A Comparative Analysis Between the Continuous and Discrete Location Routing Problem

Vos, J.P.E.

Award date:
2023

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Industrial Engineering and Innovation Sciences
Operations Planning Accounting and Control Research Group

A Comparative Analysis Between the Continuous and Discrete Location Routing Problem

Master Thesis

J.P.E. Vos

Supervisors:
Dr. L. Martin
Dr. R. van Lieshout

version 1.0

Eindhoven, May 2023

Abstract

The Location Routing Problem is a combinatorial optimization problem in which the strategic placement of depots is based on operational routing decisions. Two conventional potential methods to locate depots are continuous and discrete. In the continuous variant, uncountable infinite potential depot locations exist, while in the discrete method, the depot location is limited to customer locations. This thesis provides a comparative analysis of the influence of instance characteristics on continuous and discrete Location Routing Problems. The objective is to understand if and why certain characteristics, such as customer distribution, time windows, and capacity constraints, disproportionally influence the solution quality of the continuous and discrete methods. We use a Variable Neighborhood Search to solve the problem. This meta-heuristic solution method utilizes a set of different solution structures to repeatedly improve the incumbent solution and escape local optima. Computational experiments, using data sets of modified Solomon Instances show that a combination of customer distribution and time windows primarily affect relative performance between the continuous and discrete methods. A highly clustered distribution of customers in combination with tight time windows is detrimental to the performance of the discrete method relative to the continuous method.

Contents

Chapter 1

Introduction

The need for more efficient supply chains is ever-growing due to globalization and an increased need for sustainability. However, the most significant challenge in modern supply chains is labor shortage (New York Times, 2021). In particular, a lack of truckers is a major bottleneck for supply chains. Due to difficult working conditions and irregular working hours, the employee turnover rate for large trucking companies is as high as 90%. For this reason, logistical companies must optimize their networks as much as possible to reduce labor costs and employee turnover by optimizing route length and reducing irregular working hours. A transportation network consists of a depot from which vehicles leave and a set of locations that the vehicles have to visit. Optimizing such a network is a complicated task as the routes depend on the depot location, and simultaneously, the best depot location depends on the routes. As such, this problem calls for an integrated decision-making process.

1.1 Importance of the Location Routing Problem

In 2004, an Austrian parcel delivery service must decide whether to redesign its transportation network or stay with the current setup. Due to increasing competition, choosing the most cost-efficient option is vital. Furthermore, their network covers the entire country of Austria. On this scale, even minor changes in the transportation network can lead to considerable fluctuations in profitability. Wasner and Zäpfel (2004) determine the best possible way forward by modeling the problem as a Location Routing Problem (LRP). The LRP differs from a classic Facility Location Problem (FLP) because the LRP includes potential routes between demand points in the decision-making process instead of just the demand points. The study shows that redesigning the entire network will lead to a 14.7% reduction in operating costs.

Nanchuan, China, is a city known for the bauxite mining industry. Mining the material produces significant amounts of hazardous waste. This waste, often incendiary or even explosive, must be transported from mining sites to collection centers and eventually to a specialized recycling center, spread out over 2600 km^2. This type of problem adds complexity to the LRP. Furthermore, transporting hazardous waste comes with significant environmental and social risks, so a trade-off exists between optimizing for costs and optimizing for risk. Zhao and Ke (2017) tackle this problem by developing a multi-objective LRP. Their numerical experiments show the model can attain a 34% cost reduction and a 57% risk reduction.

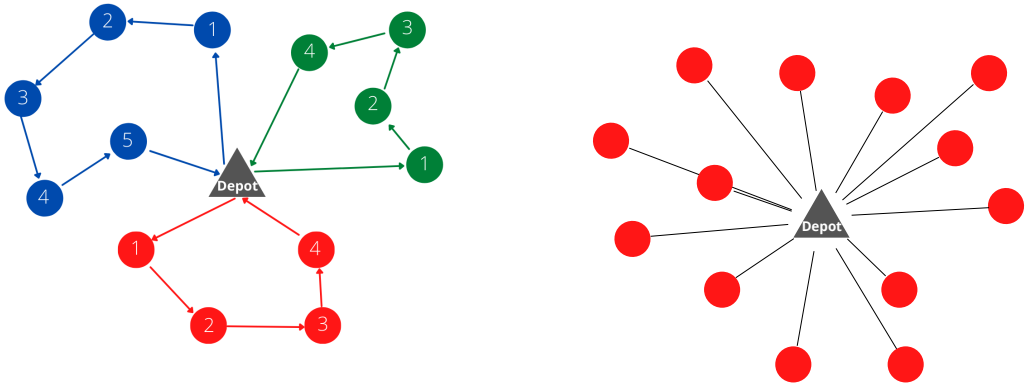
1.2 Vehicle Routing and Location Problems

The cases mentioned earlier demonstrate that solving the LRP provides significant benefits if used correctly. The LRP combines two sub-problems; the Vehicle Routing Problem (VRP) and the Facility Location Problem (FLP). The VRP aims to find the shortest possible set of routes that pass each customer location. Each route starts and ends at the depot, which has a fixed location, as opposed to the LRP, in which the depot location is a decision. Figure 1.1a provides a visualization of a VRP with three routes. The VRP has many variants that aim to simulate real-world situations, such as the capacitated VRP, the VRP with time windows, and the multi-depot VRP. Kumar and Panneerselvam (2012) provide an overview of VRP variants and solution methods.

Dantzig and Ramser (1959) introduce the VRP. They describe a real-world application concerning the delivery of gasoline to service stations. They propose an algorithmic approach that provides "near-best" solutions to the problem. Clarke and Wright (1964) propose a savings heuristic that improves on the Dantzig-Ramser approach (Toth & Vigo, 2002). A heuristic is a type of strategic solution method often using "rule of thumb" decisions and does not optimality (Romanycia & Pelletier, 1985). Since the introduction of the VRP, it has received much scientific interest. Even to this day, the VRP remains enormously relevant, demonstrated by Elshaer and Awad (2020) as they review 299 articles concerning the VRP, published between 2009 and 2017. Appendix A provides a formal model of a VRP. The model includes capacity and time window constraints, which are among the most popular features in the literature.

The other sub-problem of the LRP is the FLP. Like the VRP, the FLP consists of customer locations and a depot. However, this problem optimizes the depot's location instead of optimizing routes. Since the FLP does not consider routes, only the distance from each customer location to the depot location is important. Therefore, the depot location with the feasibly lowest total distance to each customer location is optimal. Figure 1.1b visualizes the FLP.

There are two main variations of the FLP. In the *continuous* variant, the depot has an unlimited set of potential locations within certain boundaries. The solution space for the depot is uninterrupted and, therefore, continuous in essence. In the *discrete* variant, the depot has a limited set of potential locations. The solution space consists of a finite number of points. This distinction also translates into the main problem; the LRP. O'Kelly (1986) formulates the first mathematical programming model to solve the discrete FLP, but the proposed solution method only applies to the continuous FLP. Hence, the author chooses the closest potential location to the found solution location.



(a) Example Solution VRP

(b) Example Solution FLP

Figure 1.1: Visualization VRP and FLP

1.3 Literature on the LRP

The idea of interdependency between routing and facility locations is not a new concept. Although not yet formalized as the LRP, Boventer (1961) discusses the importance of the simultaneous determination of transport and facility decisions. Maranzana (1964) develops one of the earlier algorithms to find depot locations based on a known routing configuration. The author uses the center of mass to find depot locations. In the following years, researchers made steady progress in facility location research with more notable contributions from O'Kelly (1986), who develops a mathematical formulation and a discrete solution method and Campbell (1992), who introduces the multi-allocation model. Shortly after, Campbell (1994) develops the first linear integer formulation of the problem. Aykin (1995) introduces the first solution algorithm, which considers the routing and facility location together, marking the start of LRP research.

Since then, the LRP has gained a lot of scientific interest, and there are no signs of it slowing down for the foreseeable future. Mara et al. (2021) shows a trend in increasing interest from 2015 to 2021, with an emphasis on more complex problems. A recent focal point is in multi-objective problems, such as in Ghasemi et al. (2022), which optimizes costs and reliability of delivery. Yu, Zhou et al. (2020) develop a general multi-objective model for waste collection applications, in which they transform secondary objectives into monetary objectives. Fallahtafti et al. (2021) propose a very specialized and unique model, in which the secondary objective is to minimize the risk of robbery for cash-in-transit vehicles.

Recent solution methods for the LRP are overwhelmingly from the meta-heuristic category. For example, Chobar Pourghader et al. (2021), Fallahtafti et al. (2021), Pasha et al. (2022), Pourmohammadi et al. (2021), Yan et al. (2023) and Yu, Zhou et al. (2020) all use a meta-heuristic solution method, mainly because an exact method would be too computationally expensive. Still, exact solution methods are not entirely deserted by the scientific community, as Darvish et al. (2019) and Nucamendi-Guillén et al. (2022) demonstrate. Exact methods are still relevant in specific situations as they provide higher-quality solutions for smaller instances. Recently, a new type of hybrid solution method that combines simulation and heuristics called a *simheuristic* is gaining traction among academicians. Guimarans et al. (2018), Quintero-Araujo et al. (2019) and Reyes-Rubiano et al. (2019) use a simheuristic to solve the routing. This method excels in stochastic models, in which, for example, customer demand is uncertain.

Based on the current trend in academic research, the LRP will be a topic of interest for many more years. As knowledge in this field matures, the direction of research is headed toward more complicated models. The models account for an increasing number of variables, such as environmental impact and reliability. Solution methods are still heavily dominated by meta-heuristics, but a trend is showing an increasing interest in hybrid solutions methods. Combining multiple types of solution methods that complement each other is a great way to achieve higher-quality solutions in less time. One potential issue with current solution methods is that they often need to be tuned for specific instances, limiting their general applicability. One way around this obstacle is to develop more adaptive algorithms that can automatically tune parameters.

1.4 Continuous and Discrete Models

Although the LRP has two variants, almost all studies focus on the discrete LRP. A taxonomy by Mara et al. (2021) classifies variants of the LRP and finds that only five out of 216 articles focus on the continuous LRP. Although the authors do not discuss this discrepancy in scientific interest, one could theorize several arguments. The most probable is that researchers want to use a solution method that is incompatible with the continuous LRP. Another reason to use the discrete LRP is that the use case requires a solution chosen from a pre-determined set of potential locations. In the discrete LRP, the placement of the hub must be on a pre-determined node (point), as opposed to the continuous LRP in which uncountable infinite possible locations exist. Nevertheless, the continuous LRP does have valid use cases in the real world. The most obvious use case of the

continuous LRP is in greenfield development. In this type of scenario, no existing infrastructure exists, so urban planners can place hubs anywhere in a designated area. Another situation in which the discrete LRP would not be suitable is if the number of potential hub locations is large. In such a case, it would take significant effort to construct a list of potential locations since one would have to evaluate each of them beforehand.

On the other hand, the discrete LRP also has several scenarios in which it is the obvious choice. In cases with already existing infrastructure in place, the discrete LRP will probably be the best choice. For example, Dukkanci and Kara (2017) focus on the airline industry. In this case, it makes sense to build hubs close to airports.

In the discrete LRP, there are several ways to define a list of potential hub locations. One of the most popular methods in the literature is to use the set or subset of customer locations. For example, Ferreira and de Queiroz (2018), Hemmelmayr (2015) and Wang et al. (2018) use this method. Another way is to define a set of potential locations, as demonstrated by Fazayeli et al. (2018), Hiassat et al. (2017) and Yu and Lin (2015). These locations could be hand-picked or generated according to a specified distribution. In the case of a dense, uniformly generated grid, the solution will approximate that of a continuous solution.

One of the critical parts of modeling the LRP is to choose which features to include and which solution methods to use. Choosing either the continuous or discrete method is one of those modeling decisions that could significantly impact the results. It is essential to select the best approach. Therefore, determining when one should use the continuous or discrete method is the focal point of this thesis.

1.5 Solving the LRP

Several different solution methods exist to solve the LRP. One can solve the problem to guaranteed optimality using an exact solution method, but this limits the model to small instance sizes, as solving such a problem *exactly*, requires significant computational power. Therefore, the often preferred method in the literature is a meta-heuristic method (Mara et al., 2021). Sörensen and Glover (2013) define a meta-heuristic method as *"A high-level problem-independent algorithmic that provides a set of guidelines or strategies to develop heuristic optimization algorithms."* Meta-heuristics cannot guarantee optimality but can provide adequate solutions if implemented correctly.

Meta-heuristics are the most popular type of solution method but not the only one. A more recent branch of research is in the direction of machine learning algorithms. Flach (2012) provides the following definition: *"Machine learning is the systematic study of algorithms and systems that improve their knowledge or performance with experience."* In practice, machine learning algorithms are not competitive with meta-heuristics. However, they can provide satisfactory results if used in conjunction with a meta-heuristic to form a so-called "hybrid" strategy. Chapter 4 provides a more in-depth description of various solution methods and their use cases.

1.6 Research Contribution

In the LRP literature, the choice of a continuous or discrete model is often made arbitrarily. Generally, the choice is not debated or even mentioned in the problem definition, as shown by Asefi et al. (2019), Martins de Sá et al. (2018) and Rabbani et al. (2019). This is curious as the choice does have a significant influence on real-world applicability, as well as restricting the set of potential solution methods. On top of that, in discrete models, the set of potential locations is often the same as the set of customer locations. This makes sense in some specific scenarios, such as airplane routes, but not in most situations. An alternative would be to construct a set of pre-determined potential locations, but this poses another problem. Without any restriction criteria, it could take considerable effort to evaluate every possible location in a particular area

manually. In these cases, it would be more efficient to determine the optimal location first, to then search in a limited region for candidate locations. However, this changes the requirements for the model, limiting the set of possible solution methods. Since no clear guidelines exist in which cases to use either a continuous or discrete method, this research aims to study the appropriate use cases of the continuous and discrete methods.

In this thesis, we aim to determine on a per-instance basis if the continuous or discrete approach of the LRP is best suited.

The main contribution of this research is a comparative analysis of the discrete and continuous LRP. Comparative analysis helps to understand the relations between variables and results in a specific situation. By understanding how instance characteristics influence the relative performance of the two versions of the LRP, we can choose which is best suited to a specific situation. Of course, as discussed in Section 1.4, there are many reasons to choose either the continuous or discrete method beyond their performance in finding a solution. Hence, the objective is not to appoint a definitive winner but to provide insight and understanding into which situations the discrete or continuous LRP is expected to perform better.

More specifically, this thesis presents a detailed comparative analysis of the influence of instance characteristics on the performance of the LRP in continuous and discrete space. We compare three different depot placement strategies, of which one is in the continuous space and one in the discrete space. The third method is a method inspired by Manzour-Al-Ajdad et al. (2012) that sits in between the continuous and discrete methods. This third method provides additional insights and shows how alternative methods could be viable compared to conventional ones.

