A two stage approach to solving a robust vehicle routing problem under demand uncertainty

Operations Research and Quantitative Logistics Master Thesis

Erasmus University Rotterdam
Erasmus School of Economics

Name: Giorgia d'Arcano
Student ID Number: 512396
Supervisor: Dr. T.A.B. Dollevoet
Second Assessor: Dr. W. van den Heuvel
Date: July 30 2020

Abstract

In this thesis we consider the vehicle routing problem under demand uncertainty. The deterministic vehicle routing problem has been widely studied in all its variants, from time windows to pick-up and delivery stops combined. In the last decade it has become more and more important for companies to be flexible and competitive and to do so an efficient distribution is key. Hence the importance of routing vehicles under uncertainty.

In this thesis we solve a robust vehicle routing problem under uncertainty in customer demand by means of a two-stage optimization process. Well-known heuristics are implemented to construct the routes and recourse actions are later adopted to amend possible failures. The first stage problem is solved as a deterministic vehicle routing problem, and the uncertainty is taken into account only in the second stage problem.

The results show that the two-stage approach is a fast and efficient way to tackle vehicle routing problems, adapting the initial solution to the demand realisations, as they are known, via the second stage. The constructed method achieves good results on different instances proving its performance is independent of the size of the problem.

Keywords: Robust Vehicle Routing, Tabu Search, Demand uncertainty, Recourse actions, Two-stage optimisation, Monte Carlo simulation.

Contents

1 Introduction

Since the introduction of routing problems in the 1950s, their importance has never diminished. To the contrary, more complex scenarios and variants are continuously arising. The first, and simplest, vehicle routing problem is a generalisation of the well-known Travelling Salesman problem. It considers the problem of determining a set of routes for a fleet of vehicles in order for each to transport goods from the depot to a subset of customers while minimising costs. Since then, an endless search has emerged for a system that can determine a set of routes for a fleet of vehicles, taking into account all possible limitations, such as time windows, multiple depots, heterogeneous vehicles, and even pickup and delivery coordination, all while minimizing costs.

The surge of e-commerce and relative increase of the B2C share of the market has only increased the need for efficient routing algorithms (Joerss et al., 2016). The distribution sector has not only seen changes from a quantity perspective but also from a speed perspective with many businesses, from supermarkets to the retail industries, offering same day delivery. Firms wanting to offer a high speed service are required to be flexible in their operations. In this fast growing market, having an efficient distribution is key to gain a competitive edge. B2C is not the only segment gaining competitive advantage from an efficient routing system, B2B often deals with larger quantities and regular customers to service daily, weekly or monthly. In today's dynamic consumption industry, companies need to be flexible to serve as many customers as possible, both in B2B and in B2C segments. Flexibility is necessary in order to adjust to new orders, cancellations and possible uncertainty in demand or travel times, without excess costs.

However, flexibility also entails a routing algorithm that can adapt as the orders change or are known with certainty. Demand uncertainty has been, and continues to be, a complication of efficiently determining routes and for companies to be flexible it is necessary to be able to handle some level of uncertainty. Uncertainty in vehicle routing problems can present itself in many ways, in customer demand, in the number of customers, and in travel time uncertainty. Customer demand uncertainty is faced by many industries, from waste collection to water distribution and ATM cash replenishment. Here, customers are provided the freedom of finalizing their demand once the vehicle has arrived at the customer's location or with little time notice. An example of uncertainty regarding the number of customers is faced by restaurants and other industries in the food sector, however, these follow a newspaper vendor problem rather than a routing problem. Finally, uncertainty in travel times is a well-known obstacle to many privates as well as public identities. It is important to take uncertainty in travel times into account in a routing problem as this can have a substantial effect on the real duration of the routes.

Uncertainty entails the real value of a parameter is not known, however in many settings a company might have statistical information regarding the uncertain parameter or may be able to obtain this statistical information from historical data. In such a case the problem at hand is denoted as a stochastic vehicle routing problem. The waste collection problem mentioned above, is an instance of a problem with uncertainty in the demand however there is commonly sufficient historic data in order to correctly estimate the distribution of the demand, and thus apply stochastic method as shown by Singh (2019).

In many other cases, there is no statistical information given and there is insufficient or unreliable historic data to make a safe estimate of the distribution. An example of this is given by the ATM replenishment, the demand of each customer is unknown until the customer reaches the machine, however it is important for the machine to always have sufficient cash to withdraw. As Ekinci et al. (2019) describe in their paper, it is important to correctly take the uncertainty into account in order to minimise both holding costs and replenishment costs. The authors apply robust optimization methods to identify the best replenishment policy (Ekinci et al., 2019).

