstrating the method's capacity to manage tightly constrained routing scenarios.

VNS brings advantages in the way that it can be combined with constructive heuristics or other metaheuristics like GAs to enhance performance, making it particularly well-suited for Rich VRPs. In the context of this dissertation, VNS emerges as a relevant option, since it offers the flexibility to define custom neighborhoods and guide the search effectively through a complex and constrained solution space.

2.4.5 Greedy Randomized Adaptive Search Procedure

The Greedy Randomized Adaptive Search Procedure (GRASP) is a flexible constructive metaheuristic introduced by Feo and Resende (1995) to solve complex operational research problems. Unlike population-based metaheuristics such as GA and ACO, this is a multi-start, constructive, and iterative metaheuristic, in which each iteration consists of two phases: a greedy randomized construction of a feasible solution, followed by a LS to improve it. The best solution found over all iterations is retained as the final output.

In the first phase, established as the constructive one, a solution is built using a greedy function that evaluates candidate components, for example which customer to visit next, in the context of VRP. Rather than always choosing the best available option as in pure greedy algorithms, GRASP introduces randomization by selecting from a Restricted Candidate List (RCL), which

contains the best ranked elements according to a predefined parameter. This randomness allows exploration of different regions of the solution space and helps avoid the behavior of deterministic heuristics. The second phase (LS phase) is included with the goal of improving the solution that results from the previous one, by exploring its neighborhood, such as through customer swaps or route rearrangements. This hybrid structure gives GRASP its potential as a metaheuristic, that resumes on the greedy construction that ensures feasibility, while the LS enhances quality.

Due to its simplicity and flexibility, GRASP has been widely applied to VRPs and their many extensions. For instance, Prins (2009) applied GRASP to the HVRP in its work, developing a route-first, cluster-second approach that handles different vehicle types. In addition, Subramanian et al. (2013) designed a hybrid GRASP with Iterated Local Search (ILS) and path relinking to tackle the VRPTW, showing competitive results when combining the analyzed metaheuristic with local improvement heuristics. In the context of Rich VRPs as the one presented in this work, GRASP has also been extended to handle site dependency, split deliveries, and other conditions, often through custom construction rules and neighborhood structures (Suárez and Anticona (2010)).

In the present work, GRASP is selected as the core metaheuristic due to its scalability and simplicity of parameters, ease of implementation in real complex routing problems, and ability to effectively manage the combination of constraints involved in this work, such as heterogeneous vehicle fleet, site dependency, time windows, load planning, and potential split deliveries. Its modular nature allows for the inclusion of problem-specific logic in both the construction phase and the LS. Additionally, its multi-start architecture makes it well suited to explore a large and diverse solution space within reasonable computational time.

2.5 Literature Review Summary

The review of VRP variants and metaheuristic approaches for routing problems highlights the increasing attention given to realistic and operationally constrained variants, commonly grouped under the term Rich VRPs. However, it is also evident from the literature that very few works simultaneously address the combination of constraints found in real industrial settings, highlighting a gap in the current body of work. With that, the present dissertation contributes to the literature by focusing on a Rich VRP problem that integrates multiple constraints simultaneously. The selected methodology, GRASP, stands out as a promising approach due to its flexibility, modular design, and successful applications to similarly constrained problems. Table 2.1 presents a summary of the literature research performed for this work.

Table 2.1: VRP and Metaheuristics Summary

Article	Approach	VRP	HVRP	SDVRP	VRPTW	Rich VRP
Golden et al. (1984)		x	x			
Taillard (1999)	TS	x	x			
Çağrı Koç et al. (2016)		x	x			
Chao et al. (1998)		x		x		
Baldacci et al. (2012)		x		x		
Chao and Liou (2005)	TS	x		x		
Solomon (1987)		x			x	
Desrosiers et al. (1995)		x			x	
Ombuki et al. (2006)	MOGA	x			x	
Bräysy and Gendreau (2005a)		x			x	
Vidal et al. (2013a)		x				x
Cordeau et al. (2007)		x				x
Holland (1975)	GA					
Prins (2004)	GA + LS	x				
Berger and Barkaoui (2003)	GA + LS	x			x	
Vidal et al. (2013b)	GA	x	x		x	
Dorigo et al. (1996)	ACO					
Gambardella et al. (1999)	ACS	x			x	
Yu et al. (2009)	ACO	x				x
Glover (1986)	TS					
Éric Taillard et al. (1997)	TS	x			x	
Gendreau et al. (1994)	TS	x			x	
Bräysy and Gendreau (2005b)	TS + LS				x	
Mladenović and Hansen (1997)	VNS					
Polacek et al. (2004)	VNS				x	
Hemmelmayr et al. (2012)	VNS	x				x
Feo and Resende (1995)	GRASP					
Prins (2009)	GRASP	x	x			
Subramanian et al. (2013)	GRASP	x			x	

Problem Description

In this chapter, the focus is on clearly defining the problem under study and formulating it in an operational research framework, in the context of the VRP. The objective is to provide a detailed understanding of the context in which the problem arises, the operational constraints involved, and the goals to be achieved. After that, a precise mathematical construction of the problem is followed, identifying the key decision variables, constraints, and objective function, building the foundation for the development of an appropriate optimization method.

3.1 The Problem of Company X

The main activity of Company X consists of the storage and distribution of different types of steel and construction products. Everyday, they need to deliver previously requested orders to their clients, which is the function of the logistics department. In addition, they receive orders and decide if they can be accepted or not, based on stock levels and, which is the role of the customer service team. Finally, they need to order materials from their suppliers, taking into account the actual stock levels and orders that will be delivered in the next days, which is usually done by the company administration.

Company X faces many challenges, the most critical being the logistics department, especially in route and load planning optimization. At the moment, the decision of which orders are delivered each day is made purely based on human instinct and knowledge from past years and experiences with previous clients, which is subject to human error and never leads to the most optimized solution. Furthermore, all of this planning is currently done by hand, with no technological support, resulting in a high volume of papers and unnecessary utilization of space that could be used for other purposes.

Knowing the current situation of the logistics team framework and the demanding nature of the construction sector, which represents a large part of their clients, it is relevant to implement cost-effective and reliable planning tools. Addressing this problem is essential for many reasons, such as the reduction of transportation costs, the improvement on service level and the reduction of time spent on routing activities.

Company X's operational characteristics must be explored, since they represent the way in which the problem will be formulated. First, it is important to highlight the fact that this dynamic tool is developed to manage 12 meters steel rod orders, since they represent a huge part of Company X's incoming orders. In terms of transportation modes, the company has a heterogeneous fleet of vehicles available for deliveries, with a maximum length ranging from 6 to 14 meters, a maximum load capacity going from 11000 to 35000 kilograms, a fuel consumption between 25 and 35 liters per 100 kilometers, among other less relevant properties.

Building on its operational requirements, Company X's services must be performed under some constraints, which are relevant to include in the formulation of the problem and what distinguishes this work from other routing problems. Concerning this, and after understanding better the difficulties that attract the company logistics team attention, the main constraints are clear: the sequence of order loading in the vehicles should follow a LIFO policy, which means that the order in which customers are visited must be the reverse of the vehicle loading order. Additionally, there is a maximum weight difference between the left and right sides of the vehicles, equal to 5000 kilograms, which must be respected. This is related to the fact that about 20% of Company X's customers orders can only be delivered on either side of the vehicle, causing an impact on route planning and load distribution. Furthermore, the company must deal with its customers' delivery time windows, which represent the interval of time when that customer is available to receive its order. In addition, customers can have length restrictions, which means that not every vehicle can visit every customer, for example a vehicle whose length is 12 meters is not allowed to visit a customer with a maximum accepted length of 10 meters. Finally, drivers' working hours per day should be respected, which means that each route should take the same or less time than the driver working time interval.

Some assumptions were made to simplify the mathematical formulation of the problem, in accordance with the studied literature. Company X logistics operations deal with a deterministic demand, since it is known in advance and does not vary. A heterogeneous fleet of vehicles with fixed availability is assumed, with all of them starting and ending their routes at the depot, which is assumed to be a single one. Each customer's order must be delivered in full by a single vehicle, which represents the assumption of no split deliveries. Replenishment is forbidden, stating that vehicles are not allowed to return to the depot to reload once a delivery route has started. The loading and unloading times are incorporated into the time window constraints, consisting of a fixed time if the vehicle has to stop and a variable time depending on the quantity that is being delivered at that stop. Lastly, driver and vehicle compatibility is assumed, which means that all drivers are qualified to drive any available vehicle.

The main objective of this work is to develop an automated tool that can design efficient vehicle routes that satisfy all customer demands while complying with all the operational constraints mentioned above. Regarding that, the primary goal is to reduce the total transportation cost by 10%, also aiming to reduce delivery lead times by 30%. These objectives support the company's strategic need for rapid and reliable delivery in a highly time-sensitive market.

3.2 Mathematical Formulation

This section details the mathematical model formulation that addresses the problem of Company X, which consists of a Rich VRP, since it combines some traditional variants of the simple VRP. It will expose the model in terms of notation used for sets, parameters and decision variables, objective function, and constraints. Table 3.1 summarizes all this information.