The following chapter provides a detailed problem definition that delivers the main objective and the boundaries of the problem. Chapter 3 gives additional information on the problem and discusses potential solution methods. Chapter 4 explains the process of solving the problem and which methods we use to answer the research question. Chapter 5 provides an overview of the data sets that we use in this thesis. Finally, Chapter 6 shows the results of the experiments.

Chapter 2

Problem Statement

This chapter discusses the typical circumstances in which the LRP is useful. First, an example case shows how the problem in question applies to a hypothetical real-world scenario. In this case, a supermarket chain wants to optimize its distribution network. The second part of this chapter provides the mathematical formulation of the problem.

2.1 Example Case

Suppose we have a supermarket chain operating in a certain region. The company owns multiple retail stores in which they serve customers. Most are located in high-density areas to distribute the number of customers for each store. The stores are supplied by delivery trucks that leave from and return to a single depot every day. The company has been operating for many years and has opened numerous new stores in that time. As a result of their increased throughput, the company has decided to upgrade its depot to handle the larger quantities of supplies. Since this requires a complete reconstruction of the depot, the company wants to reevaluate its decision on its location, bearing in mind the new stores they have opened.

The company aims to find the best location for the depot to minimize costs. The operating costs consist mainly of the trucks' fuel prizes and the truck drivers' wages. Other costs include the maintenance and depreciation costs of the trucks. The capacity of the trucks is high enough to supply multiple, but not all, stores in one trip, reducing the need to refill at the depot. The company must consider that the stores must be supplied within a certain time frame. The stores prefer to receive the new supplies during the least busy hours to distribute the workload on employees. Since this time frame is slightly different for each store, the company has asked each store to provide a time window in which they want to be resupplied.

In summary, this company's problem is classified as a Location Routing Problem. The objective is to find the cost-optimal location of a new depot, considering the daily resupply routes of delivery vehicles. The company is considering two alternatives to find a suitable location for the new depot. They can research the region to create a list of potential locations. Creating this list takes resources, and since the region covers a huge land area, potential locations might be missed. A benefit of this method is that all potential locations are confirmed to be viable. The company only has to find the best location from this list; this is classified as the discrete method. The alternative is to first find the optimal location for the depot without any constrictions in terms of location. Since this optimal location is probably not viable, the company will have to look for the closest viable location, which saves a lot of time compared to creating a list that covers the whole region.

The company must decide which method to use before investing many resources into it. This

company's benefit is knowing exactly where its trucks must go and when. They want to use this information to help decide which method is best suited for them.

2.2 Mathematical Formulation

This section proposes the precise formulation of the problem demonstrated in Section 2.1. The store locations are referred to as *customer nodes* and the depot location as *depot node*. The total set of nodes is V, with V_0 as the depot node, and N is the set of nodes excluding the depot node. Each customer node has a certain demand d_i that must be delivered between time a_i and b_i with $i \in V$. Demand is delivered to customer nodes by vehicle $m \in M$. We assume the fleet of vehicles is homogeneous, and therefore, the capacity of every vehicle C is the same. Each customer node is assigned to a route π_m, driven by vehicle m. Each route always starts and ends at the depot node, V_0. The cost of driving from node i to node j, $i, j \in V$, is defined as $c_{i,j}$. These costs account for fuel and wage costs. Each customer i requires a certain amount of time to service s_i. w_{im} denotes the start time of service at customer i with vehicle m

Equation 2.1 provides the objective function. The binary decision variable $x_{i,j,m}$ indicates if vehicle m drives arc (i, j). Decision variables lx and ly determine the depot locations' horizontal and vertical position, respectively. In the discrete variant, these variables must be chosen from a set Lx and Ly, while in the continuous variant, they can be any real number.

The following model is adapted from Toth and Vigo (2002). The notation "Δ^-" and "Δ^+" mean incoming and outgoing arcs respectively.

$$\min \quad \sum_{i,j \in V} c_{i,j} \cdot x_{i,j,m} \tag{2.1}$$

S.T.

$$\sum_{m \in M} \sum_{j \in \Delta^+(i)} x_{ijm} = 1 \qquad \forall i \in N \tag{2.2}$$

$$\sum_{j \in \Delta^+(0)} x_{0jm} = 1 \qquad \forall m \in M \tag{2.3}$$

$$\sum_{i \in \Delta(j)} x_{ijm} - \sum_{i \in \Delta^+(j)} x_{jim} = 0 \qquad \forall m \in M, j \in N \tag{2.4}$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,m} = 1 \qquad \forall m \in M \tag{2.5}$$

$$w_{im} + s_i + t_{ij} - w_{jm} \leq (1 - x_{ijm})\, M_{ij} \qquad \forall m \in M, (i,j) \in N \tag{2.6}$$

$$a_i \sum_{j \in \Delta^+(i)} x_{ijm} \leq w_{im} \leq b_i \sum_{j \in \Delta^-(i)} x_{ijm} \qquad \forall m \in M, i \in N \tag{2.7}$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijm} \leq C \qquad \forall m \in M \tag{2.8}$$

$$x_{ijm} \in \{0, 1\} \qquad \forall m \in M, (i,j) \in V \tag{2.9}$$

$$lx, ly \in \mathbb{R} \quad \vee \tag{2.10}$$

$$lx, ly \in Lx, Ly \tag{2.11}$$

Equation 2.2 restricts each customer to exactly one route. Equations 2.3, 2.4 and 2.5, define the path that a vehicle is allowed to take. Equations 2.6, 2.7 and 2.8 impose time window and capacity constraints. Finally, Equation 2.9 imposes a binary condition on the decision variable x and Equations 2.10 and 2.11 impose restrictions for the depot location in the continuous and discrete space, respectively. Only one of these constraints can be active.

Chapter 3

Background

This chapter provides background knowledge on the two most important components of the LRP, the Vehicle Routing Problem and the Facility Location Problem. We also discuss potential solution methods for the LRP, as well as a concise literary overview of the origins and current research trends in the LRP.

3.1 VRP

The VRP consists of a set of customer nodes (vertices) and a depot node. The objective is to find the shortest possible route that passes all customer nodes and starts and ends at the depot node. A route is a set of arcs connecting multiple vertices into a closed loop. The VRP knows many variants that increase the realism of the model. We discuss the two most popular variants below (Braekers et al., 2016).

Capacitated VRP
One of the most popular types of VRP is the capacitated VRP. In this case, the vehicle has a maximum capacity, and each customer has a specific demand that must be satisfied. The total demand of a route cannot exceed the vehicle's capacity. In case it does, the vehicle must restock at the depot, resulting in multiple routes. This constraint is necessary for scenarios in which the demand of customers takes up a significant volume or weight.

Time Windows
In many cases, customers are only available in a particular time window. Such a scenario would call for a VRP with time windows. Each customer has a specific time window in which they are allowed to be visited. Therefore, one must construct the route such that each customer is visited only in the specified time window. Since time windows can overlap, a solution with only one vehicle might not be feasible. One option to resolve this would be to relax the time window constraint by introducing a penalty for visiting outside the time window.

Depot Location
The second part of the LRP consists of placing the depot such that the routing costs are minimal. If the routes exist already, only a route's first and last stops influence the depot placement. Again, the objective is to find a location for the depot such that the total distance to each customer connected to the depot is minimal.

Discrete
For the discrete version of the LRP, limited potential locations for the depot exist. A meta-heuristic search algorithm such as tabu-search or simulated annealing is particularly suited for such a task. The iterative algorithms evaluate neighboring solutions and move toward a solution with the greatest obtainable improvement. Both aforementioned algorithms have mechanisms in

place to prevent convergence on a local optimum. Chapter 4 studies these algorithms in more detail.

3.1.1 Continuous

In the case of the continuous LRP, infinite potential locations for the depot exist. This means that the point with the least total distance to all customer nodes directly connected to the depot is optimal. As mentioned in Chapter 1, this problem is known as the Weber problem and is solved using the iterative Weiszfeld's algorithm. Weiszfeld's algorithm is relatively simple and cannot be tuned in any way except for the number of iterations. A higher number of iterations will result in a more accurate location, with the cost of increased computing power.

3.2 Solving the LRP

Numerous solution methods exist, each with its advantages and disadvantages. On a high level, four categories exist: decomposition methods, search algorithms, population-based algorithms, and machine learning algorithms.

3.2.1 Potential Solution Methods

Decomposition methods serve to optimize the computational load of using an exact algorithm. As the name suggests, the problem is structurally decomposed into a main problem and a sub-problem. A very influential decomposition technique is Benders decomposition (Benders, 1962). Benders decomposition is a technique used to solve extensive linear programming models. This decomposition method divides the variables into two sub-sets. The first-stage problem is solved using the first subset of variables, while the values of the second-stage variables rely on the values of the first subset of variables. Benders decomposition requires that the variables of the second stage are continuous. If the second-stage problem generates an infeasible solution, the method introduces a constraint called a "cut" to the first-stage problem. The first-stage problem will then be resolved until the generated solution is feasible. If the solution is feasible, an optimality cut will is introduced. These optimality cuts will change the lower and upper bound for the optimal solution. Both cuts are called "Benders cuts". The procedure terminates when the solution does not have a finite optimal solution or if the gap between the lower and upper bound is deemed sufficiently small.

Search algorithms are iterative heuristic solution methods that researchers have applied to optimization problems for quite some time. The algorithms rely on continuous improvement of the solution by evaluating nearby solutions. One disadvantage of heuristics is that they get stuck in local optima. Several different methods exist to circumvent this problem. The meta-heuristics tabu-search (Tuzun & Burke, 1999) and simulated annealing (Yu & Lin, 2015) have built-in mechanics to get out of local optima. These methods are both suited for the discrete problem. Steepest descent is a heuristic suited for the continuous case, but it requires the problem to be differentiable. On top of that, the algorithm is prone to get stuck in local optima.

Population-based algorithms rely on optimization strategies found in nature. In these algorithms, randomly generated solutions form a population. For example, genetic algorithms use the survival of the fittest strategy, where only the best-performing solutions survive in each iteration. Another often-used algorithm is particle swarm optimization. Here the population behaves like a swarm, moving toward the best-known solution. Many papers combine these population-based algorithms with other methods (Schwardt & Dethloff, 2005).

Lastly, machine learning algorithms are a relatively new type of solution method. For example, Schwardt and Fischer (2009) use a self-organizing map, a form of a neural network, to optimize a continuous LRP. Although they get a feasible result, the performance, in combination with the high set-up effort, is not on par with other solution methods.

3.3 Variable Neighborhood Search

VNS is a meta-heuristic optimization algorithm that is used to find near-optimal solutions to complex optimization problems. It is particularly useful for problems that have many local optima, as it is able to escape from these local optima and find better solutions. Mladenovic et al., 2021 describe the VNS as: "The main idea of VNS is to perform systematic changes in the neighborhood structures in order to avoid stagnating the search in attraction basins (local optima)." Mladenović and Hansen (1997) are the first to propose VNS as a meta-heuristic algorithm to solve the LRP. Currently, many variants of VNS exist, such as Reduced VNS (Yu, Maglasang et al., 2020), General VNS (Soylu, 2015), and Variable Neighborhood Descent (Mjirda et al., 2017). Hansen et al. (2017) provide an exhaustive overview of the VNS algorithm and its variants.

Sadati and Çatay (2021) build upon the basic VNS by combining it with Tabu Search, which further improves the performance. Baniamerian et al. (2019) combine VNS with a genetic algorithm, which helps the algorithm converge faster in large-scale instances. Alternatively, VNS can also be used to aid existing solution methods, as Davoodi and Goli (2019) demonstrate by using VNS to solve the main problem in Benders decomposition method. Karakostas et al. (2022) develop a VNS to solve the multi-objective LRP, in which the secondary objective is to minimize pollution. An interesting addition that the authors apply to their algorithm is an adaptive element. This adaptive element changes the order of neighborhoods based on previous successful iterations. For example, if the algorithm normally traverses from neighborhoods 1 to 2 to 3 but finds more improving moves in neighborhood 2 than 1, it will prioritize neighborhood 2 over 1. This technique will allow the algorithm to find improving moves in fewer iterations, thus making it more efficient. The reason that the order of neighborhoods cannot be determined beforehand is that different instances benefit certain neighborhoods more than others.

3.3.1 Algorithmic Approach

In VNS, one attempts to escape local optima by changing the "neighborhood" of potential solutions. A neighborhood defines the set of possible operations that the algorithm may apply to a function $f(x)$ in an effort to improve the solution.

In VNS, the objective is to minimize or maximize a function f defined on a solution space X. In this case, the objective is to minimize the costs of routing. Provided an initial solution, the algorithm starts by perturbing solution x to obtain a slightly different solution x'. The neighborhood structure $\mathcal{N}_k$, with $k = 1, ..., k_{max}$, defines the potential set of modifications. In the next step, the algorithm performs a local search on x' to find x". If $f(x") < f(x)$, then the solution will be accepted as the new current solution $f(x) = f(x")$ and neighborhood $\mathcal{N}_k(x)$ is set to $\mathcal{N}_1(x)$. If $f(x") \geq f(x)$, than the neighborhood $\mathcal{N}_k(x)$ changes to $\mathcal{N}_{k+1}(x)$. The algorithm continues until all neighborhood structures are explored so that the solution is the local minimum with respect to all neighborhoods. Figure 3.1 visualizes the process.

Neighborhoods
Developing a VNS algorithm requires making several important decisions regarding the neighborhood structures. First, which neighborhoods to use and how many is one of the most critical decisions in developing the algorithm. Adding more neighborhoods might result in better solutions but at the cost of increased computation time. The order in which to traverse neighborhoods is, in general, increasingly distant, meaning that each following neighborhood $\mathcal{N}_{k+1}(x)$ explores more distant solutions compared to neighborhood $\mathcal{N}_k(x)$.