While it is an increasingly important problem in real-life scenarios, the robust vehicle routing problem has not been treated as in depth as its stochastic counterpart. It is important for companies to have a routing system that can deal efficiently with uncertainty in order to maintain their competitive edge, from an economical point of view, and indirectly in order to minimise the number of deployed vehicles, from a sustainable point of view. Therefore it is necessary to improve on the existing robust methods, to meet the flexibility that companies today are expected to have.

This thesis aims to present a new way to solve robust vehicle routing problems with demand uncertainty, where the vehicles have a limit on their capacity. First the problem at hand is described in Section 2, then the relative literature is discussed in Section 3. Subsequently, the methodology adopted is discussed in Section 4, followed by a description of the chosen data in Section 5. Finally the results are presented in Section 6, accompanied by a sensitivity analysis in Section 7 and concluding observations.

2 Problem description

2.1 Robust vehicle routing problem

The robust vehicle routing problem is a variant of the classic vehicle routing problem (VRP) in which one or more parameters are not known with certainty. Furthermore, what differentiates the robust VRP from the stochastic VRP is that, in the former, the distribution of the uncertain parameter is unknown. Uncertainty in robust VRPs is often related to the demand quantity of customers or the travelling times between stops. As travel times are regarded to be more time dependent rather than uncertain, the focus of this thesis will be uncertainty in the demand of customers (Spliet, 2013). The objective when solving any robust optimisation problem is to determine a solution which is feasible for all, or a subset, of possible outcomes of the uncertain parameter. When solving for all possible outcomes we face a worst case scenario optimization. In some industries it is of utmost importance to be able to fulfill all customers' demand with an a priori solution and thus a worst case robust optimization is the safe approach to undertake. However, in many fields of work, such as the food and beverage distribution or retail delivery, it is reasonable to assume the worst case scenario will not occur, i.e. not all customers will require their highest possible demand on the same day. In this case taking the worst case approach can result in an over-conservative solution and an inefficient use of the fleet of vehicles. Thus a scenario with the expected values, rather than the worst case values, is considered or the uncertainty set is adjusted to include only a subset of the possible demand outcomes. Furthermore, it is often the case that, once the vehicles have been dispatched, the routes can be adjusted to incorporate new information learnt during the route regarding the uncertain parameters. This thesis presents a two-stage based approach where initial routes are defined and then adjusted if and when needed.

2.2 The approach undertaken

This thesis presents an approach divided in two stages, with the aim of finding a feasible solution for all possible demand outcomes. In the first stage, we determine a set of routes by solving a deterministic constrained VRP (CVRP). We assume some values around the demand are known, as to make some assumptions on each customer's expected demand.

The second stage is dynamic, the input routes, obtained from the first stage, can be adjusted in order to respond to the new information gathered as customers are visited. When we reach a customer, the true demand is known, thus if many customers on a route exceed their expected demand, a failure could occur. A failure takes place when a vehicle has insufficient remaining capacity to meet a customer's demand. Thus the objective of the second stage is to ensure feasibility by implementing recourse actions. We consider different recourse actions, namely partial rescheduling and detour to depot. We want to avoid trips to the depot unless strictly necessary or cost efficient, thus the route which is in need of adjustment due to failure is reviewed together with the routes of vehicles nearby. This allows for the routes to be partially rescheduled, without a time consuming algorithm that revisits all routes, and minimises trips to the depot which increase the length of the route making them less efficient.

The approach is tested many times on the same instance as the demand outcomes are random and thus, to truly test the performance of the method proposed, many tests need to be carried out. Subsequently, for each instance the worst result is taken to represent the performance in order to maintain a robust scenario and avoid assuming any probability distribution over the demand outcomes.

2.2.1 The uncertainty set

In the above section it was mentioned that the approach undertaken will be tested many times on the same instance, as the outcomes of the demand are random. Seeing as we are solving a robust CVRP, the worst result yielded will be taken to reflect the performance of the method. All possible demand occurrences together form the uncertainty set. If all customers are allowed to require their highest possible demand value simultaneously, the approach yielding the best solution would be to consider the worst case for all customers. In many real world scenarios it is very plausible to assume that the above would not occur. Therefore we limit the size of the uncertainty set by assuming there will be an even spread over the customers that exceed and those who are below their expected demand, q_i. This is done by setting an upper-bound on the total possible demand, B, of customers. The uncertainty set then reduces to:

$$Z = \left\{ \mathbf{d} : \sum_{i=1}^{n} d_i \leq B \right\} \tag{1}$$

where d_i is the demand realization of customer i, and

$$B = (\sum_{i=1}^{n} q_i) \cdot 1.05 \tag{2}$$

An alternative way of reducing the uncertainty set is by limiting the total deviation the customers can have from their nominal value. However, taking into account that the demand of customers can range from very small to very large numbers of units, limiting the percentage of deviation can be misleading on how it would limit the total demand requested.