Table 3.1: Notation used in the model formulation

Sets	**Description**
N	Set of nodes, indexed by i, j
C	Set of customers, indexed by i, $C = N \setminus \{0\}$
V	Set of available vehicles, indexed by v
L	Set of customers that require delivery on the left side, indexed by i, $L \subseteq C$
R	Set of customers that require delivery on the right side, indexed by i, $R \subseteq C$
IV	Set of valid (customer, vehicle) assignment pairs, indexed by (i, v)
Parameters	**Description**
q_i	Demand of customer $i \in C$
cm_v	Capacity of vehicle $v \in V$
ck_v	Cost per kilometer of vehicle $v \in V$
d_{ij}	Distance between node i and node j, $i, j \in N$
two_i	Opening of time window of customer $i \in C$
twc_i	Closing of time window of customer $i \in C$
tv_{ij}	Travel time between node i and node j, $i, j \in N$
tfd	Fixed loading and unloading time
tvd	Variable loading and unloading time
Δ	Maximum asymmetry allowed between left and right sides of the vehicle
M	Very large constant
Decision Variables	**Description**
x_{ijv}	Binary variable: 1 if vehicle v travels from node i to node j, 0 otherwise
y_{iv}	Binary variable: 1 if customer i can be served by vehicle v, 0 otherwise
zl_{iv}	Binary variable: 1 if customer i is served on the left side by vehicle v, 0 otherwise
zr_{iv}	Binary variable: 1 if customer i is served on the right side by vehicle v, 0 otherwise
tc_i	Continuous variable: time of arrival at customer i
f_{ijv}	Continuous variable: load flow going from node i to node j in vehicle v

The objective function and the constraints related to the model are presented below.

Objective Function

$$\min \quad \sum_{i \in N} \sum_{j \in N} \sum_{v \in V} d_{ij} \cdot ck_v \cdot x_{ijv} \tag{3.1}$$

Constraints

$$\text{s.t.} \quad \sum_{(i,v)\in IV} y_{iv} = 1 \qquad \forall i \in C \tag{3.2}$$

$$\sum_{(i,v)\in IV} q_i \cdot y_{iv} \leq cm_v \qquad \forall v \in V \tag{3.3}$$

$$\sum_{j\in N, j\neq i} x_{ijv} = \sum_{(i,v)\in IV} y_{iv} \qquad \forall i \in C,\ \forall v \in V \tag{3.4}$$

$$\sum_{j\in N, j\neq i} x_{jiv} = \sum_{(i,v)\in IV} y_{iv} \qquad \forall i \in C,\ \forall v \in V \tag{3.5}$$

$$\sum_{i\in C} x_{i0v} \leq 1 \qquad \forall v \in V \tag{3.6}$$

$$\sum_{j\in C} x_{0jv} \leq 1 \qquad \forall v \in V \tag{3.7}$$

$$two_i \leq tc_i \leq twc_i \qquad \forall i \in C \tag{3.8}$$

$$tc_j \geq tc_i + tfd + tvd \cdot q_i + tv_{ij} - M \cdot (1 - x_{ijv}) \qquad \forall i, j \in C,\ i \neq j,\ \forall v \in V \tag{3.9}$$

$$zl_{iv} + zr_{iv} = y_{iv} \qquad \forall (i,v) \in IV \tag{3.10}$$

$$zl_{iv} = y_{iv} \qquad \forall (i,v) \in IV,\ \forall i \in L \tag{3.11}$$

$$zr_{iv} = y_{iv} \qquad \forall (i,v) \in IV,\ \forall i \in R \tag{3.12}$$

$$\sum_{(i,v)\in IV} q_i \cdot zl_{iv} - \sum_{(i,v)\in IV} q_i \cdot zr_{iv} \leq \Delta \qquad \forall v \in V \tag{3.13}$$

$$\sum_{(i,v)\in IV} q_i \cdot zr_{iv} - \sum_{(i,v)\in IV} q_i \cdot zl_{iv} \leq \Delta \qquad \forall v \in V \tag{3.14}$$

$$f_{ijv} \leq cm_v \cdot x_{ijv} \qquad \forall i, j \in N,\ i \neq j,\ \forall v \in V \tag{3.15}$$

$$\sum_{j\in C} f_{0jv} = \sum_{(i,v)\in IV,\ i\neq 0} q_i \cdot y_{iv} \qquad \forall v \in V \tag{3.16}$$

$$\sum_{j\in N, j\neq i} f_{jiv} - \sum_{j\in N, j\neq i} f_{ijv} = q_i \cdot \sum_{(i,v)\in IV} y_{iv} \qquad \forall i \in C,\ \forall v \in V \tag{3.17}$$

$$x_{ijv} \in \{0,1\} \qquad \forall i, j \in N,\ \forall v \in V \tag{3.18}$$

$$y_{iv} \in \{0,1\} \qquad \forall i \in C,\ \forall v \in V \tag{3.19}$$

$$zl_{iv} \in \{0,1\} \qquad \forall i \in C,\ \forall v \in V \tag{3.20}$$

$$zr_{iv} \in \{0,1\} \qquad \forall i \in C,\ \forall v \in V \tag{3.21}$$

The objective function, presented in Equation (3.1), aims to minimize the total transportation cost associated with the delivery of steel rods to customers, from Company X's unique depot. It is calculated as the sum of distances traveled by each vehicle, weighted by the cost per kilometer of that vehicle, which reflects the heterogeneous nature of the fleet, since vehicles differ in capacity and cost efficiency.

The formulated model includes the following constraints.

Allocation and vehicle capacity: Constraint (3.2) ensures that each customer is served exactly once by one vehicle, excluding the possibility of assuming split deliveries in the problem. Constraint (3.3) limits the total weight of material assigned to any vehicle to its maximum capacity, highlighting the physical restrictions of each vehicle in Company X's heterogeneous fleet.

Flow conservation: Constraints (3.4) and (3.5) ensure that flow is conserved at customer nodes, which means that if a customer is assigned to a vehicle, that vehicle must arrive and depart from the customer node exactly once, allowing for every vehicle's route to continue. Constraints (3.6) and (3.7) guarantee that each vehicle leaves from and returns to the depot at most once, reflecting that each vehicle performs a single tour within the planning horizon, since multiple trips were not considered in the formulation of the problem.

Time windows and delivery times: Constraint (3.8) specifies that each delivery must respect the customer's time window, meaning that delivery must be after its opening and before its closing. Constraint (3.9) guarantees that the delivery time of the following customer takes into account travel time, fixed and variable unloading times, while using a very large constant formulation to cancel the constraint when the trip from i to j is not assigned to vehicle v.

Load balance and vehicle side allocation: Constraint (3.10) enforces that each customer order is assigned to one and only one side of the vehicle (left or right). Constraints (3.11) and (3.12) ensure that customers that require to be loaded on the left or right sides are correctly assigned. Constraints (3.13) and (3.14) control the weight asymmetry of the materials in the vehicle, restricting the difference between the two sides of each vehicle to a legal maximum value of 5000 kilograms, which represents a critical safety restriction in heavy products logistics.

Flow-based subtour elimination: Constraint (3.15) limits the flow variable on an arc by the vehicle's capacity, ensuring consistency between routing and load. Constraint (3.16) defines the initial amount of load carried by a vehicle, when leaving the depot, equal to the total demand of the customers assigned to it. Finally, constraint (3.17) ensures proper flow balance at each customer node, stating that the difference between incoming and outgoing flow matches the customer's demand whenever the vehicle serves that customer.

Development of a GRASP Metaheuristic for a Rich Vehicle Routing Problem

The approach used to tackle Company X's distribution problem, which was formulated as a Rich VRP, is presented in this chapter. The problem cannot be solved simply using exact methods because it involves many operational constraints, such as ensuring vehicle load balance, customer time windows, and heterogeneous fleet characteristics. A hybrid strategy was used to address this, combining the application of a metaheuristic approach based on GRASP with a mathematical modeling phase. This methodology was chosen because of its ability to balance solution exploration and refinement, adapt to unique constraints of a certain problem, and solve similar combinatorial optimization problems with accuracy.

4.1 Problem-Solving Approach

Given Company X's problem scale and its combinatorial nature, exact methods become computationally not capable, especially as the number of customers grows. With this, a metaheuristic approach was adopted to focus on computational efficiency at the cost of lower solution quality.

The methodology adopted in this dissertation integrates the formal mathematical modeling of the problem, followed by its testing with a CPLEX optimization solver tool and the development of a custom GRASP-based algorithm, coded in *Python*. The mathematical model serves as a foundation for the solution logic, providing a formal structure of decision variables, constraints, and the objective function. Even though not solved exactly due to scalability limitations, the model guides the implementation and is used to check feasibility in algorithmic steps. The GRASP heuristic, on the other hand, enables the generation of high-quality solutions within reasonable computational time, and can easily accommodate complex constraints such as time windows and vehicle load balance.

An overview of the general structure of the methodology adopted to solve Company X's problem is illustrated in Figure 4.1.