Solution Evaluation
Evaluating a specific solution is another important decision that influences the performance of the VNS. A solution is generally evaluated using a fitness function $f(x)$. A fitness function can be as simple as the total distance and a feasibility check. However, this method allows only feasible solutions, which can severely limit the algorithms' ability to escape local optima. One can include a penalty component in the fitness function to solve this problem. The severity of the

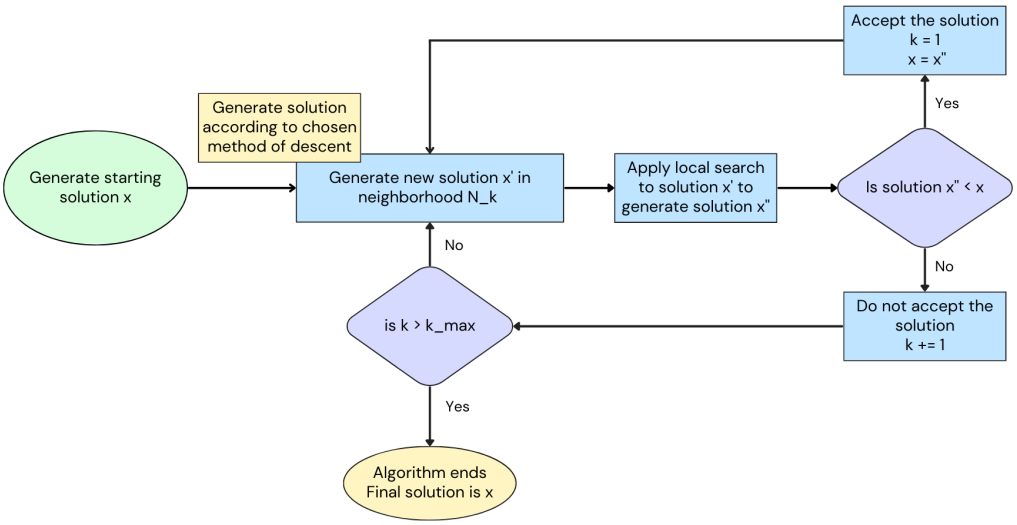


Figure 3.1: Flowchart Variable Neighborhood Search

penalty should be proportional to the degree of infeasibility of a particular solution. Allowing the algorithm to accept temporary infeasible solutions gives it a higher mobility and, thus, a better chance of escaping local optima. For example, Equation 3.1 gives a fitness function for a model with capacity constraints and time windows where α, β, γ are constants and $C(x)$ is the associated cost of component x. By altering the constants, the focus of the minimization will shift with respect to the ratio of those constants.

$$f(x) = \alpha \cdot C(violated\ time) + \beta \cdot C(violated\ capacity) + \gamma \cdot C(distance) \tag{3.1}$$

Chapter 4

Methodology

This chapter explains the methodology that we use to solve the LRP. We have three different methods to determine the depot location that we want to compare, continuous, discrete, and an in-between method. Since the methods only differ in determining the depot location, we can keep the routing part equal. Section 4.1.4 explains the three methods for choosing a depot location.

4.1 Variable Neighborhood Search

This section discusses the implementation of the VNS algorithm. The algorithm contains several parts we will discuss individually; the neighborhoods, starting solution, local search, evaluation, and termination.

4.1.1 Method of Descent

We consider two methods of descent, First Descent, and Steepest Descent. In the First Descent method, the first generated solution x' for which $f(x') < f(x) \quad \forall \mathcal{N}_k(x) \subseteq X$ holds is accepted. Alternatively, in Steepest Descent, all possible solutions are evaluated, and the best solution is chosen so that $x' = min(f(x)) \quad \forall \mathcal{N}_k(x) \subseteq X$. While Steepest Descent generally leads the search to an optimum in a shorter number of steps than First Descent, each step will take longer to compute, depending on the number of neighboring solutions to evaluate.

Preliminary testing shows that First Descent and Steepest Descent provide solutions with similar quality, but First Descent does so using substantially less computation time. Appendix B provides a numerical comparison.

4.1.2 Solution Representation

A complete solution x contains every individual route $\pi_{m,i}$ where $m = 1, ..., M$ defines the set of vehicles and $i = 0, ..., I$ defines the set of customer nodes within route m. π_m is an ordered sequence of nodes that denote the route that vehicle m takes. The route always starts and ends at the depot node; 0. The following Equation 4.1 is an example representing the model given in Figure 4.1.

$$x = \begin{Bmatrix} \pi_1 = [0, 2, 1, 5, 8, 6, 0] \\ \pi_2 = [0, 10, 12, 13, 11, 0] \\ \pi_3 = [0, 9, 7, 4, 3, 0] \end{Bmatrix} \tag{4.1}$$

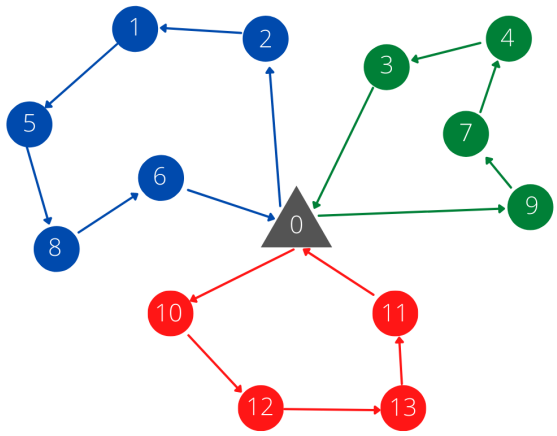


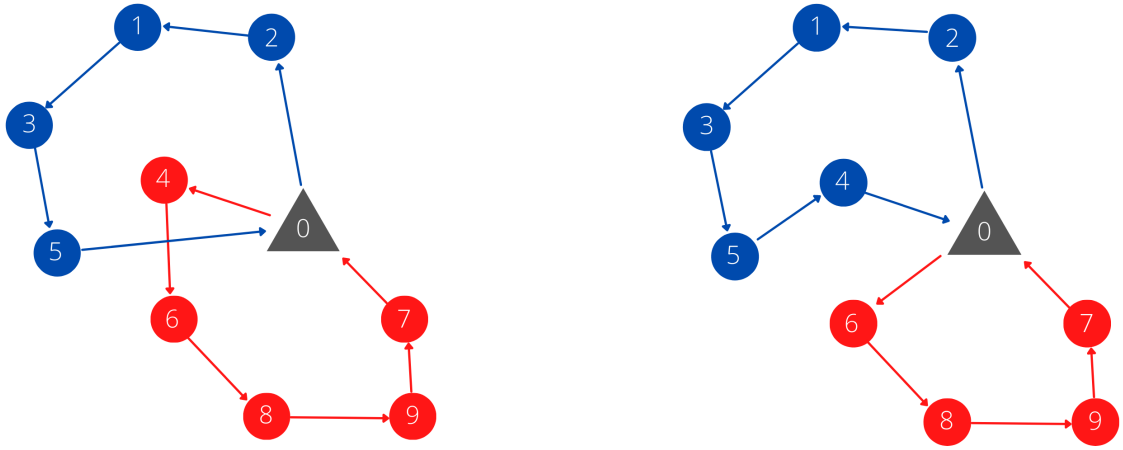
Figure 4.1: Example Solution Structure

4.1.3 Neighborhood Structures

As discussed earlier, the neighborhood structures describe the set of potential operations that may be applied to solution x to find a neighboring solution x'. These neighborhood structures should gradually reach further out to find increasingly more distant potential solutions. We consider six neighborhoods $\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3, \mathcal{N}_4, \mathcal{N}_5, \mathcal{N}_6$.

$\mathcal{N}_1$ Insertion

The first neighborhood structure is called *insertion*. Insertion is a straightforward operator that removes customer node i from route π_m and inserts it before customer node j in route π_n where $m \neq n$. To evaluate all possible transformations, we must insert each node before every other node in the model, except for the ones in the same route. This leads to a growth rate of $O(N^2)$, where N is the total number of customers. This could be a limiting factor in trying to solve larger instances. However, as discussed earlier, the First Descent method allows us to drastically reduce computation time while surrendering little to no solution quality. Figure 4.2 depicts an example of the insertion operation.



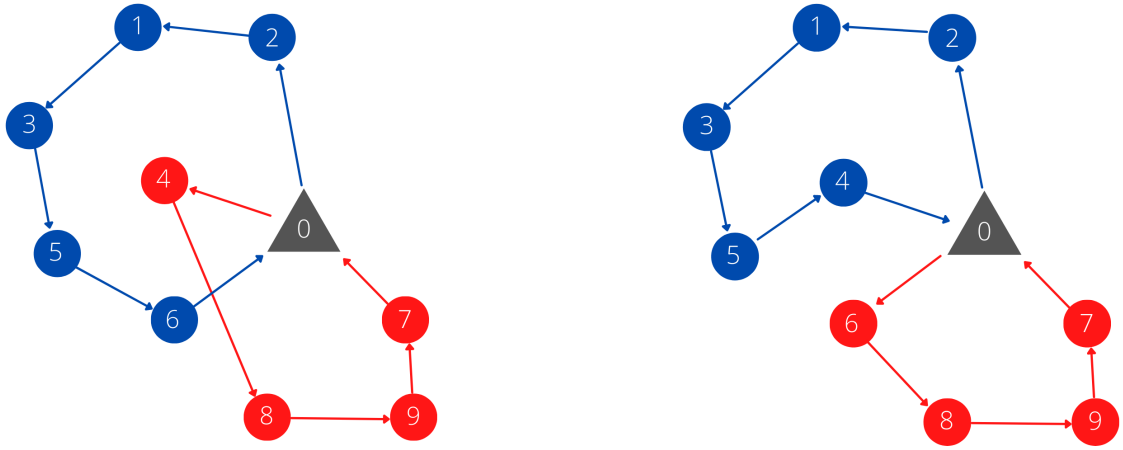
(a) Before Insertion (b) After Insertion

Figure 4.2: Example Insertion Method

$\mathcal{N}_2$ Swap

Neighborhood $\mathcal{N}_2$ consists of the *swap* operation. The swap operation swaps the position of two customer nodes in different routes, as visualized by Figure 4.3 formally written as:

$$\begin{aligned} \pi_m &= [0, ..., i, ..., 0] \rightarrow \pi_m = [0, ..., j, ..., 0] \\ \pi_n &= [0, ..., j, ..., 0] \rightarrow \pi_n = [0, ..., i, ..., 0] \end{aligned} \tag{4.2}$$



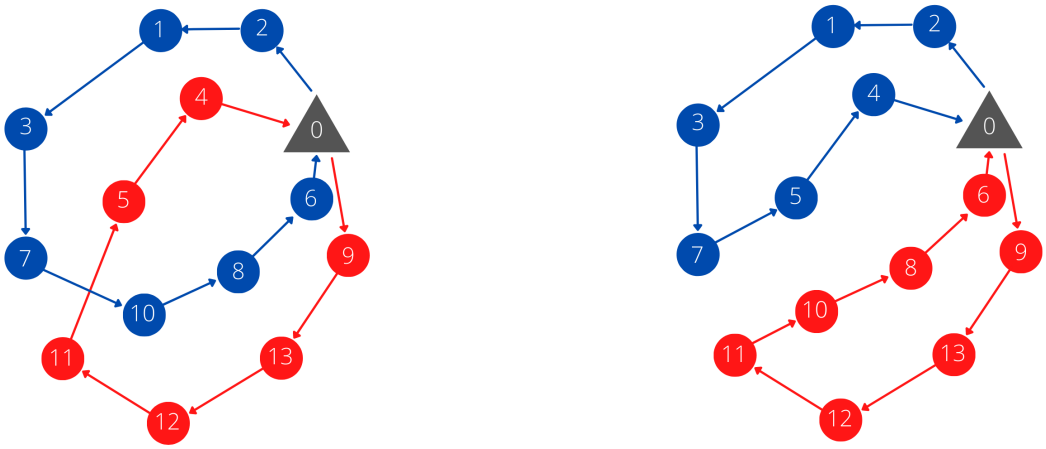
(a) Before Swap (b) After Swap

Figure 4.3: Example Swap Method

$\mathcal{N}_3$ 2-opt

The 2-opt algorithm is first introduced by Croes (1958) as an efficient process for solving the traveling salesman problem (TSP). The 2-opt algorithm permutes the route in such a way that it eliminates crossovers. While for the TSP eliminating a crossover is always beneficial, that is not always the case for the VRP or LRP due to time window constrictions. Nevertheless, the

algorithm can still provide improvements that the insertion or swap methods cannot. In intra-route permutations, the method works by swapping two nodes and reversing the series of nodes between them. In inter-route permutations, the algorithm selects two nodes and swaps the nodes that come after the selected nodes (Tavares et al., 2009), as depicted in Figure 4.4.



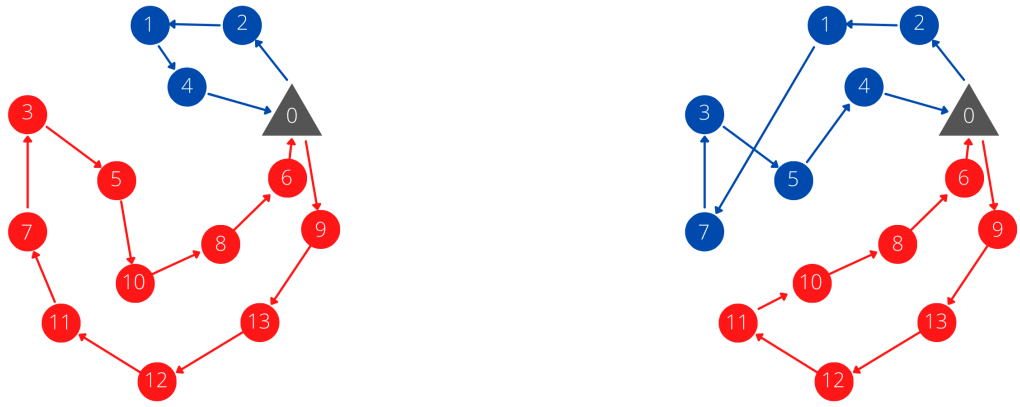
(a) Before 2-opt

(b) After 2-opt

Figure 4.4: Example 2-opt Method

$\mathcal{N}_4$ Extended Insertion

Much like *insertion*, in extended insertion, a part of one route is inserted into another route. However, it is possible to insert multiple nodes simultaneously in extended insertion. The minimum number of nodes to be inserted in another route is one, and the maximum is limited to the length of the route. The number of available permutations using this method is in the order of $O(N^3)$. Since this causes the algorithm to suffer critical computational limits, we need to lower the number of available options. Only allowing the chain of nodes to be inserted before the geometrically closest node in a different route brings the order of possible permutations down to $O(N^2)$. Figure 4.5 shows how section $[7, 3, 5]$ is inserted in another route. Since node 4 is geometrically closest to node 7, the section is inserted before node 4. This modification results in a somewhat less-than-optimal configuration temporarily, but this will be easily fixed in the local search part.



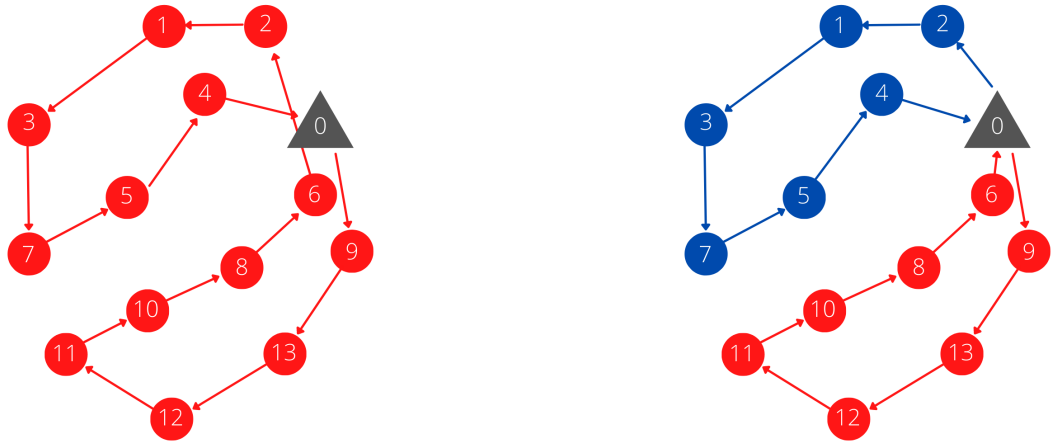
(a) Before Extended Insertion

(b) After Extended Insertion

Figure 4.5: Example Extended Insertion Method

$\mathcal{N}_5$ Split and Merge

In this neighborhood structure, a route is split in two, or two routes are merged. The algorithm evaluates all possible splits for each route that contains more than one node. Secondly, all possible merge combinations are evaluated as well. The algorithm chooses the operation that results in the best fitness. Figure 4.6 provides an example of a split operation in which the route is split after node 6. The merge operation is essentially a reverse split operation.