2.2.2 Assumptions

Before describing how the robust VRP will be solved in this thesis, the underlying assumptions that shape the problem at hand are presented:

1. all vehicles must start and end their route at the depot,
2. vehicle routes can be adapted during the day as information regarding the customers' demands becomes available,
3. all vehicles have a limit on capacity Q,
4. all vehicles have a limit on the duration of their route L,
5. each customer must be visited exactly once by one vehicle only,
6. customer's demand is only known upon arrival at the customer's location,
7. only the nominal value of each customer's demand is known at the start of the day,
8. servicing times of customers are known,
9. distances between customers are symmetric, i.e. distance from a to b is the same as the distance from b to a.

Assumptions 1, 5 and 9 are quite common in many VRP problem formulations. Assumptions 3 and 4 are the feasibility constraints, if a vehicle violates either constraint, the solution is not feasible. Both constraints are in place to include real world limitations such as the quantity a vehicle can carry and, less commonly seen in literature, a limit on the time a route can take to ensure the driver's shift is respected. Furthermore, a failure occurs when a vehicle would violate the capacity constraint by servicing its current customer. And finally, assumptions 2, 6 and 7 are related to the uncertainty parameters of our problem formulation.

2.2.3 Objective

As in most vehicle routing problems, the objective is to find a set of routes with minimal cost while respecting a set of limitations. As mentioned in section 2.2.2, there are two feasibility constraints imposed on the routes: an upper bound on the quantity a vehicle can carry, and an upper bound on the length of each route. The first constraint is easy to verify by a simple check of the total demand of customers assigned to a vehicle. In the first stage the expected demand is considered, and in the second stage the above is substituted by the real demand realisation, once the customer has been reached.

The second limitation is concerning the length of the route including customer service time, to ensure that no driver is assigned a route longer than a shift's length. As stated above, we assume that the servicing time of each customer is given, as well as the time required to cover a certain distance. The route has a duration, given by the time required to service all customers and return to depot, and a cost, determined by the total distance traversed plus any incurred penalties. In the first stage, a penalty is incurred when either the duration limit or the capacity of the vehicle are exceeded on any route. However, routes which violate the capacity constraint are allowed as input for the second stage, as they might be feasible under the real demand realisations. On the other hand, before initiating the second stage, if there is a route which violates the duration constraint, all routes are adjusted in order for the route in question to become duration-feasible. In the second stage, and overall, capacity constraints are never exceeded via recourse actions, which change the duration of routes, so a penalty is set in place in the event the duration limit must be exceeded on some routes. Of course a route will exceed its duration limit if, and only if, there is not other way to service its customers.

2.2.4 First stage problem

The first stage problem is a deterministic CVRP. The demand of each customer is considered a given by taking for each customer the nominal value plus a predetermined percentage of the latter. For the remainder of this paper this percentage will be referred to as the customer deviation, *cd*, as it indicates how the customer is expected to deviate from its nominal value. The problem at hand then consists in determining a set of routes which minimise the costs related to the total distance traversed by the fleet of vehicles.

2.2.5 Second stage problem

A failure in a route occurs if a vehicle cannot fulfill a customer's demand. The second stage problem consists in determining which recourse action to adopt in order to amend or prevent a failure in a route. There are two recourse actions considered: a partial rescheduling of routes, taking a subset of vehicles into consideration, and a detour to depot such that a vehicle can restock to full capacity. Furthermore, there are two alternatives to when recourse actions can be considered: when a failure occurs or every time a vehicle reaches a customer and thus a new demand is known with certainty. The drawback of the second case is that a check needs to be performed as many times as there are customers however it prevents a vehicle from travelling to a customer if it has close to no capacity left and it is almost certain a failure would occur. Due to the uncertainty of the future demand while a route is ongoing it is very difficult to estimate the benefit of preventive recourse actions, therefore they are initially only considered upon failure, and when a failure will occur with certainty at the next customer.