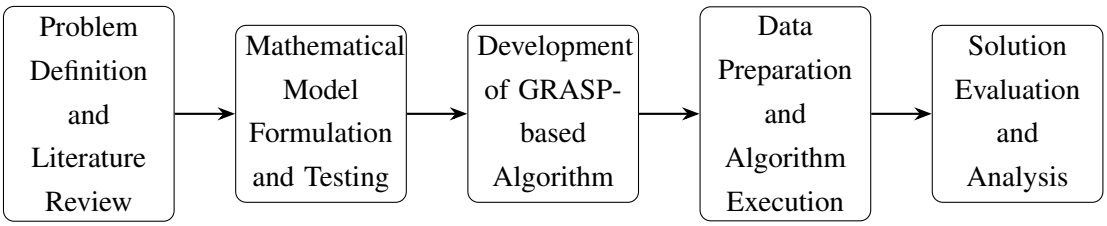

Figure 4.1: Overview of the adopted methodology

4.2 Model Definition and Validation

The mathematical model was created to achieve two complementary purposes. By defining the objective function, decision variables, and constraints involved, it first acts as a formal base for understanding the problem's structure and complexity. After that, it enables the validation and feasibility check of heuristic solutions by providing near-optimal solutions for smaller instances, using an exact solver. For this reason, the model was developed and tested using IBM ILOG CPLEX Optimizer, which is a commercial solver designed for mixed-integer programming problems in the area of operational research.

The testing of the model was restricted to small instances which were used to benchmark the feasibility of the model solutions. The development of the mathematical model in CPLEX included all pertinent operational constraints, including capacity limitations per vehicle, time window compliance, route continuity, and vehicle loading balance, closely following the formulation provided in the previous chapter.

Small-scale instances were defined and tested, which allowed direct benchmarking of the parameters used and analysis of the level of constraint satisfaction. These instances were intentionally kept small to ensure that they could be solved near optimality within a reasonable time. Regarding the instances considered for testing, which were kept similar to real data from Company X, a summary of them is provided in Table 4.1, while both distance and travel time matrices referring to these instances are available in Appendix A.

Table 4.1: Instance parameters considered for CPLEX validation

Parameter	**Instance 1**	**Instance 2**	**Instance 3**
Number of customers (n)	3	6	8
Number of vehicles (m)	1	2	3
Vehicle capacities (cm_v)	[4]	[8, 10]	[6, 8, 4]
Vehicle cost per km (ck_v)	[1.0]	[1.0, 1.2]	[1.0, 1.1, 0.85]
Depot index	0	0	0
Customer demands (q_i)	[0, 1.2, 2, 0.4]	[0, 0.2, 2, 0.6, 1.5, 2.5, 1]	[0, 0.6, 2, 0.8, 1.6, 2.5, 1.8, 0.8, 1.2]
Not allowed (customer, vehicle) pairs (IV)	-	(3,1), (6,2)	(1,3), (3,1), (3,2), (4,2), (4,3), (6,1), (6,3), (7,1), (7,3), (8,3)
Customers on left side (L)	-	{3}	{3, 4}
Customers on right side (R)	{1}	{2, 5}	{2, 5}
Distances matrix (d_{ij}) - Appendix A	Symmetric 4x4	Symmetric 7x7	Symmetric 9x9
Travel times matrix (tv_{ij}) - Appendix A	Symmetric 4x4	Symmetric 7x7	Symmetric 9x9
Time window open (two_i)	[0, 150, 80]	[0, 200, 0, 120, 0, 50]	[0, 200, 0, 120, 0, 50, 0, 0]
Time window close (twc_i)	[400, 480, 480]	[480, 480, 480, 480, 380, 480]	[480, 480, 480, 480, 380, 480, 250, 480]
Fixed unloading time (tfd)	5	5	5
Variable unloading time (tvd)	0.5	0.5	0.5
Max load asymmetry (Δ)	2	2	2

It can be observed from Table 4.1 that the instances that were considered for the tests evolved from the first to the last, in terms of the variables involved and the parameters to solve, bringing more complexity to the solver in a gradual way.

After the instance parameters were defined, each configuration was solved using the CPLEX optimizer with the objective of reaching an exact or nearly optimal solution, depending on the instance size. Furthermore, the goal of the mathematical model validation was to assess the performance of the heuristic in small instances, so it could evolve into a fully developed algorithm, capable of solving any type of problem size.

The solutions discovered for Instances 1, 2, and 3 are shown in Table 4.2. The precision of

the model and the solver's ability to handle the specified constraints were confirmed by the fact that every instance complied with the imposed asymmetry limit, vehicle capacities, route start and finish at the depot, among all the other constraints mentioned in Chapter 3. Moreover, it was not possible to show the increase in the computing times for larger instances through the CPLEX Optimizer due to the limit of maximum variables in its non-paid version, which is the one that was used.

Table 4.2: Instance solutions

Parameter	**Instance 1**	**Instance 2**	**Instance 3**
Solution Objective	45.5	57.3	89.5
Vehicle 1 Route	0 > 3 > 1 > 2 > 0	0 > 1 > 2 > 6 > 0	0 > 4 > 0
Vehicle 2 Route	-	0 > 5 > 3 > 4 > 0	0 > 7 > 1 > 8 > 2 > 6 > 0
Vehicle 3 Route	-	-	0 > 5 > 3 > 0

The solutions presented above validated the viability and coherence of the suggested mathematical model. For every instance tested, optimal or near-optimal solutions were found, and all constraints, including those related to vehicle capacity, time windows, and load asymmetry, were successfully enforced. These results confirmed the formulation and its application, offering a solid basis for the development of the GRASP metaheuristic, in order to handle larger and more realistic examples, since the structure and constraint logic of the model were proved reliable.

4.3 Design of GRASP-based Solution

The GRASP-based algorithm was created to effectively handle larger and more realistic instances of the problem that are unsolvable with exact methods such as the one used in Chapter 3, building on the previously validated mathematical model.

4.3.1 General GRASP Structure

Each iteration of a GRASP algorithm consists of two main phases: the construction of a feasible solution, and a subsequent local search that attempts to improve this previously built solution. Due to its flexibility, and effectiveness, this method has been applied to a wide range of problems in logistics, since they usually need to deal with a lot of conditions. A generic GRASP pseudo-code, is presented in Algorithm 1.

```
procedure GRASP()
InputInstance();
for GRASP stopping criterion not satisfied do
    ConstructGreedyRandomizedSolution(Solution);
    LocalSearch(Solution);
    UpdateSolution(Solution, BestSolutionFound);
end
return BestSolutionFound;
```

Algorithm 1: Generic GRASP Procedure pseudo-code. Source: Feo and Resende (1995).

The main principle of GRASP consists of combining greedy heuristics with randomness. In the construction phase, represented in Algorithm 2 instead of deterministically selecting the best candidate element at each step, GRASP introduces a level of controlled randomness. Rather than choosing the best element as in pure greedy algorithms, it evaluates and creates a list of possible candidates based on a greedy function, which is the RCL. This list contains a subset of good candidates, from which one is selected at random and added to the partial solution that is being built. This methodology allows for the exploration of a diverse set of high-quality solutions and allows the algorithm to explore a wider solution space over multiple iterations.

```
procedure ConstructGreedyRandomizedSolution(Solution)
Solution = {};
for Solution construction not done do
    MakeRCL(RCL);
    s = SelectElementAtRandom(RCL);
    Solution = Solution ∪ {s};
    AdaptGreedyFunction(s);
end
```

Algorithm 2: Generic GRASP Construction Phase pseudo-code. Source: Feo and Resende (1995).

Once a complete solution is constructed, the local search phase is applied to refine the solution by exploring its neighborhood, which usually leads to improvements in solution quality. In order to find a more feasible solution, the algorithm methodically looks at minor adjustments to the current solution, such as reordering, swapping, or relocating elements. These modifications, known as neighborhood moves, are evaluated based on their impact on the objective function while ensuring that all the constraints of the problem remain satisfied. This process continues until no further improvement is found within the neighborhood, at which point the solution is considered locally optimal. In the context of this work, the local search intensifies the search around promising areas of the solution space, not only exploring different regions but also exploiting solutions effectively to enhance quality. The pseudo-code of Algorithm 3 represents a generic GRASP local search

phase procedure.

```
procedure local(P, N(P), s)
for s not locally optimal do
    Find a better solution t ∈ N(s);
    s ← t;
end
return s as local optimal for P;
```

Algorithm 3: Generic GRASP Local Search Phase pseudo-code. Source: Feo and Resende (1995).

In the context of the Rich VRP addressed in this work, GRASP was chosen for several reasons. First, the addressed problem includes a combination of challenging conditions such as vehicle side-loading restrictions, which is not common in VRP related problems, making the design of an exact heuristic difficult. GRASP's flexibility allows for the integration of these constraints directly into the solution construction logic and neighborhood moves. Additionally, GRASP's iterative structure makes it easy to integrate several improvement techniques without sacrificing its overall simplicity. Finally, GRASP has recently shown robust performance in related VRP contexts, such as in the work of Haddadene et al. (2016), making it a suitable approach to handle the scale and complexity of the problem faced by Company X.

4.3.2 GRASP Approach for Company X's Problem

To effectively solve the routing problem faced by Company X, a customized GRASP metaheuristic was developed, adapted to the specific operational constraints and characteristics of its distribution context, including a heterogeneous fleet, time windows, side-loading restrictions, and asymmetry between the left and right sides of each vehicle. The goal of this section is to detail the specific logic implemented in each phase of the algorithm.