(a) Before Split (b) After Split

Figure 4.6: Example Split and Merge

4.1.4 Depot Location

The final neighborhood structure $\mathcal{N}_6$ determines the depot's location. We compare two conventional methods for the continuous and discrete LRP and a method that attempts to bridge the gap between the two techniques.

Weiszfeld's Algorithm for the Continuous LRP

Weiszfeld's algorithm is an iterative technique to calculate the geometric median of a set of points (Eftelioglu, 2015). The geometric median defines a point m in Euclidean space with the lowest sum of distances to a given set of points $a_1, a_2, a_3, ..., a_n \in A$. Formally, the geometric median is defined as:

$$m(A) = arg \ \ min \sum_{i=1}^{n} ||m - a_i||$$

In the LRP, m is the depot, and the set of points A consists of all the connection nodes, defined as the first and last node of every route. Applying Weiszfeld's algorithm to the LRP ensures that the depot is placed such that the total distance is minimized, given a set of routes. However, this does not always guarantee that the depot is placed optimally, as this method, for example, does not consider time windows. Figure 4.7 shows the final solution and how the continuous method locates depots each iteration. The under-left most node is the starting location and star-shaped markers indicate a chosen solution. The instance is from the random data set, with 50 customers.

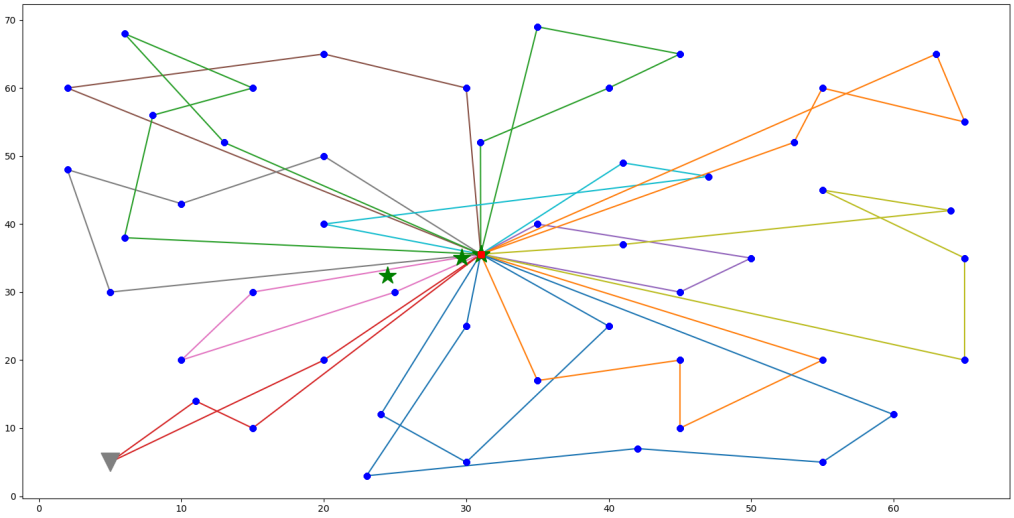


Figure 4.7: Example Continuous Method, Solution Cost = 1084.7

Customer Based Location for the Discrete LRP

A popular method in the literature is to place the depot at the same location as one of the customer nodes. Therefore, the set of potential depot locations is identical to that of customer locations. This relatively simple method is particularly useful in some specific real-world scenarios. One obvious disadvantage of this method is that the potential set of depot locations is very limited, especially in instances with a relatively low number of customer locations. An advantage is that it is possible to choose a location according to a fitness function and not just based on distance. Figure 4.8 shows how the discrete method behaves. Again, the starting location for the depot is the under-left most node. Triangle-shaped markers indicate an evaluated location, and star-shaped markers indicate a chosen location, within an iteration.

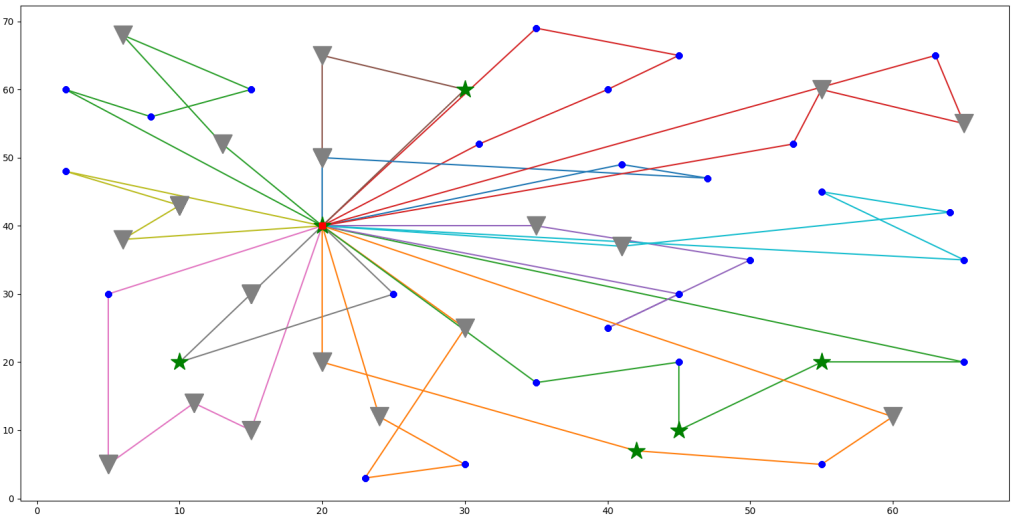


Figure 4.8: Example discrete Method, Solution Cost = 1110.83

Dynamic Grid Structure

This unconventional method attempts to combine the strengths of both aforementioned techniques. It is based on a method from Manzour-Al-Ajdad et al. (2012). The idea is to create a virtual set of potential locations at a set distance around the current depot location. The algorithm evaluates all surrounding points using a fitness function and chooses the best location. If no improvement over the current location is possible, the distance to the surrounding points is reduced, and the process repeats until it meets a termination criterion. The depot location in each individual iteration is fundamentally discrete, but this method still allows a virtually infinite set of potential locations while still evaluating the locations according to a fitness function. Again, 4.9 shows the behavior of the grid method.

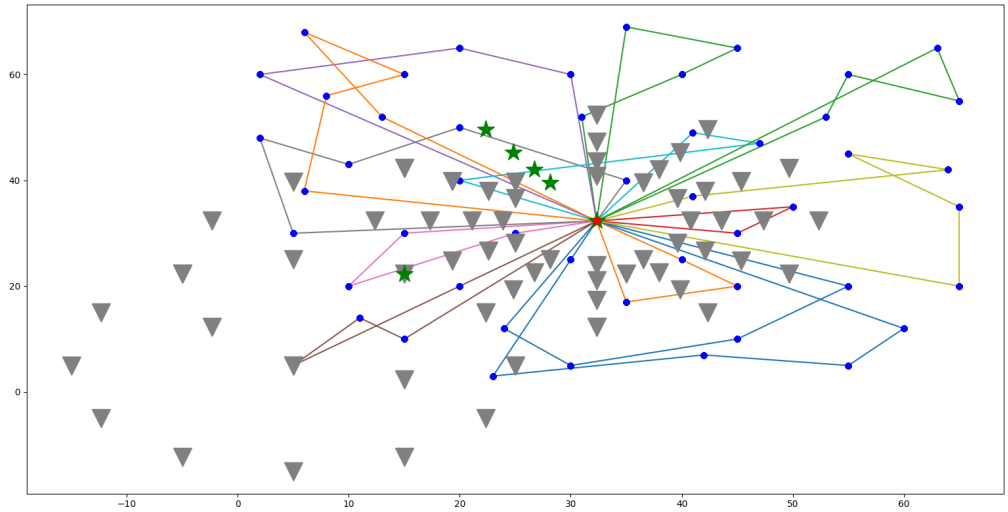


Figure 4.9: Example Dynamic Grid Method, Solution Cost = 1078.67

4.1.5 Local Search

We perform a local search after each permutation in one of the neighborhood structures. The local search focuses on intra-route optimization since the neighborhood structures focus mainly on inter-route optimization. Since we optimize each route individually in this step, the problem is reduced to a TSP with time windows. One of the most commonly used heuristics to solve the TSP is the 2-opt algorithm (Bentley, 1992). 2-opt is designed to eliminate crossovers in routes. However, due to time window constrictions, crossovers are sometimes unavoidable. The insertion method is another candidate that can deliver good results. Section 6.1.1 compares both methods numerically to find the best-suited one.

4.1.6 Starting Solution

The VNS algorithm only changes an already existing solution. Therefore, we must provide the algorithm with a starting solution. We test three different methods for generating a starting solution, a Nearest Neighbor heuristic, a Steepest Descent Savings heuristic, and a First Descent Savings heuristic. Nearest Neighbor starts by adding the closest node to the depot to an empty route. After each node, the next closest node is added to the route on the condition that the route is feasible respective to capacity and time window constraints. If a node added to the route causes the route to be infeasible, it is instead added to a new empty route. This process continues until all nodes are part of a route. The savings heuristics start by assigning all nodes as separate routes. The next step is to find an improvement by combining two routes. Steepest Descent evaluates all possible combinations, choosing the one providing the steepest increase in solution quality.

Alternatively, First Descent chooses the first combination that it finds that provides an increase in solution quality. Section 6.1.2 provides a numerical comparison between the three heuristics.

The starting location of the depot is also part of the initial solution. The discrete method only allows depots directly on customer nodes, so one of the customer nodes will function as a depot location for all methods. We choose 3 customer nodes at random for each test to minimize variability. We use the same random seed for all solution methods to make the most direct comparison.

4.1.7 Feasibility

Since the algorithm uses a fitness function to determine the quality of a certain solution, it is possible to traverse infeasible solutions. This behavior is desired during the optimization phase to better escape local optima but could result in an infeasible final solution. To solve this, the algorithm checks each individual route for feasibility before terminating. Each feasible route is *locked* as well as the depot location, and the algorithm reruns with the infeasible routes and an increased penalty for breaking time windows. This process repeats until all routes are feasible.

4.2 Comparative Analysis

After tuning the algorithm to provide adequate and consistent results, we start with the comparative analysis. In this part, we test each of the three algorithms on their provided quality of solution for a variety of different instances. The objective is to understand how certain instance characteristics influence the relative performance of the continuous and discrete methods. We will test the customer distribution, instance size, widening of time windows, decreasing vehicle capacity, and the combined effect of widening time windows and decreasing vehicle capacity. Lastly, we test how penalties for violating time window restrictions influence both methods in a more realistic scenario.

Chapter 5

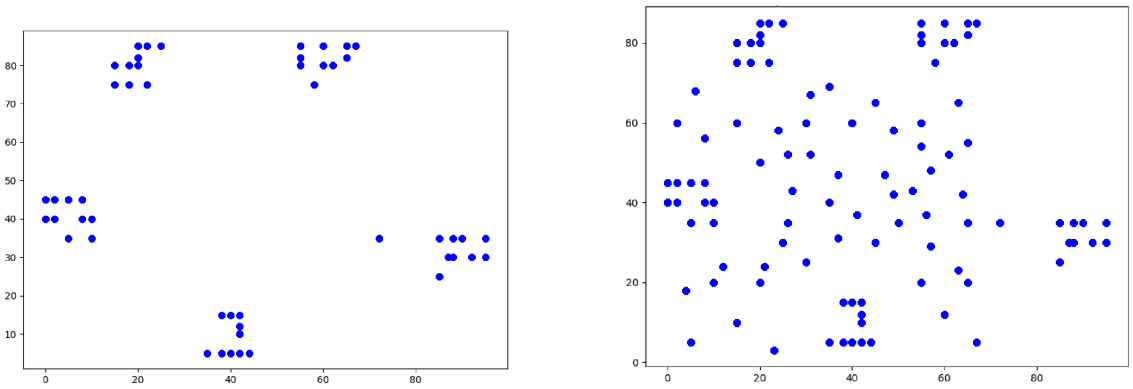
Data

The data fed into the models is essential to the results. We use multiple data sets to test different elements of the models. For example, a uniformly distributed data set can perform differently than a clustered one. Different-sized data sets can also perform relatively differently under varying solution methods. For example, Saifullah Hussin and Stützle (2014) compare two other search heuristics and find that the size of the data set influences the relative performance between the two heuristics.

5.1 Data Sets

Among the most popular data sets used in the routing literature are the Solomon instances (Solomon, 1987). This data set contains a variety of instances with different characteristics. The data set is composed of six different problem types (C1, C2, R1, R2, RC1, RC2). Each set contains between eight and twelve instances with 100 customers. Problems of type 1 have narrower time windows and smaller vehicle capacities compared to problems of type 2. This results in more, shorter routes. The letter of the data set denotes the customer distribution. C (clustered) problems have customers in groups with chosen time windows such that exactly one vehicle can service an entire cluster. R (random) problems have a randomly uniformly distributed set of customers in a square. The time windows are generated according to a function in which the center is based on a randomly distributed number within an interval, while the width of the time window is generated according to a normal distribution. The RC (semi-clustered) problems are a combination of clustered and randomly distributed customers.

An important detail about the semi-clustered data set is that the distribution of customers is unequal between the different sizes. For example, Figure 5.1 compares an instance from the semi-clustered set with 50 customers and one with 100 customers. The instance with 100 customers provides a far more uniformly distributed set of customers, which benefits the discrete method. In the VRP, this difference is largely irrelevant as the depot is given and usually somewhere in the middle. This is one of the limitations of using the Solomon Instances for the LRP and should be considered while interpreting the results. More concretely, we expect the discrete method to perform substantially worse for the semi-clustered set with 50 customers compared to the instance with 100 customers.



(a) 50 Customers (b) 100 Customers

Figure 5.1: Distribution Semi-Clustered Instance

5.2 Modifying the Data Sets

In order to find a better understanding of how certain instance characteristics affect the different solution methods, we need to modify some of the characteristics. The given time windows in the first data sets are quite narrow, meaning that there is only a small window of time in which a customer can be serviced. By gradually widening the time windows in both directions, we can better understand how changing this characteristic influences the performance of the three solution methods. We can do a similar thing for vehicle capacity constraints. The default capacity constraints are pretty lax compared to the time window constraints, meaning that in most cases, the structure of a route is limited by time window constraints as opposed to capacity constraints. This means that we should restrict the capacity constraints further in order to observe any effects on the solutions. We can do this by multiplying the default capacity with a number between zero and one.

Chapter 6

Numerical Experiments

This chapter first discusses the parameter settings of the algorithm, as well as the performance of the isolated VRP part in several different environments. We also examine the robustness of the algorithm to provide a better understanding of the reliability of the final results. In the second part of this chapter, we perform the comparative analysis.