2.2.5.1 The choice between recourse actions As mentioned in Section 2.2.2, we have two possible recourse actions: detour to depot and partial rescheduling. The detour to depot, as the name describes, amends a failure in a route by adding a visit at the depot and then returning to the failed customer. The advantage of this recourse action is that it is extremely fast, as it concerns only one route. The disadvantage however, is that it increases the length and thus the duration of the route, and depending on the location of the failed customer with respect to the depot, the increase in duration can be substantial.

The second recourse action considered is a partial rescheduling. A partial rescheduling was chosen over a full rescheduling as it is necessary that the recourse actions have very low computation time as these decisions are being made in real time. The advantage of this method is that it avoid idle trips to the depot, however the disadvantage is that it involves adapting multiple routes, not only the one where a failure occurs. Furthermore, moving the unserviced customers to a different route might merely create an additional failure if the route does not have sufficient capacity once all its customers' demand is known.

As stated previously, we assume there is a limit on the duration of a route. This refrains the method from always selecting the detour to depot recourse action. We therefore put in place some decision criteria on how to select which recourse action to undertake at each failure, considering the remaining length of the route when a failure occurs, the remaining expected demand of the route and its remaining capacity.

2.2.6 Buffer parameters

There are two buffer parameters in the first stage problem, the first concerns the capacity of vehicles and the second regards the customer deviation parameter. In the first stage problem a constrained VRP is solved taking a lower capacity, Q', than the real capacity Q. The difference is the buffer parameter, referred to here as Safety Space (SS), chosen for all vehicles to ensure there is some additional space in case multiple customers' demand realization are above the expected demand. The second buffer parameter regards each customer individually and, since it is a percentage, its impact is dependent on the nominal demand of each customer. The size and impact of the second buffer on a vehicle is dependent on the customers the route includes while the first buffer parameter is fixed for all vehicles.

The choice of these buffer parameters is made to minimize total costs. If the parameters are set too low, there could be many failures in the second stage leading to high penalty costs. On the other hand, if the buffer parameters are set too high, the first stage solution will be too conservative and thus incur in higher, unnecessary, costs. The values taken for these buffer parameters will be analysed in detail in the sensitivity analysis in Section 7.

2.3 Problem statement

Given a fleet of homogeneous vehicles with a certain capacity, Q, and a set of customers with uncertain demand, the objective is to determine a set of routes with duration lower than L, that service all customers at minimal cost. We assume full communication between vehicles and depot in real time and use this to improve the routes as information becomes available. The first stage solution needs not be feasible for all possible outcomes, as the routes can be amended in the second stage if failures occur. Due to the buffer parameters, it may be that a route appears infeasible in the first stage but it is not infeasible for the real vehicle capacity. The main objective of this thesis is to present an approach to solving routing problems with demand uncertainty which is tractable in real life scenarios due to its performance, its real-time adjustments to routes, and its low run time.

3 Literature review

3.1 The deterministic VRP

The Vehicle Routing Problem is a well known combinatorial optimization problem which concerns itself with finding an optimal set of routes to be used by a fleet of vehicles to fulfill a set of customer demands (Ben-Tal et al., 2009). The VRP is an NP-hard problem and thus finding optimal solutions becomes exponentially more difficult as the size of the problem increases. Since it was first proposed in 1959 by Dantzig and Ramser (1959), the VRP and its variants have been widely studied by many. From the capacitated VRP (CVRP), where vehicles' capacity is limited, to more complex variants including time windows, deadlines and heterogeneous vehicles. Toth and Vigo represent the pillars of this research and are credited with some of the most important work surrounding these problems. The work presented by Toth and Vigo (2002), overviews over 40 years of research since the introduction of the VRP by Dantzig and Ramser (1959), sharing the most prominent approaches to solving the classic VRP along with the CVRP, and other variants, analysing exact algorithms, heuristics and metaheuristics for all the variants of the problem. Many authors focused on solving a specific variant of the VRP such as Solomon and Desrosiers (1988), who presented a survey on the available algorithms for Time Window Constrained routing and scheduling problems. Baldacci et al. (2010), give an extensive overview of the exact algorithms to solve CVRP. The authors analysed exact methodologies such as Branch-and-cut methods, Branch-and-cut-and-price methods and set partitioning with additional cuts. Exact algorithms are however a small subset of methodologies to solve the VRP. Many applications turn to metaheuristics and heuristics as the computation times of the exact algorithms are quite unappealing in large problem instances. Common heuristics include sequential insertion heuristics and improvement heuristics, and common metaheuristics such as simulated and deterministic annealing, Tabu Search and genetic algorithms (Toth and Vigo, 2002). Bullnheimer et al. (1999) introduced an improved Ant colony based algorithm to solve a CVRP. The authors improved the run time and objective value of the Ant colony algorithms previously adopted to solve VRPs by considering only a subset of all vertices, a candidate list, when iteratively constructing and improving routes via a 2-opt heuristic (Bullnheimer et al., 1999). Subsequently, the proposed algorithm was compared with five commonly used metaheuristics on many different problem instances, and the authors found that the Tabu Search outperformed the adapted ant colony algorithm which in turn was superior to the simulated annealing and the neural network based algorithms. Golden et al. (1998), overview three common metaheuristics for solving VRPs: Tabu Search, simulated annealing and deterministic annealing. The methods are tested on various problem instances and compared on objective value and required computation time, finding that the best performing algorithm was, once again, the Tabu Search. Furthermore, the authors concluded that all of the above mentioned methods outperformed all neural network based approaches (Golden et al., 1998). Gendreau et al. (1994), introduce a Tabu Search based heuristic called Taburoute to solve VRPs with restrictions on vehicle capacity and length of route. During the iterations the authors allow for infeasible routes and include two penalty terms, one per restriction, in the objective function (Gendreau et al., 1994). Gendreau et al., credit their algorithm's performance to two heuristics designed by the authors for a TSP problem: the GENI, insertion heuristic, and the US, improvement method (Gendreau et al., 1994). The main feature of GENI is that it allows for a vertex to be inserted between two vertices which are not adjacent by considering all pairs of vertices within the route (Gendreau et al., 1992).