4.3.2.1 Construction Phase

The construction phase of the GRASP algorithm tailored for the needs of Company X plays a critical role in generating the base for the optimal or near-optimal solutions that would be the final output, after the local search phase. This phase is designed to progressively assign customer orders to vehicle routes, every time these respect all the constraints related to Company X's problem, by combining greedy selection principles with a randomized component that enhances solution diversity. Instead of deterministically choosing the most cost-effective option at every step, the algorithm evaluates multiple insertion possibilities and selects among them using a probabilistic strategy. This approach allows the algorithm to explore different areas of the solution space between each iteration, avoiding settling too soon on solutions that are not globally optimal. As mentioned, each assignment of possible candidates is made with strict adherence to Company

X's operational constraints, such as vehicle capacity, customer time windows and load asymmetry requirements, ensuring that only valid partial solutions are used in the next steps of the algorithm's construction phase.

At the beginning of the procedure, all vehicles start with empty routes and all customer orders are marked as unassigned. After that, the algorithm enters an iterative loop that continues until all orders are successfully assigned. During each iteration, it evaluates all feasible ways of inserting each unassigned order into the route of each vehicle in any position. For each feasible insertion, the associated cost is computed based on how much it increases the route distance, through another method that calculates the cost of the trip between the local of the order that is being assigned and the last assigned order for the route in question. This is done by calculating the distance between the two points and multiplying it by the cost per kilometer of that vehicle.

Only feasible insertions are retained, these are the ones that respect the conditions of the problem. From these, the algorithm starts building a RCL, by selecting the insertions whose cost is within a threshold determined by a tunable parameter α, consisting of choosing only the insertions whose cost is below a certain value. This promotes the exploration of diverse solutions by allowing the selection of near-optimal insertions. After that, an element from the RCL is chosen randomly and inserted into the corresponding route, introducing the randomness associated with the GRASP metaheuristic in the algorithm.

After an order is inserted, the algorithm needs to calculate the times related to the insertion of that order, so it is possible to control the time window constraints of the next insertions. For that, it starts by recalculating the vehicle's arrival time at the customer location, which is done based on the travel time from the previous stop, a fixed unloading duration per stop, and a variable unloading component, proportional to the order size. Following this, the process is repeated until all orders are assigned to feasible positions in vehicle routes, guaranteeing that only feasible solutions are generated and passed to the second phase of the metaheuristic: the local phase.

A summary of the construction phase process is represented in Figure 4.2.

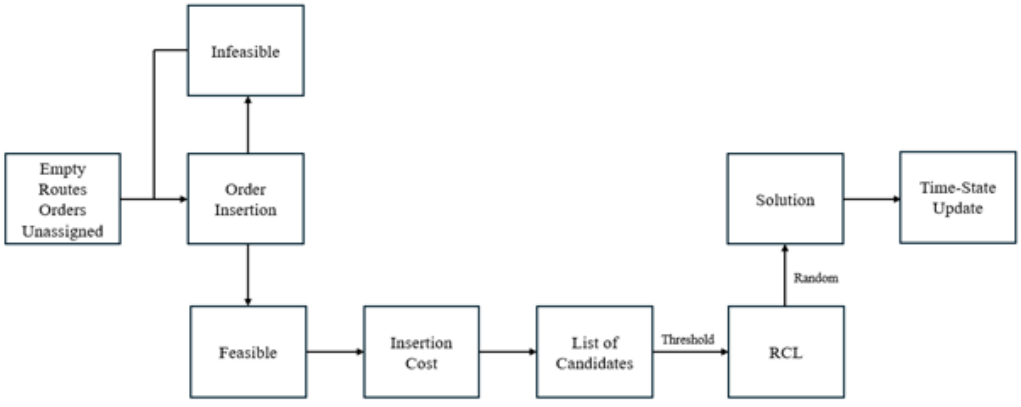

Figure 4.2: Construction Phase Summary

4.3.2.2 Local Search Phase

This phase aims to improve the quality of the solution by exploring its local neighborhood, which is defined as solutions that differ by a single relocation of an order, and iteratively attempting to achieve better configurations. This process continues until reaching a solution where no further improvement can be found through any individual move, known as the local optimum.

The local search implemented in this work follows a first-improvement descent strategy, which means that it is characterized by the objective of reducing the total cost of the solution. At each iteration, it examines the possibility of moving a customer from one position to another, either within the same route or to a different route entirely. This means that for every customer assigned to a vehicle route, the algorithm attempts to remove that customer and reinsert them in every possible position of every other route. This exhaustive but structured search ensures that a wide neighborhood is considered while avoiding the computational cost of full enumeration or backtracking.

Furthermore, each relocation attempt triggers a feasibility check to ensure that the solution remains valid under all problem constraints. This procedure is used to validate the updated routes and rebuild time-related state variables if necessary. It is critical to preserve the integrity of the solution by including this method, which ensures that improvements can only be accepted when they comply with the operational conditions imposed by the problem of Company X.

If a feasible relocation results in a lower total cost, based on the total travel distance multiplied by the vehicle cost per kilometer, the algorithm updates the best current solution by integrating the relocated element and restarts the search. This behavior is characteristic of a first-improvement approach, which operates as soon as an improvement move is identified, terminating the current iteration early, allowing the algorithm to proceed from the new solution found. This method avoids unnecessary computation and often leads to faster convergence, which was the reason for it being adopted to solve this problem, even knowing that the possibility of becoming trapped in local optimum exists.

The local search phase reaches an end when no improvement can be found in all possible relocation moves. At this point, the solution is considered locally optimal. Even though it may not be globally optimal, especially when the neighborhood structure is rich and leveraged, it typically represents a significant improvement over the initial construction.

A summary of the local search phase process is represented in Figure 4.3.

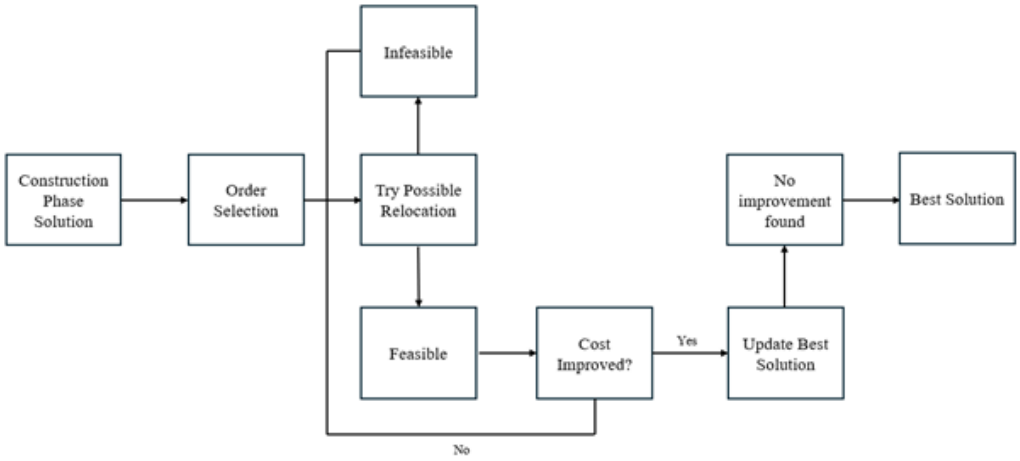

Figure 4.3: Local Search Phase Summary

4.3.2.3 GRASP Iteration and Stopping Criteria

The GRASP metaheuristic iterations go through successive rounds of solution construction and local search, aiming to progressively improve the overall solution quality. In each iteration, a new solution is constructed using the greedy randomized strategy, followed by the application of the local search procedure to reach a locally optimal configuration. If this new solution is better than the best solution found so far, it is stored.

To ensure a balance between solution quality and computational effort, a fixed number of iterations is adopted as stopping criteria. In this implementation, the algorithm runs for a predetermined number of iterations, which can be defined by the user and provides sufficient exploration of the solution space while keeping the execution time practical for every day use, since the delivery planning of Company X is done every work day. Alternative stopping criteria such as time limits were considered but not adopted. This approach allows the method to explore diverse regions of the solution space while maintaining simplicity and control over runtime, which cannot be extended.

4.3.2.4 GRASP Configuration and Tuning

The GRASP metaheuristic developed in this work was implemented in Python 3.10, which was chosen for its simplicity, flexibility, and also due to the author's familiarity and prior experience with its syntax and ecosystem, leading to a more efficient and structured development process. The implementation followed an object-oriented approach, with distinct classes created to represent vehicles, customer orders, and the routing solution itself. In addition, different *Python* scripts were created to maintain a modular coding structure, where each of them handled its own relevant data and logic, allowing a better understanding of external users.

It is important to clarify the organization of the code developed for the implementation of the GRASP algorithm, since five different *Python* scripts were created for that. First, the

data_loader.py module handles input data loading and preprocessing of postal codes associated with customer orders. After that, the script named *distance_travel_matrix.py* builds both the distance and travel time matrices between locations. Since a real route API such as Google Maps was not used, distances are calculated based on the coordinates of postal codes, through the haversine formula, which is represented in Equations 4.1, 4.2 and 4.3. In addition, the distances resultant from the haversine formula were multiplied by a factor of 1.4, due to the fact that haversine distances are considered in a straight line, which does not happen in real-world roads.

$$a = \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cdot\cos(\phi_2)\cdot\sin^2\left(\frac{\Delta\lambda}{2}\right) \tag{4.1}$$

$$c = 2\cdot\arctan 2\left(\sqrt{a}, \sqrt{1-a}\right) \tag{4.2}$$

$$d = R\cdot c \tag{4.3}$$

Where:

- ϕ_1, ϕ_2 are latitudes in radians,
- λ_1, λ_2 are longitudes in radians,
- R is the Earth's radius (6371 km),
- d is the distance between the two points.