6.1 Parameter Tuning

This section explains how we establish certain parameter settings and choose which algorithms to use in cases with multiple viable options. For the tuning, we only use the VRP part of the algorithm, meaning the depot location is fixed. We do this because the optimal solution for nearly all Solomon instances is known. This allows us to compare the generated solutions with the optimal ones.

6.1.1 Local Search

As Chapter 4 explains, we have two candidate algorithms to perform the local search. The 2-opt algorithm and the insertion method. As part of the VNS, we test both methods on the Solomon semi-clustered data set with 50 customers. This data set contains eight instances with a combination of clustered and randomly distributed customers. Table 6.1 shows the averaged results. The insertion method provides a better result but with an increase in computational time.

	Cost	Computation Time (s)
2-opt	822.4	221.5
Insertion	811.2	269.6

Table 6.1: Local Search Algorithms

6.1.2 Initial Solution

Since the VNS requires a feasible solution to start, it is important to find which method works best. Three possible algorithms are the Nearest Neighbor, First Descent savings, and Steepest Descent Savings heuristics. We test all three heuristics in combination with the VNS algorithm on the random data set with 50 customers. In Table 6.2, we see that the Steepest Descent Savings heuristic provides the highest average quality solutions but with a significantly higher computation time, making the method unviable in most circumstances. Between Nearest Neighbor and the First

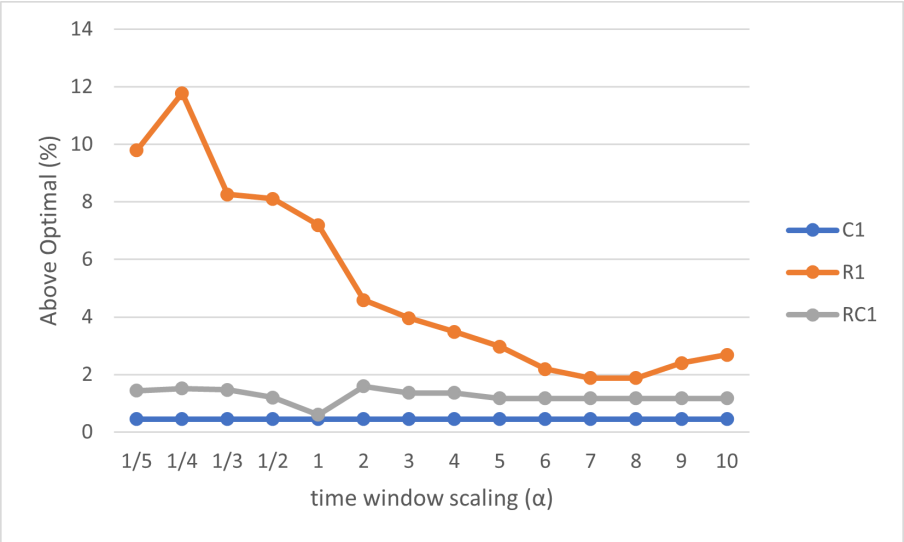
Descent Savings heuristic, Nearest Neighbor is the better choice, with a higher quality solution and less computation time.

	Cost	Computation Time
Nearest Neighbor	825.03	30.04
Savings First Descent	846.08	40.94
Savings Steepest Descent	822.13	190.27 /

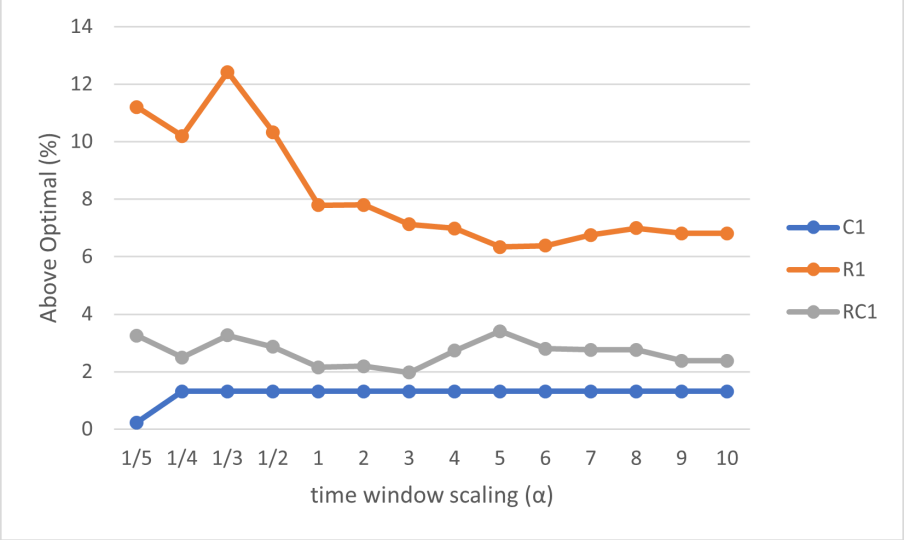
Table 6.2: Starting Solution

6.1.3 Fitness Function

The fitness function consists of three parts; the cost of distance driven, the penalty for capacity violations, and the penalty for time window violations. Each part of the fitness function is multiplied by a constant, which allows the focus to shift between the three different elements. We have the constant α for time windows, β for capacity, and γ for driven distance. To find the best ratio between the constants, we test various options for α on multiple data sets. We test three data sets with 25 and 50 customers. These are the clustered, random and semi-cluster sets of the Solomon instances. We let α run from 0.2 to 10. Figure 6.1 shows the results. First of all, the clustered and semi-clustered data sets show no clear trend, as they are both close to or optimal in all cases. Note that a slight discrepancy will always exist between the results and the optimal values due to different rounding methods. This discrepancy is around 0.5% for the clustered set with 25 customers, as those results are confirmed to be optimal. The random data set does show a trend in both cases. For the random set with 25 customers, the optimal value for α is between 7 and 8. The optimal value for the random set with 50 customers is between 5 and 6. We aim to optimize the algorithm in the most general form. Therefore, α is set to 6.5.



(a) N = 25



(b) N = 50

Figure 6.1: Time Window Penalty Factor

Changing the value of β results in little to no performance change. This is because in these data sets, the time windows are practically always the limiting factor for a route. For this reason, the value of β is set to 1. Since the performance of the fitness function is a result of the ratio between the elements and not the numerical value, γ is also set to 1.

6.1.4 VRP Performance

This section discusses the performance of the VRP part of the VNS algorithm as well as its reliability and robustness. Figure 6.2 provides the optimality gaps of all the data sets. Notable is that the quality of the solutions degrade as the size of the instances increases, except for the clustered set.

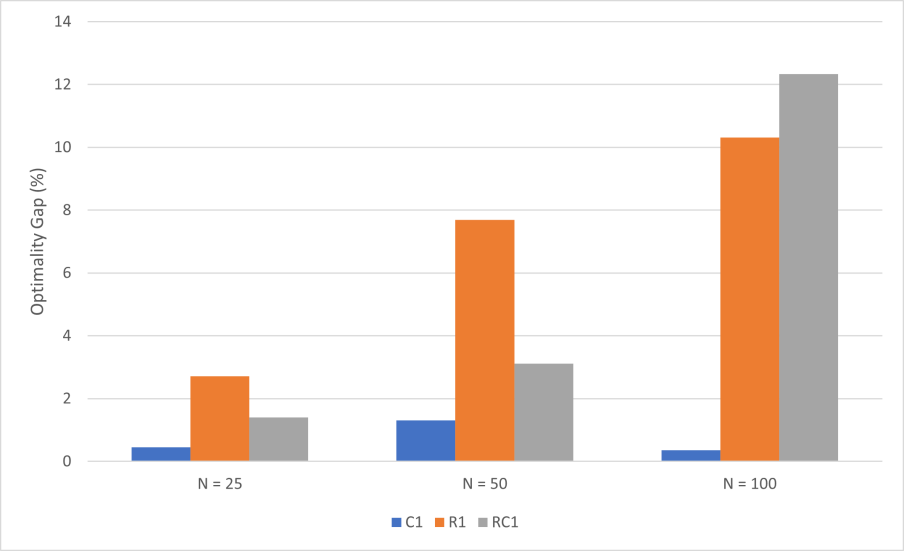
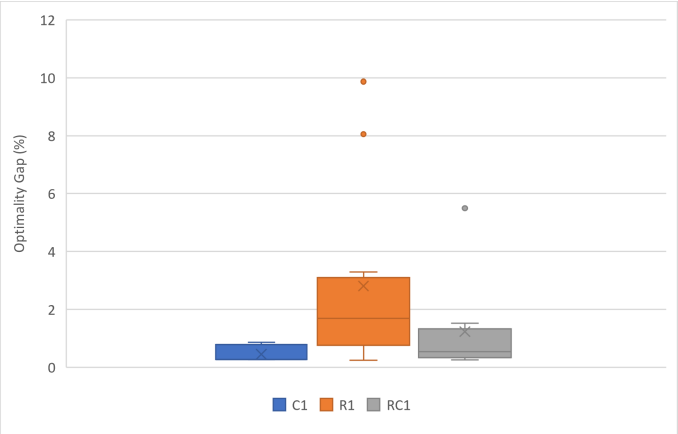
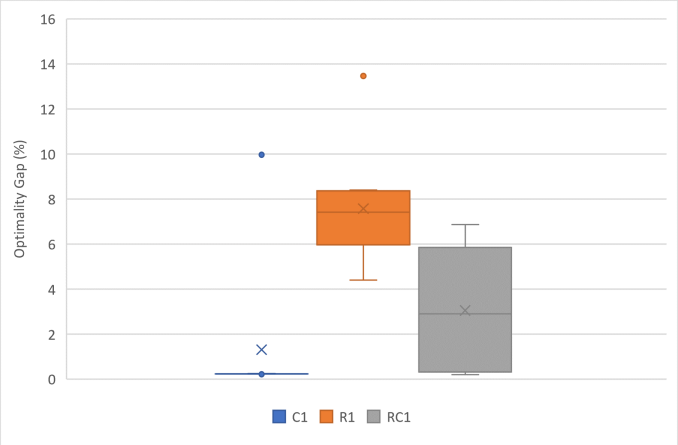


Figure 6.2: Optimality Gaps in the VRP

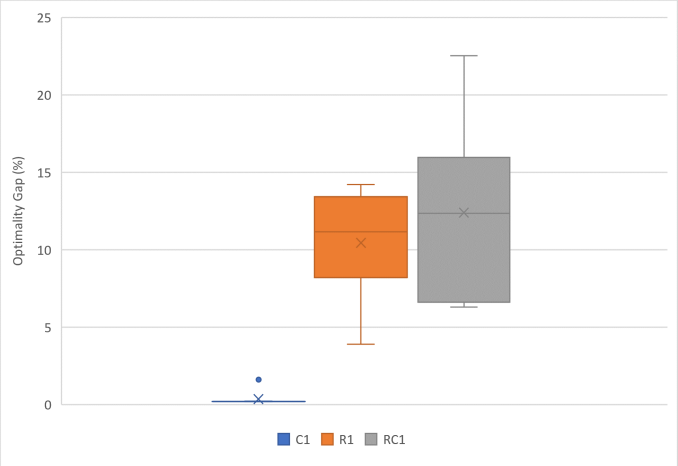
Figure 6.3 shows a more detailed representation of the data in the form of box plots. We see that instance size does not influence the clustered data set, but it does negatively influence the other two data sets. For the random and semi-clustered data sets, we can conclude that the method's reliability decreases as the instance size increases since the range of the boxes increases. Outliers exist in all size groups and, in some cases, skew the data a little.



(a) N = 25



(b) N = 50



(c) N = 100

Figure 6.3: Performance of the VNS algorithm for the VRP

6.1.5 LRP Performance

This section discusses several aspects of the LRP algorithm. First of all, we look at the reliability of the three methods in different circumstances. Much like the importance of the starting solution in the VRP, the starting position of the depot in the LRP is of great importance. The Solomon instances provide a depot location, but since this location is never on a customer node, it is an infeasible starting location for the discrete method. This is why every depot starting location will be on a random customer node, with the same seed for all three methods. Figure 6.4 shows a box plot for each method on the first instance of the semi-clustered set with 50 customers and eleven different starting locations. The continuous method shows a statistically significantly lower mean, as well as a low variance, compared to both other methods. The mean of the discrete and grid method is not significantly different, but the grid method has a couple of outliers in both directions. Interestingly, the grid method produces the best individual solution, showing its potential. However, the high variance is a major drawback.

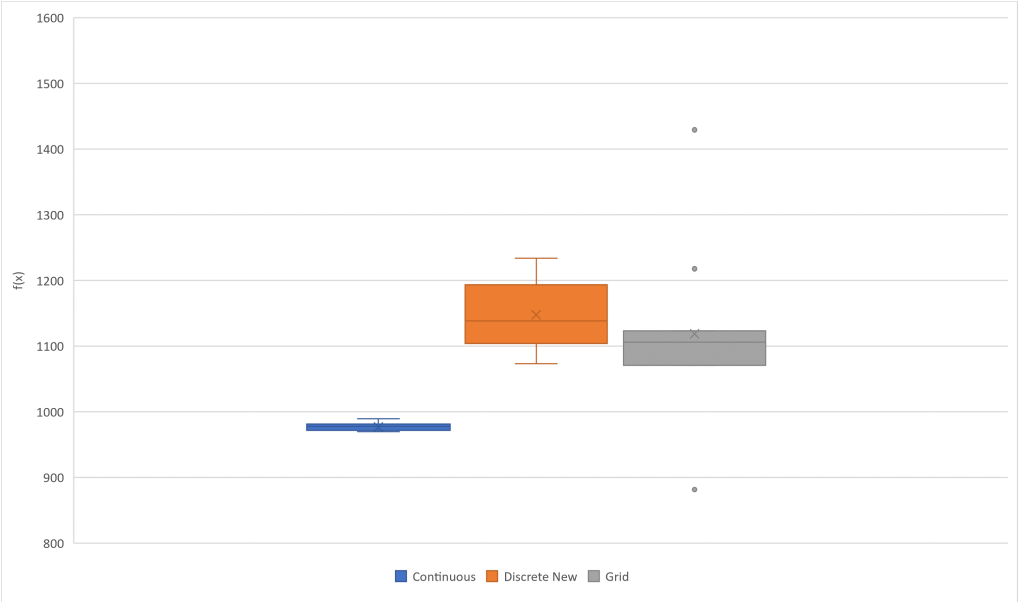


Figure 6.4: Box plot, Semi-Clustered N = 50

Lastly, to increase the reliability of the discrete algorithm, the number of potential depots in each iteration is increased from eight to ten, based on tests, visualized in Figure 6.5. Half of the potential locations are the closest nodes, and the other half are random to escape from local optima. However, the algorithm is still not as reliable as the continuous or circle structure method. The reason for the low reliability is twofold. For example, suppose a scenario with one optimal node and all the other nodes are significantly worse. The first reason why the algorithm cannot find the optimal node is that it does not evaluate the node on pure chance. Increasing the number of potential depots decreases the likeliness of this happening, hence why Figure 6.5 shows a decreasing trend up to a certain point. The second reason is that even if the optimal node is evaluated, it is first evaluated on the current routing solution. This is necessary because calculating the routing for each potential depot location on every iteration is computationally too expensive. As a result, the algorithm sometimes gets stuck in a local optimum.