3.2 The non-deterministic VRP

All of the above mentioned work concerns the deterministic VRP, where all information regarding the problem instance is known. However, in realistic settings it is often the case that not all information is given with certainty. The non deterministic setting can be further split into stochastic and robust settings, where the difference is that in the former, the probability of occurrence around the uncertain parameter is known, while in the robust setting distributions remain unknown. Both the stochastic and robust VRP have the objective of finding the best possible set of routes which is feasible for all, or a percentage of, possible outcomes of the uncertain parameter. It is important to note that often, in robust problems, some statistical information regarding the unknown distribution is given or can be estimated, for instance from historical data. When this is the case a stochastic nature can be included and thus the methods described for the SVRP can be implemented in a robust setting (Ben-Tal et al., 2009). There is more literature available surrounding the SVRP, this is partly due to the fact that a stochastic approach is often preferred as numerical experiments have shown that robust approaches lead to higher objective values (Maggioni et al., 2015).

3.2.1 The SVRP

In 1996, Gendreau et al. (1996), review the available literature around SVRP and find that most methods approached the problem in one of two ways. The first method is the chance constrained programming (CCP), where one seeks a solution for which the probability of failure is below a predetermined threshold (Gendreau et al., 1996). A failure occurs when a vehicle cannot fulfil a customer's demand. The alternative approach widely adopted is a stochastic program with recourse (SPR), or two stage stochastic optimization, where a first stage decision is made to minimize expected costs of the second stage decisions where costs related to recourse actions are considered (Gendreau et al., 1996). Recourse actions are undertaken when a failure occurs in most approaches. However, there are also papers who adopted preventive recourse actions. The second stage costs are made up of the first stage cost, such as the cost associated with the routes, and the cost of recourse actions, such as a penalty in case of failure. According to Gendreau et al. (1996), the SPR is typically more difficult to solve than CCPs, however taking into account the cost of recourse actions in the objective function makes the latter more meaningful. A similar approach can be undertaken in the robust case, therefore the recourse actions adopted in the literature of both SVRP and RVRP are investigated. Yang et al. (2000) study a SVRP with demand uncertainty and adopt a SPR where the recourse action is a Detour-to-deposit (DTD) whenever a failure occurs. Another recourse action was undertaken by Ak and Erera (2007), who solved the problem by applying a paired locally approach where vehicles are paired a priori and then only one vehicle per pair can go back to depot to restock once the capacity is met. The authors reported an improvement in costs over the general DTD approach by minimizing the number of times a vehicle returns to depot, and thus its related cost. Laporte et al. (2002), follow a similar approach by implementing a method based on the L-shaped algorithm from Laporte and Louveaux (1993), and having a DTD recourse action when a failure occurs. The authors though struggle to find optimal solutions with more than ten customers when the expected filling rate of the vehicles is high (Laporte et al., 2002). According to Oyola et al. (2018), the most common recourse actions adopted in stochastic VRP papers are: DTD, preventive DTD, penalty in case of failure, and re-optimization of a single route before it applies DTD. Finally, a last method evaluated is that of Secomandi and Margot (2009), who apply partial reoptimization on a subset of all possible Markov decision process states and solves for an optimal policy. The main drawback of this approach is that it necessitates of a large amount of memory to solve even a small instance.