The script with the name *constraints.py* contains the feasibility-checking functions and enforces all restrictions related to the problem, including time windows, vehicle capacities, customer-vehicle incompatibilities, and load asymmetry constraints. Furthermore, the core GRASP logic, which includes the construction phase, local search, and best solution tracking, is implemented in *grasp_vrp.py*. Finally, *main.py* script is responsible for the overall coordination of the process, including data loading, algorithm execution, and results presentation.

Object classes were created, where key operational parameters are stored for every vehicle, such as its route, remaining capacity, current location, and time state. Similarly, each order includes relevant information such as demand, status, time windows, delivery postal code, and delivery side request. The solution is stored as a dictionary that maps each vehicle to an ordered list of customer deliveries, allowing efficient access and manipulation during the construction and local search phases.

To ensure solution feasibility, specific validation functions were created and used throughout the algorithm. During the construction phase, a function was defined to verify that the addition of a new customer respects all relevant constraints, such as vehicle capacity, time windows, delivery side restrictions, and customer-vehicle compatibility. In the local search phase, a similar function

was created to evaluate the feasibility of a modified route and update the time variables accordingly. This control ensures that only valid solutions are accepted during the search. Furthermore, the quality of a solution is evaluated based on the total transportation cost, which is calculated as the sum of distances traveled by all vehicles, weighted by the respective cost per kilometer.

Regarding parameter settings, the algorithm uses a fixed number of iterations as a stopping criterion, ensuring control over execution time and repeatability. This decision is aligned with the operational needs of Company X, where the delivery planning process occurs on a daily basis and must be completed in a reasonable time. The level of greediness during solution construction is controlled by the parameter α, which defines the threshold for the Restricted Candidate List (RCL). Its value was defined as equal to 0.3 to maintain a reasonable compromise between exploration and exploitation.

Finally, the entire solution logic is encapsulated in a master routine that coordinates the construction and local search phases across iterations, always keeping track of the best solution found.

Results

This chapter presents the results obtained from applying the GRASP-based metaheuristic, described in the previous chapter to a real-world scenario, which are datasets of orders provided by Company X, for different working days. The main objective is to evaluate the algorithm's ability to generate feasible and efficient delivery plans considering the company's operational conditions, including vehicle capacities, time windows, customer delivery sides, and load asymmetry restrictions.

Although the results are not directly compared to the current planning method of the company, which is manual and difficult to measure, the generated solutions are discussed in terms of feasibility and overall transportation cost. This preliminary evaluation aims to assess the algorithm's practical applicability and serve as a foundation for future benchmarking against existing practices.

5.1 Dataset Description

The datasets used in this study come from real operational data provided by Company X. They represent complete lists of orders for specific working days, offering a realistic scenario to evaluate the performance of the developed GRASP metaheuristic. The raw data was obtained from an internal *Excel* spreadsheet used by the company, which included information such as customer name and number, ordered product, order quantity, delivery address and postal code, order status, time window, and required unloading side (left or right side of the vehicle).

Furthermore, the available fleet of Company X was considered, including vehicles' maximum load capacity, length and cost per kilometer, which represent the relevant attributes of the vehicle class that are considered in the algorithm. Compatibility constraints between specific orders and vehicles were also defined based on operational limitations, and delivery sides were classified into left or right according to unloading logistics at the customer sites.

In order to prepare both datasets for use in the algorithm, a structured preprocessing phase was carried out. This phase involved few simple steps:

- Filtering the Order spreadsheet data for a specific working day;
- Filtering the Order status by "For Delivery";
- Filtering the product type to steel rods of 12 meters, since the algorithm is going to be applied to this type of products;
- Associating the data types of the Order spreadsheet with the ones used in the algorithm;
- Corresponding Order spreadsheet information with Order class attributes;
- Associating the data types of the Vehicle spreadsheet with the ones used in the algorithm;
- Corresponding Vehicle spreadsheet information with Vehicle class attributes.

The defined instances derived from these datasets serve as the basis for evaluating the feasibility and efficiency of the proposed solution. The first and smaller instance includes a total of 24 customer orders, whose quantities represent approximately 37% of the sum of the maximum capacities of all available vehicles. This dataset is presented in Table 5.1. After that, the second and larger instance is shown in Table B.1, with 43 orders representing almost 100% of the sum of the maximum capacities of all vehicles, almost reaching the limit for the execution of the algorithm.

Table 5.1: Order dataset from Company X

Customer	Product	Quantity	Delivery Postal Code	Status	Unloading Side	Time Window Open	Time Window Close
Y	Steel Rod (12m)	2.00	8100-221	For Delivery	Left	8	17
Y	Steel Rod (12m)	0.01	7670-053	For Delivery	Right	8	17
Y	Steel Rod (12m)	0.06	7670-053	For Delivery	-	8	17
Y	Steel Rod (12m)	0.13	7670-053	For Delivery	-	8	17
Y	Steel Rod (12m)	0.02	7670-053	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8900-404	For Delivery	-	8	17
Y	Steel Rod (12m)	1.52	8000-072	For Delivery	-	8	17
Y	Steel Rod (12m)	1.00	8900-027	For Delivery	-	8	17
Y	Steel Rod (12m)	2.00	8900-027	For Delivery	-	8	17
Y	Steel Rod (12m)	1.00	8900-027	For Delivery	-	8	17
Y	Steel Rod (12m)	0.40	8125-406	For Delivery	Left	8	17
Y	Steel Rod (12m)	0.40	8125-406	For Delivery	Right	8	17
Y	Steel Rod (12m)	4.00	8125-406	For Delivery	-	8	17
Y	Steel Rod (12m)	0.22	8125-406	For Delivery	-	8	17
Y	Steel Rod (12m)	0.30	7630-569	For Delivery	-	8	17
Y	Steel Rod (12m)	0.19	7630-569	For Delivery	-	8	17
Y	Steel Rod (12m)	0.43	7630-569	For Delivery	Right	8	17
Y	Steel Rod (12m)	0.92	8200-416	For Delivery	-	8	17
Y	Steel Rod (12m)	0.88	7645-309	For Delivery	-	8	17
Y	Steel Rod (12m)	0.88	7645-309	For Delivery	-	8	17
Y	Steel Rod (12m)	4.60	8005-143	For Delivery	-	8	17
Y	Steel Rod (12m)	4.60	8005-143	For Delivery	-	8	17
Y	Steel Rod (12m)	0.88	8200-600	For Delivery	-	8	17
Y	Steel Rod (12m)	0.88	8200-600	For Delivery	-	8	17

A summary of the properties of the available vehicles of Company X is presented in Table 5.2.

Table 5.2: Vehicle dataset from Company X

Vehicle Nr.	Plate	Maximum Load Capacity	Length	Cost/km (€)
7	XX-XX-XX	1	7.3	0.39
8	XX-XX-XX	9	12	0.54
9	XX-XX-XX	9	12	0.54
10	XX-XX-XX	10	12	0.54
11	XX-XX-XX	10	12	0.54
15	XX-XX-XX	10	13	0.54
16	XX-XX-XX	12	13.5	0.54
17	XX-XX-XX	15	13.5	0.54

5.2 GRASP Execution Details

The GRASP algorithm was executed to solve a daily routing instance composed of a set of customer orders and a fleet of available vehicles. The procedure followed the classic GRASP structure: an iterative combination of a greedy randomized construction phase and a local search improvement phase. As mentioned previously, both phases were carefully designed to handle the particular constraints of the company's operations, including heterogeneous vehicles, time windows, requested unloading sides, and asymmetric loading constraints.

The execution of the GRASP algorithm for a different number of maximum iterations in three independent tests reveals important information about performance consistency, solution quality, and computational effort. For that, all experiments were conducted on a Lenovo ThinkPad E14 with an Intel Core Ultra 7 155H chip and 16GB of RAM, without requiring any form of parallelism or hardware acceleration. Execution times were consistently within acceptable limits for daily use, as shown in both cases, demonstrating that the algorithm is computationally viable for real-time or near-real-time routing decisions.

The stopping criteria for the algorithm was defined as a fixed number of iterations. For the first and smaller instance (Table 5.1), the algorithm was executed considering different iteration limits, for two values of α. The computational time, as well as the best cost found and number of iterations for each scenario are represented in Table 5.3 and 5.4.