Table 6.3 shows the standard deviations for each instance in the semi-clustered set from eleven random starting locations. The standard deviation of the discrete method is consistently higher than the standard deviation of the continuous method. This means that the discrete method is less reliable compared to the continuous method within the specified circumstances. The grid method has a slightly higher standard deviation than the discrete method and is the most unreliable.

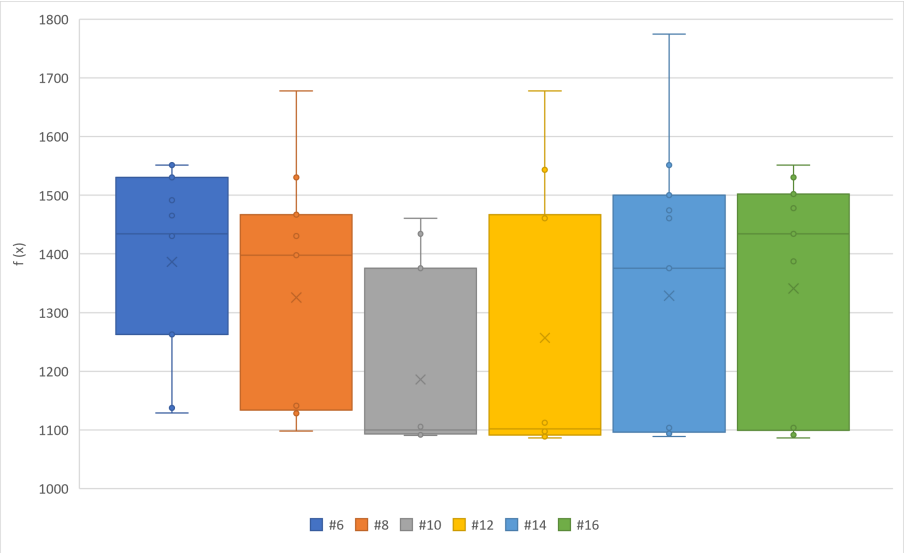


Figure 6.5: Number of Potential Depots in the Discrete Method for Each Iteration

Instance	Continuous	Discrete	Grid
RC101	5.80	194.39	5.35
RC102	26.84	164.54	161.66
RC103	31.19	72.48	91.91
RC104	1.72	87.80	104.43
RC105	3.22	54.67	194.03
RC106	24.39	45.58	95.82
RC107	1.60	92.79	136.73
RC108	3.25	72.98	143.03
Average	12.25	98.15	116.62

Table 6.3: Standard Deviation 11 Random Starting Locations

6.2 Comparative Analysis

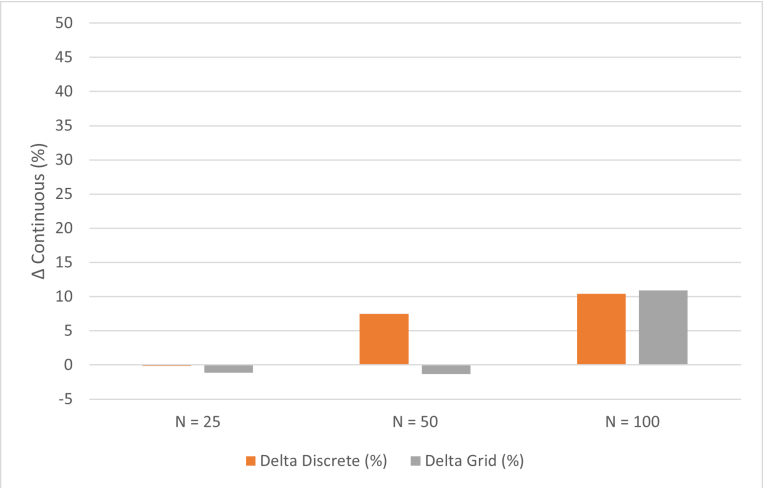
In this section, we perform a comparative analysis. We test how certain instance characteristics influence the relative performance of the three solution methods, the continuous, discrete, and grid method.

6.2.1 Customer Distribution

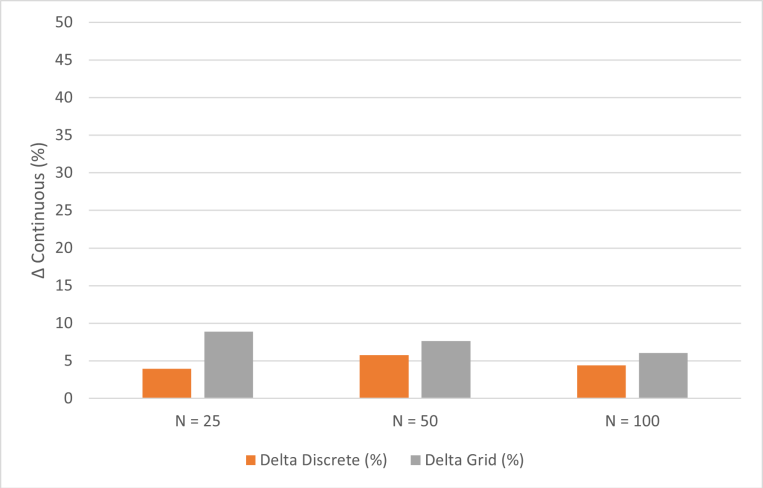
As discussed in Chapter 5, we test three different types of customer distribution; clustered, random, and semi-clustered, in combination with three different instance sizes: 25, 50, and 100. Each instance is tested three times with a randomly chosen customer as the depot starting location. Figure 6.6 shows the mean difference in solution quality between the continuous and discrete methods expressed as a percentage of the continuous method. Each data set contains between eight and twelve instances, in combination with the three randomly chosen depot starting positions resulting in each statistic having between 24 and 36 individual data points. Overall, the continuous method performs slightly better than the discrete method.

The semi-clustered data set with 50 customers shows the largest difference by a wide margin, with the discrete method having over 40% higher costs than the continuous method. This is curious as the difference is less than 10% on the same data set but with 100 customers and less than 5% with 25 customers. This indicates that the data sets do not just differ in size but

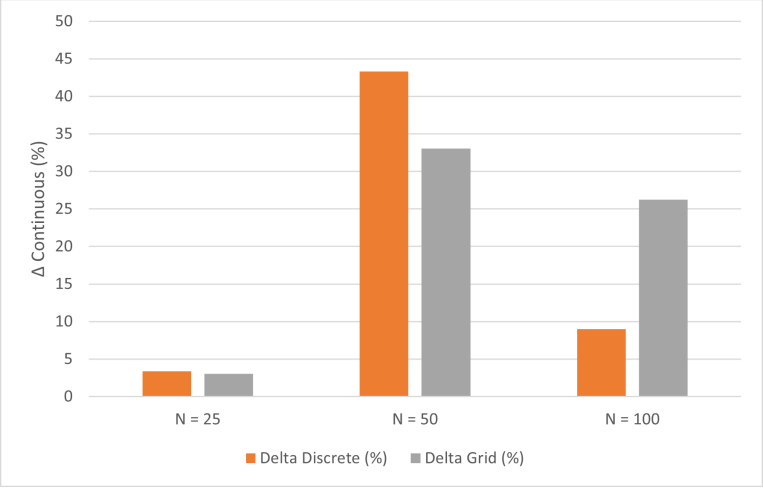
have another fundamental difference contributing to the discrepancy between the continuous and discrete methods. This difference is the distribution of the customers. As explained in Chapter 5, the semi-clustered data set with up to 50 customers has only clusters, while the same data set with up to 100 customers has a mix of clusters and uniformly distributed customers. This is a situation in which the discrete method should be avoided.



(a) Clustered



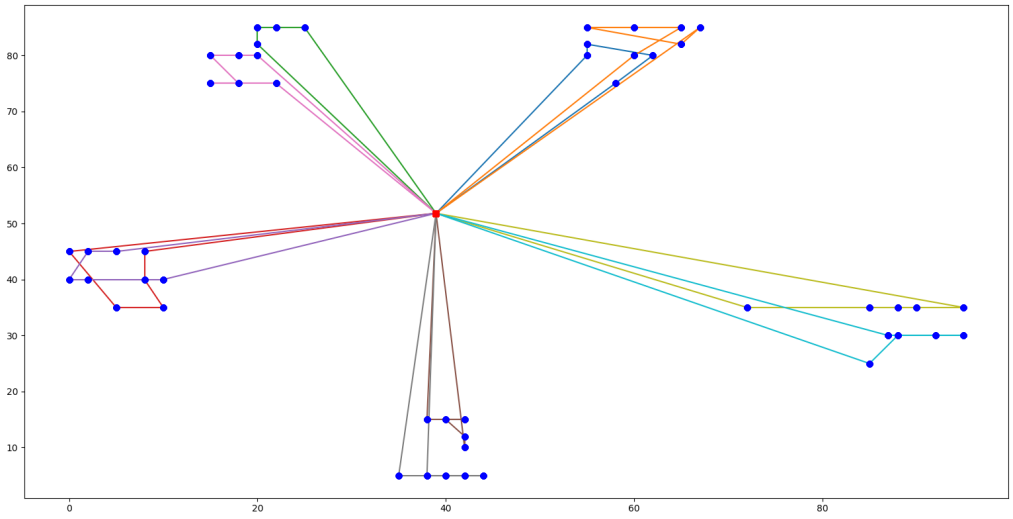
(b) Random



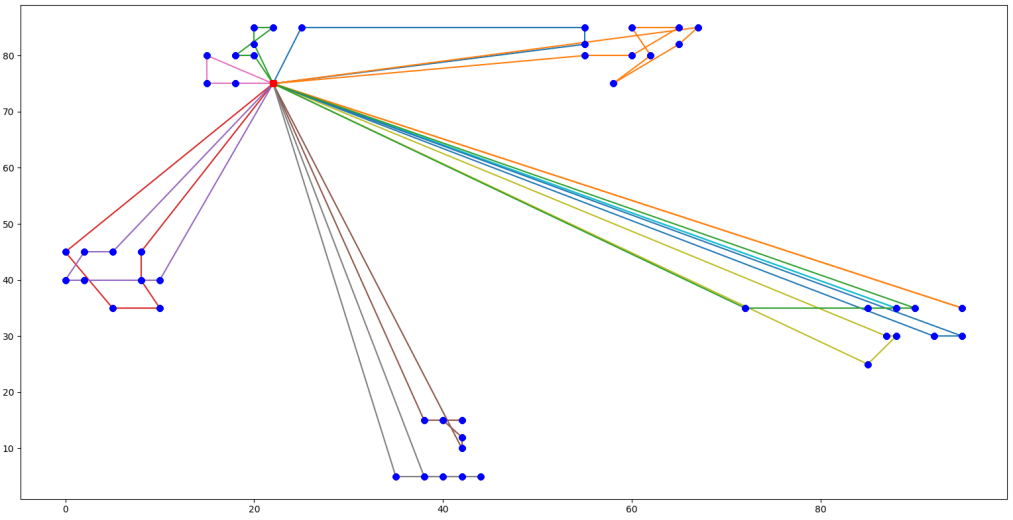
(c) Semi-Clustered

Figure 6.6: Customer Distribution, Percentile Difference with the Continuous Method

Figure 6.7 compares a continuous and discrete solution and shows why the discrete method suffers so much in these circumstances. The clusters form a circle with the optimal depot location somewhere in the center. Since the discrete method only allows placement on customer nodes, it forces the placement in one of the clusters. This means the cluster opposite, in relation to the center, is very far away. To still conform to the specified time windows, we need to assign extra vehicles to service that cluster. Since that cluster is so far away, the costs increase dramatically.



(a) Continuous Method, Solution Cost = 977.76



(b) Discrete Method, Solution Cost = 1552.18

Figure 6.7: Example Solutions Semi-Clustered 50 Customers

6.2.2 Modifying Constraints

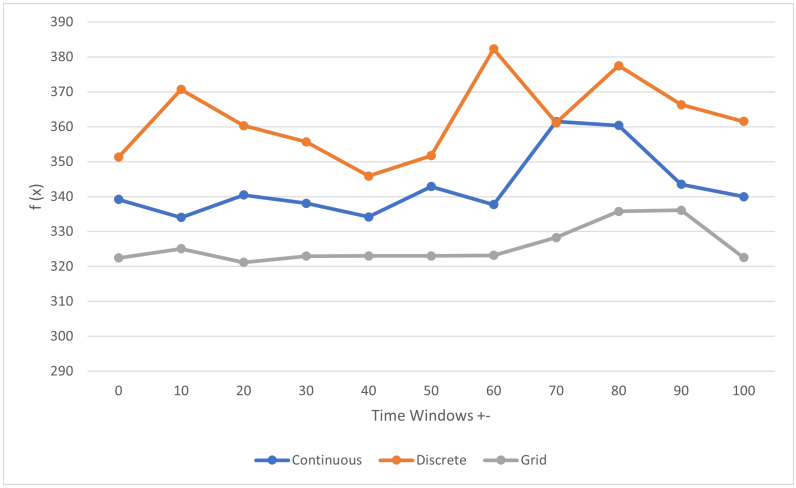
This section looks at what happens when the time windows and capacity constraints are altered. In the case of the time windows, we widen each time window with a certain amount of time. Alternatively, the capacity constraints are further restricted instead of loosened, as the problem is mostly time window restricted.

Time Windows

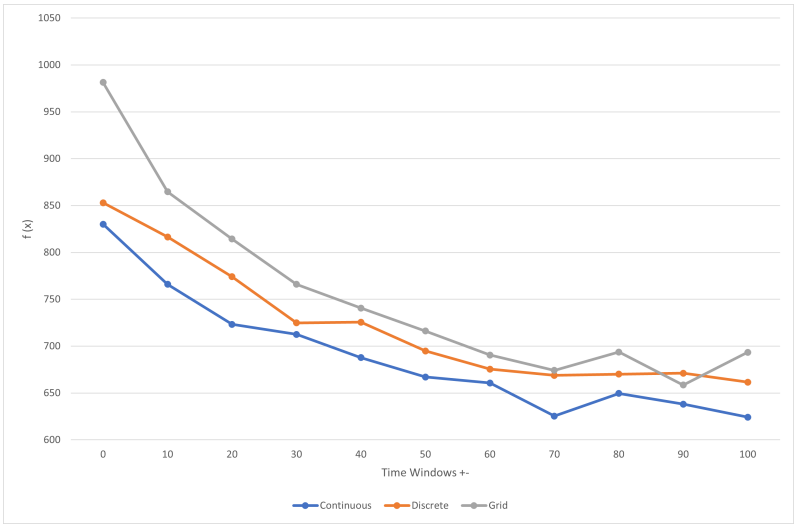
Each customer i has a specific time window in which they need to be serviced. The time window consists of a starting time (a_i) and end an end time (b_i). We can relax these time window restrictions by lowering time a_i and increasing time b_i with a certain amount of time. Figure 6.8 compares the three methods on each data set with 50 customers. The time windows range from the default (0) to widened by 100 units.

We expect the objective function $f(x)$ to decrease as the time windows widen since lower restrictions should result in a lower total cost. This effect is clearly observable for the random and semi-clustered sets, where all methods show a decreasing trend as the time windows are widened. However, this effect is not present for the clustered set, and in fact, the solution seems to regress as the restrictions are relaxed. This is because, as discussed in Chapter 5, the time windows for the clustered set are chosen so that one vehicle can service one cluster. The time windows are not a restricting factor in this problem; thus, loosening these restrictions has little effect on improving the solution. Moreover, it is actually counterproductive as the wide time windows allow for an increased number of feasible solutions and, therefore, an increased number of potential local optima for the VNS to get stuck in.

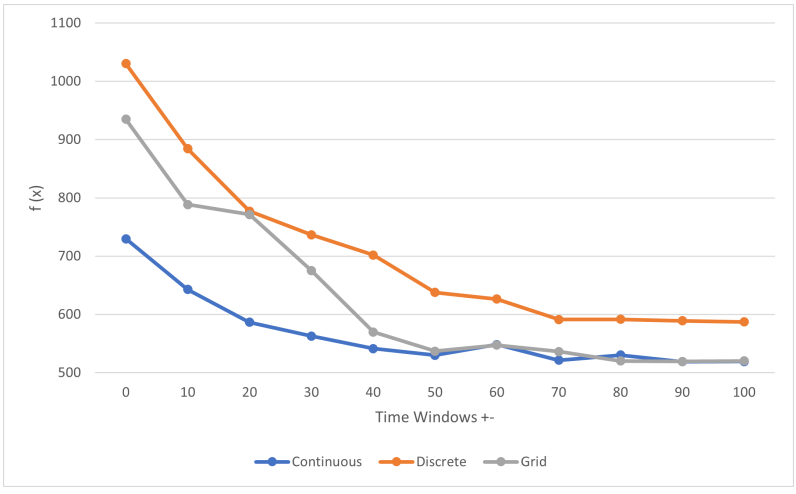
We see that all methods benefit from widening time windows for both the random and semi-clustered sets. However, the methods do not all benefit equally. The difference between the continuous and discrete methods in the random set remains stable across the x-axis. In the semi-clustered set, the difference between the continuous and discrete methods decreases as the time windows are widened.



(a) Clustered



(b) Random

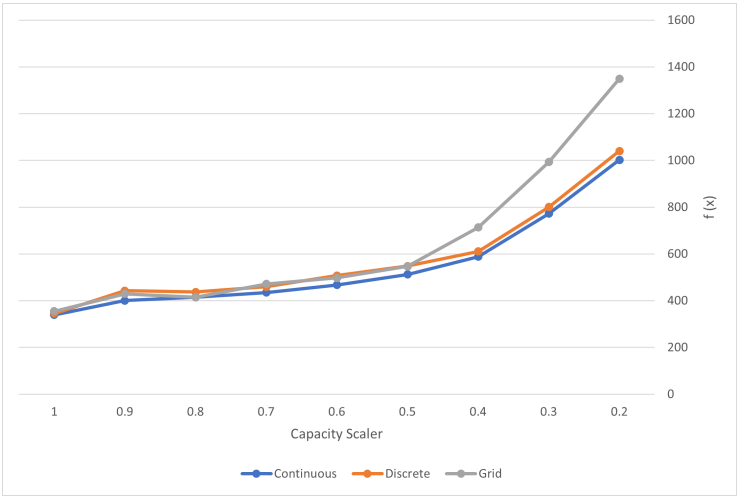


(c) Semi-Clustered

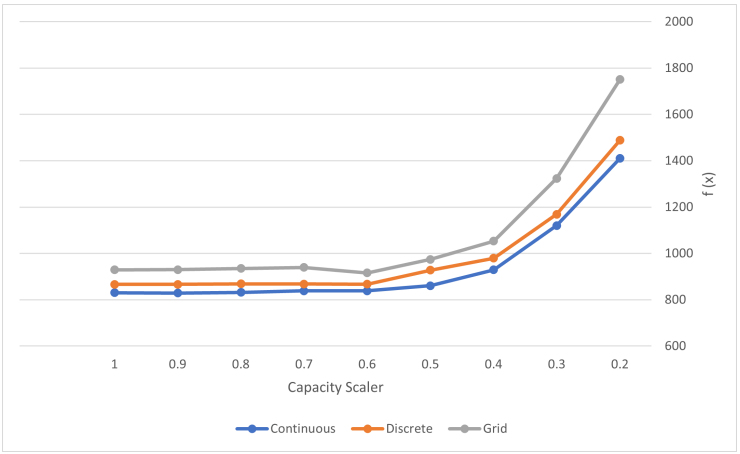
Figure 6.8: Widening Time Windows N = 50

Capacity

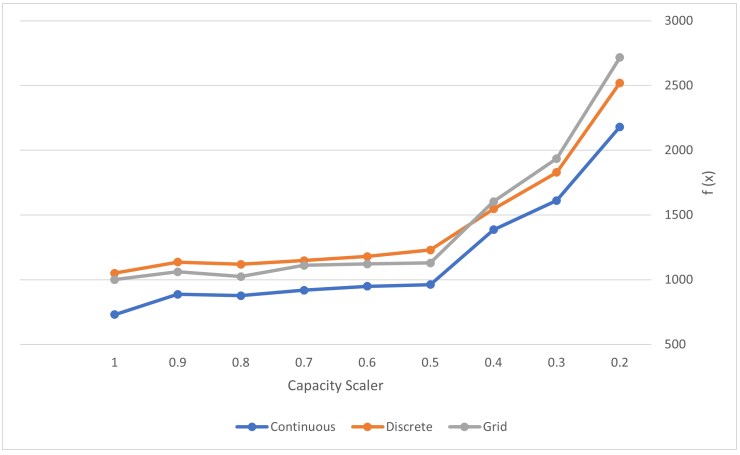
To understand the effect of capacity constraints on the different methods, we scale the capacity of the vehicles by a certain factor. This factor ranges from 1 (default) to 0.2. In Figure 6.9, we see that for the random and semi-clustered data sets, the capacity has little influence on the three methods up until around 0.5. At this point, all methods suddenly trend upward sharply. The difference between the continuous and discrete remains equal, but the grid method diverges upwards as the capacity becomes increasingly limited. In the clustered set, all methods gradually increase with the grid method again diverging. This indicates that the random and semi-clustered data sets are mostly limited by time windows, while the clustered set is more limited by capacity constraints, as the time windows had little effect on this data set. Interestingly, the effect of altering the capacity of vehicles does not show a difference between the continuous and discrete methods, but it does show a difference for the grid method in all data sets.



(a) Clustered



(b) Random

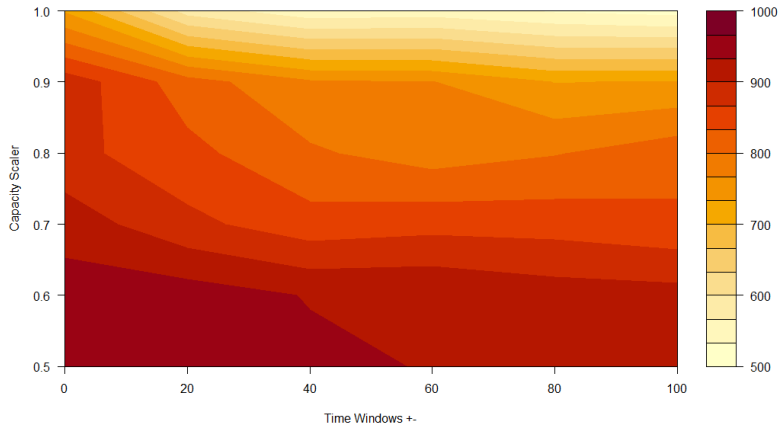


(c) Semi-Clustered

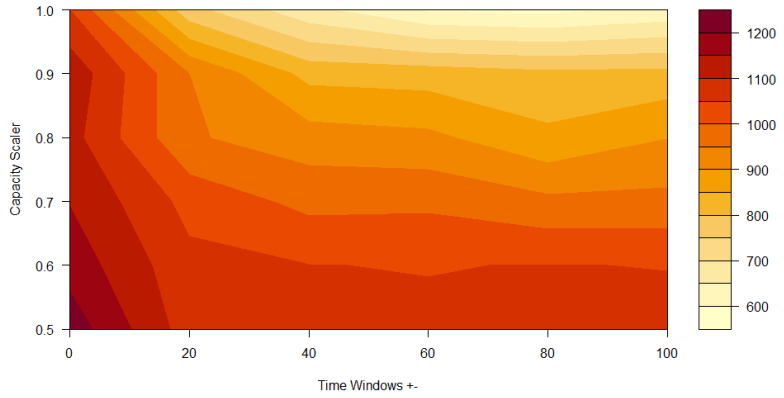
Figure 6.9: Restricting capacity N = 50

Combined Effect

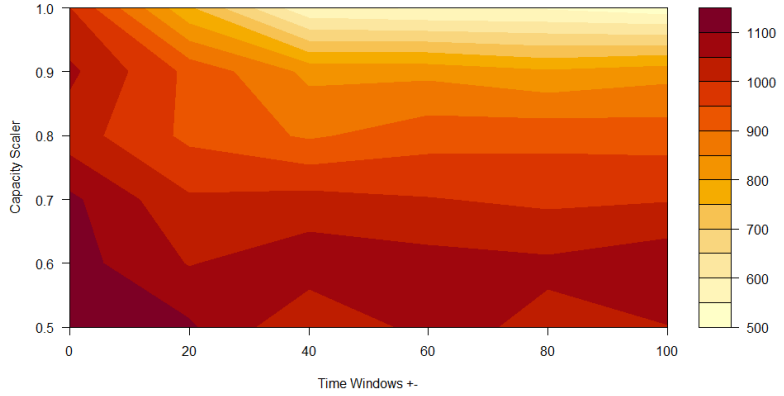
Contour plots show how the methods behave if time windows and capacity change simultaneously. Figure 6.10 shows contour plots for all three methods on the semi-clustered data set with 50 customers. The near vertical lines on the left side of the discrete plot indicate that the discrete method first benefits heavily from the widened time windows and, secondly, from increased capacity. The diagonal lines on the left side of the continuous plot indicate it is more equally affected by time windows and capacity changes. The grid method shows lines changing direction in some cases, resulting in higher variability in the results.



(a) Continuous



(b) Discrete



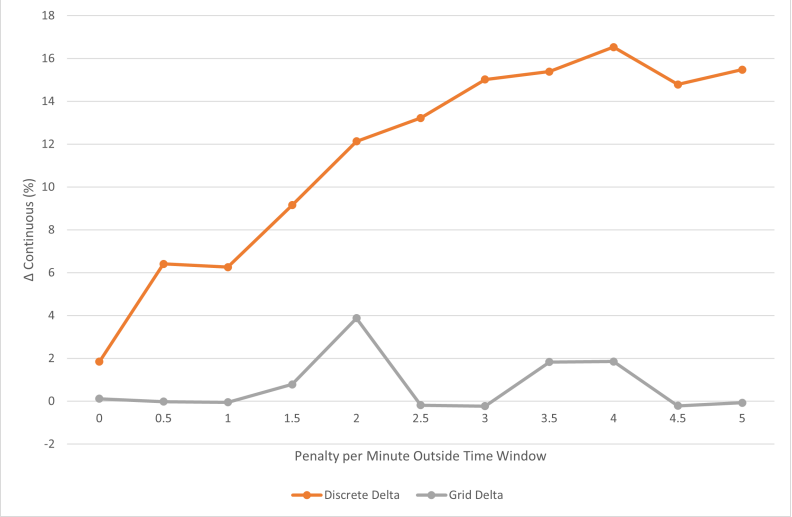
(c) Grid

Figure 6.10: Time Windows and Capacity, Semi-Clustered N = 50

6.2.3 Realistic Scenario

Since the previous experiments lean more towards the theoretical side, we try to explore a more realistic scenario in this section. To do this, we need to make a couple of changes. The first change is switching from hard-time windows to soft-time windows. This means that instead of forcing the algorithm to find a solution that respects all time windows, we let it decide to arrive earlier or later than the specified window, but with an added penalty cost for every minute outside the window. Vehicle capacity constraints are still strictly enforced, as vehicle overloading should always be avoided. We also include a fixed cost per vehicle or route used in the solution and realistic costs per kilometer driven with respect to fuel and employee wages.

A truck with a capacity of around 10,000 kg costs €828 per month to lease. Leasing removes the need for a more complicated cost structure that includes depreciation and maintenance costs. The current diesel price is €1.80, and the trucks use an average of 35 liters per 100 kilometers. This results in an average fuel cost of €0.63 per kilometer. The average salary for truck drivers is around €15.56 per hour or €0.26 per minute. The only parameter that is still undefined is the penalty cost per minute outside the specified time window. Since this is not a directly tangible cost, it will be interesting to see how changing this variable affects the three solution methods. We let the penalty costs range from €0.00 to €5.00 per minute. Figure 6.11a shows the relative difference between the continuous and the discrete and grid methods on the semi-clustered data set with 50 customers. As the penalty for missing time windows increases, the discrete method diverges further from the continuous and grid methods. This is consistent with earlier results showing that stricter time windows disproportionately affect the discrete method.



(a) Increasing Penalty Soft Time Windows, Semi-Clustered N = 50

Chapter 7

Conclusion

In this thesis, we compare the discrete and continuous Location Routing Problem. The problem includes capacity and time window constraints. We use a modified version of the Solomon instances to understand how certain instance characteristics influence the relative performance of the continuous and discrete LRP. To solve the problem, we use a Variable Neighborhood Search meta-heuristic, which utilizes a variety of potential solution structures to escape local optima.

The experiments show that the difference in solution quality is low for small instances for all tested distributions. The gap between the continuous and discrete methods grows for larger instances, favoring the continuous method. The discrete method should be avoided in case of a very clustered distribution with complex time windows. In all other tested cases, the discrete method proves a viable strategy, although very rarely the optimal one.

Further testing shows that widening time windows disproportionally benefits the discrete method but only for the semi-clustered data set. In the random data set, the difference between the continuous and discrete methods remains equal. However, the grid method does benefit more from widening time windows for the random data set.

Limiting the capacity of vehicles shows no relative change between the continuous and discrete methods. Only the grid method is affected to a larger extent. Combining the effects of widening the time windows and decreasing vehicle capacity shows that the discrete method is mostly affected by widening time windows first and then capacity changes. The continuous method is affected more equally by both effects.

In a more realistic scenario, we tested what would happen in case of soft time windows and increasing penalties for time window violations. The tests show that as the penalty increases, the discrete method performs relatively worse. The grid method performs similarly to the continuous method but suffers more from occasional outliers.