Table 5.3: Test results for different limit of iterations ($\alpha = 0.3$)

Max Iterations	Run 1		Run 2		Run 3	
	Computing Time (s)	Best Cost (€)	Computing Time (s)	Best Cost (€)	Computing Time (s)	Best Cost (€)
1	19	434.6	18	421.5	18	510.9
2	20	430.4	20	390.6	20	411.3
5	28	394.4	26	371.6	27	380.8
10	37	409.9	38	340.6	36	390.5
20	54	382.1	56	370.5	55	356.6
50	130	349.4	123	332.6	140	332.7
100	227	318.4	224	331.6	231	322.0
200	444	328.5	444	338.5	514	328.6

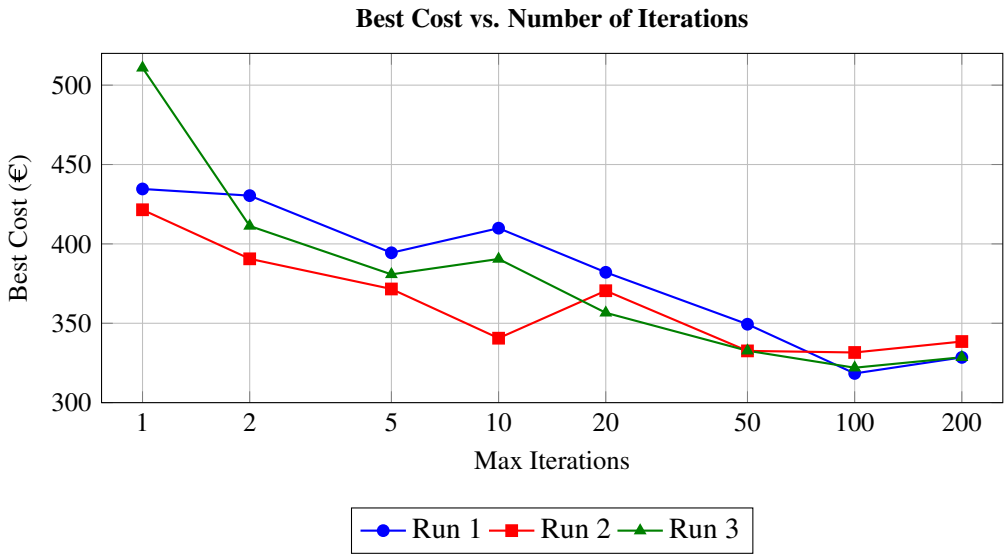

Figure 5.1: Solution best cost for different iteration limits ($\alpha = 0.3$)

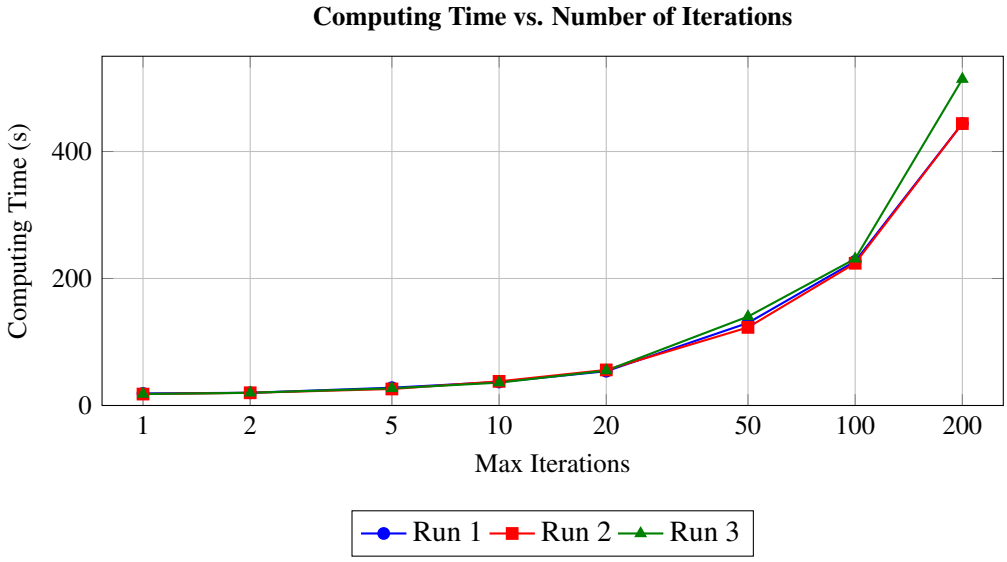

Figure 5.2: Solution computing time for different iteration limits ($\alpha = 0.3$)

For α equal to 0.3, it is evident that the quality of the solution improves with an increasing number of iterations, as shown in Figure 5.1. This comes from the fact that, in all three tests, the best cost obtained at lower iteration values (such as 1 or 2) is considerably higher than the costs obtained when the algorithm is allowed to run for 100 or 200 iterations, with an average cost reduction of about 25%. This confirms the expected behavior of GRASP, where more iterations allow for a larger exploration of the solution space, increasing the probability of finding a better solution. However, this improvement is not linear, since the gains from moving from 5 to 50 iterations are substantial, the improvements between 100 and 200 iterations are minimal, with 100

iterations showing better results than the case for 200. This indicates that excessively high iteration limits may lead to longer computing times without proportionate benefits in solution quality.

Represented in Figure 5.2, the computational time increases exponentially with the number of iterations, which was expected since each iteration involves both a constructive and a local search phase, that are computationally demanding. For example, the average computing times increase from less than 30 seconds for up to 5 iterations to more than 400 seconds for 200 iterations. This aspect is particularly relevant in the context of daily operational planning, such as in the case of Company X, where the algorithm must produce a solution in a reasonable time. It supports the need to define a maximum number of iterations that balances quality and efficiency, which will be around 100 from what was observed in the previous tests.

Another relevant output is the natural variability in solution quality across the three runs for each iteration limit. This is a direct consequence of the randomized elements in the GRASP construction phase, which influence the starting point of each local search, which is different for every iteration. Although this greedy randomness benefits a larger exploration of the solution space, it also means that different runs result in different solutions, especially when the iteration limit is low, as for the results for 1 iteration, where variability is high. However, for higher iteration limits, this variability tends to reduce, with the algorithm consistently converging toward similar values for the best cost found. This suggests that the algorithm becomes more robust and stable with more iterations, reinforcing the value of longer runs when a higher computational time is allowed.

Table 5.4: Test results for different limit of iterations ($\alpha = 0.5$)

Max Iterations	**Run 1**		**Run 2**		**Run 3**	
	Computing Time (s)	**Best Cost (€)**	**Computing Time (s)**	**Best Cost (€)**	**Computing Time (s)**	**Best Cost (€)**
1	20	491.7	24	460.3	21	416.3
2	25	503.0	25	342.4	24	396.4
5	37	394.1	33	380.5	35	405.5
10	45	411.2	43	375.3	48	371.0
20	88	399.8	82	373.7	81	391.4
50	181	377.8	190	337.5	173	344.1
100	341	340.4	352	378.2	327	376.4
200	707	301.0	551	322.2	643	321.8

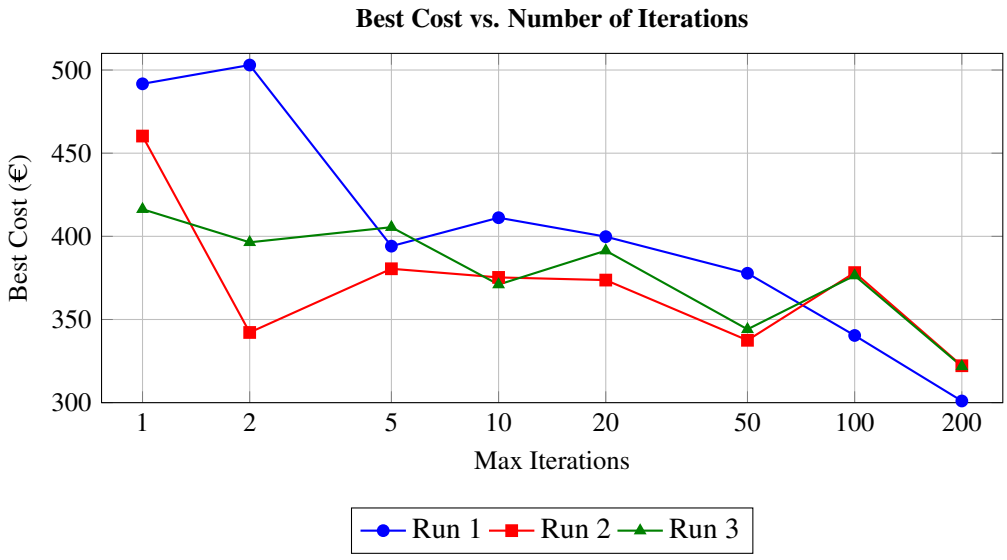

Figure 5.3: Solution best cost for different iteration limits ($\alpha = 0.5$)