One of the limitations of this thesis is the use of the Solomon Instances. These data sets were made primarily to test VRP solution methods. They do not necessarily represent a realistic scenario. Should further research be conducted, the use of an LRP-specific data set would be highly recommendable.

In general, when using the discrete method, researchers should take special care in situations with a combination of strict time windows and a clustered distribution of customers. This combination of characteristics can cause a detrimental loss of quality to the solution of the discrete method. The grid method shows a situational improvement over the continuous method but is, in its current implementation, too unreliable. The grid method is much more likely to get stuck in a local optimum. Should the reliability be improved, it could become competitive with the continuous method.

Bibliography

Asefi, H., Lim, S., Maghrebi, M., & Shahparvari, S. (2019). Mathematical modelling and heuristic approaches to the location-routing problem of a cost-effective integrated solid waste management. *Annals of Operations Research*, *273*(1-2), 75–110.

Aykin, T. (1995). The hub location and routing problem. *European Journal of Operational Research*, *83*(1), 200–219.

Baniamerian, A., Bashiri, M., & Tavakkoli-Moghaddam, R. (2019). Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking. *Applied Soft Computing*, *75*, 441–460.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, *4*, 238–252.

Bentley, J. J. (1992). Fast Algorithms for Geometric Traveling Salesman Problems. *ORSA Journal on Computing*, *4*(4), 387–411.

Boventer, E. (1961). The Relationship Between Transportation Costs and Location Rent in Transportation Problems. *Journal of Regional Science*, *3*(2), 27–40.

Braekers, K., Ramaekers, K., & Van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, *99*, 300–313.

Campbell, J. F. (1992). Location and allocation for distribution systems with transshipments and tranportion economies of scale. *Annals of Operations Research*, *40*, 77–99.

Campbell, J. F. (1994). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, *72*(2), 387–405.

Chobar Pourghader, A., Sabk Ara, M., Moradi Pirbalouti, S., Khadem, M., & Bahrami, S. (2021). A multi-objective location-routing problem model for multi-device relief logistics under uncertainty using meta-heuristic algorithm. *Journal of Applied Research on Industrial Engineering*, *9*(3), 354–373.

Clarke, G., & Wright, J. W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, *12*(4), 568–581.

Croes, G. A. (1958). A Method for Solving Traveling-Salesman Problems. *Operations Research*, *6*(6), 798–812.

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, *6*(1), 80–91.

Darvish, M., Archetti, C., Coelho, L. C., & Speranza, M. G. (2019). Flexible two-echelon location routing problem. *European Journal of Operational Research*, *277*(3), 1124–1136.

Davoodi, S. M. R., & Goli, A. (2019). An integrated disaster relief model based on covering tour using hybrid Benders decomposition and variable neighborhood search: Application in the Iranian context. *Computers & Industrial Engineering*, *130*, 370–380.

Dukkanci, O., & Kara, B. Y. (2017). Routing and scheduling decisions in the hierarchical hub location problem. *Computers & Operations Research*, *85*, 45–57.

Eftelioglu, E. (2015). Geometric Median. *Encyclopedia of GIS*, 1–4.

Elshaer, R., & Awad, H. (2020). A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Computers & Industrial Engineering*, *140*, 106242.

Fallahtafti, A., Ardjmand, E., Young, W. A., & Weckman, G. R. (2021). A multi-objective two-echelon location-routing problem for cash logistics: A metaheuristic approach. *Applied Soft Computing*, *111*.

Fazayeli, S., Eydi, A., & Kamalabadi, I. N. (2018). Location-routing problem in multimodal transportation network with time windows and fuzzy demands: Presenting a two-part genetic algorithm. *Computers & Industrial Engineering*, *119*, 233–246.

Ferreira, K. M., & de Queiroz, T. A. (2018). Two effective simulated annealing algorithms for the Location-Routing Problem. *Applied Soft Computing*, *70*, 389–422.

Flach, P. (2012). *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University.

Ghasemi, P., Hemmaty, H., Pourghader Chobar, A., Heidari, M. R., & Keramati, M. (2022). A multi-objective and multi-level model for location-routing problem in the supply chain based on the customer's time window. *Journal of Applied Research on Industrial Engineering*.

Guimarans, D., Dominguez, O., Panadero, J., & Juan, A. A. (2018). A simheuristic approach for the two-dimensional vehicle routing problem with stochastic travel times. *Simulation Modelling Practice and Theory*, *89*, 1–14.

Hansen, P., Mladenović, N., Todosijević, R., & Hanafi, S. (2017). Variable neighborhood search: basics and variants. *EURO Journal on Computational Optimization*, *5*(3), 423–454.

Hemmelmayr, V. C. (2015). Sequential and parallel large neighborhood search algorithms for the periodic location routing problem. *European Journal of Operational Research*, *243*(1), 52–60.

Hiassat, A., Diabat, A., & Rahwan, I. (2017). A genetic algorithm approach for location-inventory-routing problem with perishable products. *Journal of Manufacturing Systems*, *42*, 93–103.

Karakostas, P., Sifaleras, A., & Georgiadis, M. C. (2022). Variable neighborhood search-based solution methods for the pollution location-inventory-routing problem. *Optimization Letters*, *16*(1), 211–235.

Kumar, S. N., & Panneerselvam, R. (2012). A Survey on the Vehicle Routing Problem and Its Variants. *Intelligent Information Management*, *04*(03), 66–74.

Manzour-Al-Ajdad, S. M., Torabi, S. A., & Salhi, S. (2012). A hierarchical algorithm for the planar single-facility location routing problem. *Computers & Operations Research*, *39*(2), 461–470.

Mara, S. T. W., Kuo, R. J., & Asih, A. M. S. (2021). Location-routing problem: a classification of recent research. *International Transactions in Operational Research*, *28*(6), 2941–2983.

Maranzana, F. E. (1964). On the Location of Supply Points to Minimize Transport Costs. *Journal of the Operational Research Society*, *15*(3), 261–270.

Martins de Sá, E., Morabito, R., & de Camargo, R. S. (2018). Benders decomposition applied to a robust multiple allocation incomplete hub location problem. *Computers & Operations Research*, *89*, 31–50.

Mjirda, A., Todosijević, R., Hanafi, S., Hansen, P., & Mladenović, N. (2017). Sequential variable neighborhood descent variants: an empirical study on the traveling salesman problem. *International Transactions in Operational Research*, *24*(3), 615–633.

Mladenovic, N., Sleptchenko, A., Sifaleras, A., & Omar, M. (2021). *Variable Neighborhood Search* (N. Mladenovic, A. Sleptchenko, A. Sifaleras & M. Omar, Eds.; Vol. 12559). Springer International Publishing.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, *24*(11), 1097–1100.

New York Times. (2021). The Biggest Kink in America's Supply Chain: Not Enough Truckers - The New York Times.

Nucamendi-Guillén, S., Martínez-Salazar, I., Khodaparasti, S., & Bruni, M. E. (2022). New formulations and solution approaches for the latency location routing problem. *Computers & Operations Research*, *143*, 105767.

O'Kelly, M. E. (1986). The Location of Interacting Hub Facilities. *Transportation Science*, *20*(2), 92–106.

Pasha, J., Nwodu, A. L., Fathollahi-Fard, A. M., Tian, G., Li, Z., Wang, H., & Dulebenets, M. A. (2022). Exact and metaheuristic algorithms for the vehicle routing problem with a factory-in-a-box in multi-objective settings. *Advanced Engineering Informatics*, *52*, 101623.

Pourmohammadi, P., Tavakkoli-Moghaddam, R., Rahimi, Y., & Triki, C. (2021). Solving a hub location-routing problem with a queue system under social responsibility by a fuzzy metaheuristic algorithm. *Annals of Operations Research*, 1–30.

Quintero-Araujo, C. L., Guimarans, D., & Juan, A. A. (2019). A simheuristic algorithm for the capacitated location routing problem with stochastic demands. *Journal of Simulation*, *15*(3), 217–234.

Rabbani, M., Heidari, R., & Yazdanparast, R. (2019). A stochastic multi-period industrial hazardous waste location-routing problem: Integrating NSGA-II and Monte Carlo simulation. *European Journal of Operational Research*, *272*(3), 945–961.

Reyes-Rubiano, L., Ferone, D., Juan, A. A., & Faulin, J. (2019). A simheuristic for routing electric vehicles with limited driving ranges and stochastic travel times. *Statistics and Operations Research Transactions*, *1*(1), 3–24.

Romanycia, M. H., & Pelletier, F. J. (1985). What is a heuristic? *Computational Intelligence*, *1*(1), 47–58.

Sadati, M. E. H., & Çatay, B. (2021). A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, *149*, 102293.

Saifullah Hussin, M., & Stützle, T. (2014). Tabu search vs. simulated annealing as a function of the size of quadratic assignment problem instances. *Computers & Operations Research*, *43*, 286–291.

Schwardt, M., & Dethloff, J. (2005). Solving a continuous location-routing problem by use of a self-organizing map. *International Journal of Physical Distribution & Logistics Management*, *35*(6), 390–408.

Schwardt, M., & Fischer, K. (2009). Combined location-routing problems-a neural network approach. *Annals of Operations Research*, *167*(1), 253–269.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, *35*(2), 254–265.

Sörensen, K., & Glover, F. (2013). Metaheuristics. *Encyclopedia of operations research and management science*, *62*, 960–970.

Soylu, B. (2015). A general variable neighborhood search heuristic for multiple traveling salesmen problem. *Computers & Industrial Engineering*, *90*, 390–401.

Tavares, L. G., Lopes, H. S., & Lima, C. R. (2009). Construction and improvement heuristics applied to the capacitated vehicle routing problem. *2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings*, 690–695.

Toth, P., & Vigo, D. (2002). *The vehicle routing problem*. Society for Industrial & Applied Mathematics.

Tuzun, D., & Burke, L. I. (1999). A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, *116*(1), 87–99.

Wang, Y., Assogba, K., Liu, Y., Ma, X., Xu, M., & Wang, Y. (2018). Two-echelon location-routing optimization with time windows based on customer clustering. *Expert Systems with Applications*, *104*, 244–260.

Wasner, M., & Zäpfel, G. (2004). An integrated multi-depot hub-location vehicle routing model for network planning of parcel service. *International Journal of Production Economics*, *90*(3), 403–419.

Yan, T., Lu, F., Wang, S., Wang, L., Bi, H., Yan, T., Lu, F., Wang, S., Wang, L., & Bi, H. (2023). A hybrid metaheuristic algorithm for the multi-objective location-routing problem in the early post-disaster stage. *Journal of Industrial and Management Optimization*, *19*(6), 4663–4691.

Yu, V. F., & Lin, S. Y. (2015). A simulated annealing heuristic for the open location-routing problem. *Computers & Operations Research*, *62*, 184–196.

Yu, V. F., Maglasang, R., & Tsao, Y. C. (2020). A reduced variable neighborhood search-based hyperheuristic for the shelf space allocation problem. *Computers & Industrial Engineering*, *143*, 106420.

Yu, X., Zhou, Y., & Liu, X.-F. (2020). The two-echelon multi-objective location routing problem inspired by realistic waste collection applications: The composable model and a metaheuristic algorithm. *Applied Soft Computing*, *94*, 106477.

Zhao, J., & Ke, G. Y. (2017). Incorporating inventory risks in location-routing models for explosive waste management. *International Journal of Production Economics*, *193*, 123–136.

Appendix A

Mathematical Formulation VRP

In this chapter, the notation and formulation of the capacitated VRP with time windows are explained.

The problem formulation is as follows: A set of vehicles; K, capacity C_k, will deliver goods to each customer in the set of vertices V, this set also represents the nodes in the CVRP where 1 is the depot. d_i represents the demand for customer $i \in V$. If a vehicle travels from customer i to customer j, both from the set V, the binary variable x_{ij} will be 1, indicating the arc from i to j has been chosen. The time it takes for a vehicle to drive from i to j is defined as t_{ij} and the time it takes to service customer i is defined as s_i. The start time of service at customer i with vehicle k is defined as w_{ik}.

Delivering goods to customers, whether they are individuals or organizations, is usually only possible within a certain time-window. Therefore the vehicles are only allowed to deliver in the interval a_i, b_i. The earliest possible departure from, and the latest possible arrival at the depot are defined as E and L respectively.

Sets	Description
V	Set of nodes, indexed by $i, j \in V$, the depot is 0
K	Set of vehicles, indexed by k
Parameters	**Description**
c_{ij}	Costs associated with arc (i, j)
d_i	Demand of customer i
C_k	Capacity of vehicle k
a_i, b_i	Time window for customer i
t_{ij}	Travel time from customer i to customer j
s_i	Additional service time for customer i
w_{ik}	Start time of service at customer i with vehicle k
E	Earliest possible departure from the depot
L	Latest possible arrival at the depot
Decision Variables	**Description**
x_{ijv}	Binary variable to choose arc between i and j with vehicle v

Table A.1: Notation

The following model is adapted from Toth and Vigo (2002). The notation "Δ^-" and "Δ^+" mean incoming and outgoing arcs respectively.

$$\min \quad \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{ij} \cdot x_{ijk} \tag{A.1}$$

Equation A.1 gives the objective of the problem, which is to minimize the total costs associated with the chosen routes.

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1 \qquad \forall i \in N, \tag{A.2}$$

$$\sum_{j \in \Delta^+(0)} x_{0jk} = 1 \qquad \forall k \in K, \tag{A.3}$$

$$\sum_{i \in \Delta(j)} x_{ijk} - \sum_{i \in \Delta^+(j)} x_{jik} = 0 \qquad \forall k \in K, j \in N, \tag{A.4}$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,k} = 1 \qquad \forall k \in K, \tag{A.5}$$

$$w_{ik} + s_i + t_{ij} - w_{jk} \leq (1 - x_{ijk})\, M_{ij} \qquad \forall k \in K, (i,j) \in A \tag{A.6}$$

$$a_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq w_{ik} \leq b_i \sum_{j \in \Delta^-(i)} x_{ijk} \qquad \forall k \in K, i \in N, \tag{A.7}$$

$$E \leq w_{ik} \leq L \qquad \forall k \in K, i \in \{0, n+1\}, \tag{A.8}$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq C \qquad \forall k \in K, \tag{A.9}$$

$$x_{ijk} \in \{0,1\} \qquad \forall k \in K, (i,j) \in A. \tag{A.10}$$

Equation A.2 restricts each customer to exactly one route. Equations A.3, A.4 and A.5, define the path that a vehicle is allowed to take. Equations A.6, A.7, A.8 and A.9 impose time window and capacity constraints. Finally, Equation A.10 imposes a binary condition on the decision variable x.

Appendix B

First and Steepest Descent

Instance	Method	Mean solution cost	Mean computation time (s)
R1 $N = 25$	First descent	513.1	2.1
R1 $N = 25$	Steepest descent	517.6	3.9
R1 $N = 50$	First descent	837.1	22.1
R1 $N = 50$	Steepest descent	859.8	48.8
R1 $N = 100$	First descent	1387.0	286.1
R1 $N = 100$	Steepest descent	1359.4	940.1

Table B.1: First Descent and Steepest Descent