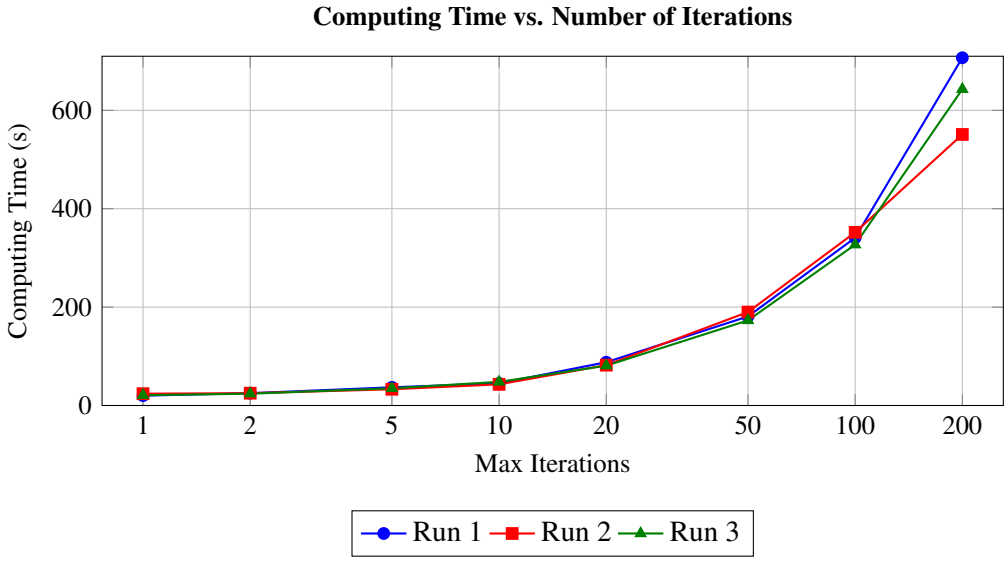

Figure 5.4: Solution computing time for different iteration limits ($\alpha = 0.5$)

On the other hand, for α equal to 0.5, similar trends are observed, where the quality of the solution tends to improve as the number of iterations increases (Figure 5.3) and the variability across the three different runs is common. In all three runs, the best costs achieved with a limited number of iterations are nearly 30% higher than those obtained with a larger iteration limit, confirming that the GRASP metaheuristic benefits from extended search effort. This demonstrates the expected behavior of GRASP: increasing the iteration limit allows for a more extensive exploration of the solution space, which enhances the probability of finding a better solution. However, not like in the first case, there was a huge cost reduction for a limit of 200 iterations, but also a higher

computing time (Figure 5.4), which clearly shows the trade-off between run times and solution quality.

After the small instance, it was decided to test a more complex scenario in which delivery demands are almost equal to the sum of the maximum capacities of the vehicles available in Company X. The details from the runs performed for this data set are presented in Table 5.5.

Table 5.5: Test results for different limit of iterations ($\alpha = 0.3$)

	Run 1			Run 2		
Max Iterations	**Computing Time (s)**	**Best Cost (€)**	**Failed Iterations**	**Computing Time (s)**	**Best Cost (€)**	**Failed Iterations**
1	17	No solution	1	16	No solution	1
2	17	No solution	2	17	No solution	2
5	18	954.4	4	17	No solution	5
10	19	880.5	9	18	912.4	8
20	19	911.5	19	18	No solution	20
50	24	893.3	45	27	869.4	42
100	32	835.5	89	33	820.2	86
200	47	838.4	181	55	846.6	174
500	104	778.6	443	118	793.9	432
1000	186	717.1	892	199	798.3	881

The results obtained from running the GRASP algorithm in a more complex instance, which significantly increases the difficulty of the problem, and the results clearly reflect that. At lower iteration limits, the algorithm was unable to produce any feasible solution in either of the two runs, highlighting the complexity and the need for a broader and intense exploration of the solution space. As the maximum number of iterations increases, the algorithm begins to find some feasible solutions by improving the number of successful iterations. For example, from 20 iterations onward, feasible solutions are consistently obtained, and the associated best cost improves significantly as the iteration limit increases.

However, similar to the previous experiments with smaller instances, improvements come at the cost of higher computing times. This suggests that beyond a certain point, increasing the number of iterations does not proportionally improve the solution quality and may become inefficient in practice. In addition, including the number of failed iterations provides additional insight into the algorithm's performance. It quantifies how often the algorithm was unable to generate a feasible solution in a given run, and proves to be a useful indicator of how the algorithm copes with the tight feasibility constraints of a heavily loaded instance. These findings set the importance of defining a good relation between the maximum number of iterations and the number of successful iterations for each considered instance.

Finally, these results highlight a good balance between solution quality and computational runtime for the developed GRASP-based algorithm. While a higher number of iterations generally produces better results, the associated increase in computational time may not be justifiable in time-sensitive scenarios, once the quality of the solution is similar for more than 100 iterations. In the case of Company X, where delivery plans must be generated daily, an iteration limit of around

100 was found to provide a good compromise between the production of high-quality solutions in approximately four minutes. This configuration ensures that the algorithm remains practical for everyday use, while still delivering significant improvements in routing efficiency.

5.2.1 Results Benchmark

In order to assess the practical effectiveness of the proposed GRASP-based solution, its performance must be benchmarked against the current operational strategy employed by Company X. The comparison is based on real data from the company, namely the total transportation cost in 2024. This value was then divided by the company working days in that year, resulting in an average daily transportation cost of approximately 642 euros, represented in Table 5.6. The current method consists of manually planned deliveries, often based on experience and operational constraints, but without the support of an optimization tool. The objective of this benchmark is to determine whether the proposed algorithm can provide valid and cost-effective delivery plans that reduce Company X's current transportation cost and delivery lead times.

Table 5.6: Company X Transportation Costs (2024)

Description	**Instance 1**
Total Transportation Cost (2024)	160428,12 €
Working Days (2024)	250
Average Daily Transportation Cost	641,71 €

The values that resulted from both tested instances represent a 60% reduction in delivery lead times for Company X since the larger instance would normally take 4 to 5 working days to be completely delivered, but considering that the company delivers other types of materials apart from steel rods, which represent low quantities and were not considered for the development of the solution.

With that, considering a normal day an average of both instances, the average daily transportation cost after the implementation of the algorithm would be near the value of 550 euros. From that, a reduction of 14,3% in the total transportation cost would be expected, adding an extra 4,3 percentage points improvement to the predefined objective.

Conclusions and Future Work

This dissertation addressed the problem of optimizing transport routes and load planning in a real-world logistics context, where decisions are entirely made by humans, through the development and implementation of an algorithmic approach based on the GRASP metaheuristic. The application of this method to real operational data from a company allowed for an evaluation of its effectiveness in producing high-quality solutions under real logistic restrictions.

A mathematical formulation of the company's model was developed, characterizing all the operational conditions that need to be accounted for. After that, this model was tested using IBM ILOG CPLEX Optimizer, to validate its feasibility in small instances, before trying to handle large amounts of data. The results of these tests indicated that the built model was valid, taking the project to the next step: the development of a GRASP algorithm, in *Python*, that translated the previously validated model and could handle larger instances of data and different types of data files. Lastly, the GRASP algorithm was tested with real data from Company X and provided overall good solutions.

The results showed a clear relationship between the number of iterations considered and the quality of the solutions obtained. It became evident that increasing the number of iterations generally leads to significant cost reductions, with improvements of approximately 25% when comparing low iteration limits (1 or 2) with higher values such as 100 or 200 iterations. However, the improvement was not linear, since moving from 5 to 50 iterations resulted in considerable gains, but the difference between 100 and 200 iterations was minimal, with the 100-iteration case even showing slightly better results, within less computing time. This suggests that excessively high iteration limits may lead to longer computation times without proportional improvements in solution quality, reinforcing the importance of identifying an appropriate trade-off between runtime and solution quality. In the context of Company X, it is important to achieve it because they follow a daily delivery planning and it should not be time-consuming.

Despite the promising outcomes, it was not possible to directly compare the algorithm's performance with the current transport planning practices at Company X, due to the lack of cost data from the existing process. As such, a quantitative validation of the practical benefits could not

be conducted. However, based on the cost reductions observed in the computational experiments, it is reasonable to expect that the implementation of this method would lead to the achievement of the predefined objectives, such as the 10% reduction in total transportation costs and the 30% reduction in average delivery lead time.

Future developments should begin with a direct comparison between the costs associated with the routes generated by the algorithm and the actual transportation costs incurred by Company X, to assess the real-world value of the developed solution. Following that, integrating the model with a real-time routing API such as Google Maps would allow for a more precise solution, since distances and travel times are currently calculated based on postal codes and not on real addresses and road distances. In addition, the development of a Power BI dashboard would help the company in monitoring the implementation of the algorithm over time, providing visual insights into cost trends, route efficiency, and even maps showing the routes for the day.

In conclusion, this work demonstrates the potential of algorithmic methods to significantly enhance transport planning processes and a significant improvement in the optimization of logistics operations in the metal distribution sector. Although the current solution already shows strong results, there remains substantial room for refinement and expansion by addressing common inefficiencies, with a clear path forward for future research and practical implementation, enhancing potential to build a more cost-effective logistics framework.

Bibliography

Baldacci, R., Mingozzi, A., and Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218:1–6.

Berger, J. and Barkaoui, M. (2003). *A Hybrid Genetic Algorithm for the Capacitated Vehicle Routing Problem*, pages 646–656. Springer.

Bortfeldt, A. and Wäscher, G. (2013). Constraints in container loading – a state-of-the-art review. *European Journal of Operational Research*, 229:1–20.

Bräysy, O. and Gendreau, M. (2005a). Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation Science*, 39:104–118.

Bräysy, O. and Gendreau, M. (2005b). Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation Science*, 39:119–139.

Chao, I.-M., Golden, B. L., and Wasil, E. A. (1998). *A New Algorithm for the Site-Dependent Vehicle Routing Problem*, pages 301–312. Springer.

Chao, I.-M. and Liou, T.-S. (2005). *A New Tabu Search Heuristic for the Site-Dependent Vehicle Routing Problem*, pages 107–119. Springer.

Clarke, G. and Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12:568–581.

Cordeau, J.-F., Laporte, G., Savelsbergh, M. W., and Vigo, D. (2007). *Vehicle Routing*, volume 14, pages 367–428. Elsevier.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem.

Deloitte (2024). Restructuring the supply base: Prioritizing a resilient, yet efficient supply chain.

Desrosiers, J., Dumas, Y., Solomon, M. M., and Soumis, F. (1995). *Time constrained routing and scheduling*, volume 8, pages 35–139. Elsevier.

DispatchTrack (2025). The top 7 building products and supplies delivery challenges.

Dorigo, M., Maniezzo, V., and Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26:29–41.

Feo, T. A. and Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133.

Gambardella, L. M., Éric Taillard, and Agazzi, G. (1999). Macs-vrptw: A multiple ant colony system for vehicle routing problems with time windows. *New Ideas in Optimization*, pages 63–76.

Gendreau, M., Hertz, A., and Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, 40:1276–1290.

Ghiani, G., Laporte, G., and Musmanno, R. (2003). *Introduction to Logistics Systems Planning*. Wiley.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13:533–549.

Golden, B., Assad, A., Levy, L., and Gheysens, F. (1984). The fleet size and mix vehicle routing problem. *Computers and Operations Research*, 11:49–66.

Haddadene, S. R. A., Labadie, N., and Prodhon, C. (2016). A grasp × ils for the vehicle routing problem with time windows, synchronization and precedence constraints. *Expert Systems with Applications*, 66:274–294.

Hartmanis, J. (1982). Computers and intractability: A guide to the theory of np-completeness (michael r. garey and david s. johnson). *SIAM Review*, 24:90–91.

Hemmelmayr, V. C., Cordeau, J.-F., and Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers and Operations Research*, 39:3215–3228.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. The MIT Press.

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59:345–358.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, 24:1097–1100.

Ombuki, B., Ross, B. J., and Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24:17–30.

Polacek, M., Hartl, R. F., Doerner, K., and Reimann, M. (2004). A variable neighborhood search for the multi depot vehicle routing problem with time windows. *Journal of Heuristics*, 10:613–627.

Porter, M. E. (1985). *Competitive Advantage: Creating and Sustaining Superior Performance*. New York: Free Press.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers and Operations Research*, 31:1985–2002.

Prins, C. (2009). *A GRASP × Evolutionary Local Search Hybrid for the Vehicle Routing Problem*, pages 35–53. Springer.

Solomon, M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254–265.

Subramanian, A., Uchoa, E., and Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers and Operations Research*, 40:2519–2531.

Suárez, J. G. and Anticona, M. T. (2010). *Solving the Capacitated Vehicle Routing Problem and the Split Delivery Using GRASP Metaheuristic*, pages 243–249. Springer.

Taillard, E. D. (1999). A heuristic column generation method for the heterogeneous fleet vrp. *RAIRO - Operations Research*, 33:1–14.

Vidal, T., Crainic, T. G., Gendreau, M., and Prins, C. (2013a). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, 231:1–21.

Vidal, T., Crainic, T. G., Gendreau, M., and Prins, C. (2013b). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers and Operations Research*, 40:475–489.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183:1109–1130.

Yu, B., Yang, Z.-Z., and Yao, B. (2009). An improved ant colony optimization for vehicle routing problem. *European Journal of Operational Research*, 196:171–176.

Çağrı Koç, Bektaş, T., Jabali, O., and Laporte, G. (2016). Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research*, 249:1–21.

Éric Taillard, Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31:170–186.

CPLEX Instances - Distance and Travel Time Matrices

Table A.1: Distance Matrix used for Instance 1

	1	2	3	4
1	0	15	11	12.5
2	15	0	7	15
3	11	7	0	13
4	12.5	15	13	0

Table A.2: Travel Time Matrix used for Instance 1

	1	2	3	4
1	0	30	22	25
2	30	0	14	30
3	22	14	0	26
4	25	30	26	0

Table A.3: Distance Matrix used for Instance 2

	1	2	3	4	5	6	7
1	0	10	11	12.5	7.5	5	6
2	10	0	8.5	15	8.5	10	6.5
3	11	8.5	0	9	6	8.5	4
4	12.5	15	9	0	3.5	8	11.5
5	7.5	8.5	6	3.5	0	9.5	7.5
6	5	10	6.5	8	9.5	0	7
7	6	6.5	4	11.5	7.5	7	0

Table A.4: Travel Time Matrix used for Instance 2

	1	2	3	4	5	6	7
1	0	20	22	25	15	10	12
2	20	0	17	30	17	20	13
3	22	17	0	18	12	17	8
4	25	30	18	0	7	16	23
5	15	17	12	7	0	19	15
6	10	20	13	16	19	0	14
7	12	13	8	23	15	14	0

Table A.5: Distance Matrix used for Instance 3

	1	2	3	4	5	6	7	8	9
1	0	10	11	12.5	7.5	5	6	10	15
2	10	0	8.5	15	8.5	10	6.5	8	9
3	11	8.5	0	9	6	8.5	4	10.5	11
4	12.5	15	9	0	3.5	8	11.5	20	23
5	7.5	8.5	6	3.5	0	9.5	7.5	12	15
6	5	10	6.5	8	9.5	0	7	7	6
7	6	6.5	4	11.5	7.5	7	0	12	11
8	10	8	10.5	20	12	7	12	0	18
9	15	9	11	23	15	6	11	18	0

Table A.6: Travel Time Matrix used for Instance 3

	1	2	3	4	5	6	7	8	9
1	0	20	22	25	15	10	12	20	30
2	20	0	17	30	17	20	13	16	18
3	22	17	0	18	12	17	8	21	22
4	25	30	18	0	7	16	23	40	46
5	15	17	12	7	0	19	15	24	30
6	10	20	13	16	19	0	14	14	12
7	12	13	8	23	15	14	0	24	22
8	20	16	21	40	24	14	24	0	36
9	30	18	22	46	30	12	22	36	0

Order Dataset

Table B.1: Order dataset from Company X

Customer	Product	Quantity	Delivery Postal Code	Status	Unloading Side	Time Window Open	Time Window Close
Y	Steel Rod (12m)	1.84	8005-527	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8005-527	For Delivery	-	8	17
Y	Steel Rod (12m)	1.00	8700-491	For Delivery	-	8	17
Y	Steel Rod (12m)	0.76	8700-491	For Delivery	-	8	17
Y	Steel Rod (12m)	1.00	8135-024	For Delivery	-	8	17
Y	Steel Rod (12m)	1.00	8135-024	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8600-281	For Delivery	-	8	17
Y	Steel Rod (12m)	0.80	8550-255	For Delivery	-	8	17
Y	Steel Rod (12m)	0.60	8550-255	For Delivery	-	8	17
Y	Steel Rod (12m)	0.24	8550-255	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8400-525	For Delivery	-	8	17
Y	Steel Rod (12m)	0.48	8200-416	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8500-001	For Delivery	-	8	17
Y	Steel Rod (12m)	0.08	8500-001	For Delivery	-	8	17
Y	Steel Rod (12m)	0.32	8500-001	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8500-001	For Delivery	-	8	17
Y	Steel Rod (12m)	0.40	8600-156	For Delivery	-	8	17
Y	Steel Rod (12m)	0.40	8600-156	For Delivery	-	8	17
Y	Steel Rod (12m)	0.18	8125-213	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8900-038	For Delivery	-	8	17
Y	Steel Rod (12m)	1.84	8900-038	For Delivery	-	8	17
Y	Steel Rod (12m)	7.36	8125-213	For Delivery	-	8	17
Y	Steel Rod (12m)	0.24	8125-213	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8125-213	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8125-213	For Delivery	-	8	17
Y	Steel Rod (12m)	1.04	8400-535	For Delivery	-	8	17
Y	Steel Rod (12m)	8.40	8400-535	For Delivery	-	8	17
Y	Steel Rod (12m)	0.48	8400-535	For Delivery	-	8	17
Y	Steel Rod (12m)	0.48	8200-428	For Delivery	-	8	17
Y	Steel Rod (12m)	0.48	8200-428	For Delivery	-	8	17
Y	Steel Rod (12m)	2.76	8200-428	For Delivery	-	8	17
Y	Steel Rod (12m)	0.92	8200-428	For Delivery	-	8	17
Y	Steel Rod (12m)	4.80	8125-481	For Delivery	-	8	17
Y	Steel Rod (12m)	4.80	8125-481	For Delivery	-	8	17
Y	Steel Rod (12m)	1.76	8400-556	For Delivery	-	8	17
Y	Steel Rod (12m)	0.88	8400-556	For Delivery	-	8	17
Y	Steel Rod (12m)	2.64	8400-556	For Delivery	-	8	17
Y	Steel Rod (12m)	0.88	8400-556	For Delivery	-	8	17
Y	Steel Rod (12m)	5.00	8900-276	For Delivery	-	8	17
Y	Steel Rod (12m)	5.00	8900-276	For Delivery	-	8	17
Y	Steel Rod (12m)	1.76	8600-292	For Delivery	-	8	17
Y	Steel Rod (12m)	2.64	8600-292	For Delivery	-	8	17
Y	Steel Rod (12m)	4.40	8600-292	For Delivery	-	8	17