3.2.2 The RVRP

Now the literature surrounding RVRP is discussed. All papers presented solve a RVRP with demand uncertainty. The set, which includes all possible outcomes of the uncertain parameter, is called the uncertainty set. The uncertainty can be considered as a whole or a subset can be taken to include only a percentage of the possible outcomes. The choice between the two approaches is based on the assumptions that can be made around the uncertainty set and on how conservative a solution is required to be. The first approach was taken by Sungur et al. (2008), who study an approach which identifies an optimal set of routes feasible for all possible demand outcomes. The authors make two crucial assumptions, i) that there exists a solution feasible for all outcomes, and ii) that all customers could require their highest possible demand simultaneously. The approach presented would lead to an over-conservative solution. As it can often be assumed that not all customers will require the highest possible demand simultaneously, a solution which is very conservative is not always necessary. More complex is the method proposed by Lee et al. (2012), who combine the column generation algorithm with branch and price to solve a VRP with deadline under demand and travel time uncertainty. Finally, the last paper reviewed is the one by Gounaris et al. (2013). The authors undertake two approaches depending on the shape of the uncertainty set. If the uncertainty set is rectangular, the problem reduces to a deterministic CVRP, and if the uncertainty set is not rectangular, the authors develop robust counterparts of well known methods for solving deterministic CVRPs, such as the Miller-Tucker-Zemlin formulation and Two-Index Vehicle Flow Formulation (Gounaris et al., 2013). Few papers have adopted a robust approach to solve a RVRP as a worst case scenario often results in an over-conservative solution and in two stage optimization it is difficult to incorporate costs of more complex recourse actions. This paper aims to integrate information from the best known methodologies for solving the different variants of the VRP in an approach that would be adopted in many realistic settings.

4 Methodology

As mentioned in Section 2.2, there are two main phases: the first, where the routes are defined, and the second, where the routes' feasibility is tested via a Monte Carlo simulation in which recourse actions are undertaken to avoid and amend failures. In this section we describe in detail how the problem is formulated and how the first and second stage problems are going to be solved.

4.1 Parameters

The problem is structured as follows: there are n customer and m vehicles given. For each customer, $i \in I$, we are given its demand range $[a_i, b_i]$, from which we determine its nominal demand value q_i, and its required service time, s_i. For each vehicle we are given a capacity limit, Q, and the limit on the route's duration, L, set to a 9-hour work shift.

Furthermore we set two buffer parameters: the customer deviation, cd_i, and the safety space, SS. The customer deviation shows how we expect the customer to deviate from its nominal value or, alternatively, how safe we want to be in considering the customer's demand. Given the customer deviation, we determine the anticipated demand of each customer, $q'_i = q_i \cdot (1 + cd_i)$, where the customer deviation parameter takes on a value between zero and one. The second buffer parameter, the safety space, concerns each vehicle. The safety space is a value between zero and one which defines how much space on the vehicle to leave idle as a safety measure for uncertain demand. From the safety space we determine the capacity of the vehicles considered in the first stage, $Q' = Q \cdot (1 - SS)$. When the buffer parameters are set to zero, we take on a less conservative approach and risk incurring more failures, however this can also lead to lower costs. This will be evaluated in the sensitivity analysis in Section 7.

4.2 Problem structure

Given a list of customers, their location, their nominal demand and their service time, and given a fleet of vehicles with a certain capacity and duration limit, we construct our problem.

The buffer parameters, SS and cd_i, introduced in Section 2.2.6, are chosen such that the first stage solution is a good starting point for the second stage. It is important to find the correct values for these buffer parameters via a sensitivity analysis (see Section 7). If the buffer parameters are set too high, the first stage will output very conservative solutions, and few failures will occur in the second stage, however may require an additional vehicle for it to reach a feasible solution. If, to the contrary, the buffer parameters are set too low and the demand outcome is above the expected demand, the routes will incur many failures. As the second stage problem is in real time we want to avoid an excessive number of failures.

The last parameter is the neighbourhood distance d, this is not to be confused with the neighbourhoods of the Tabu Search later discussed. The neighbourhood distance is used to define a list of neighbours for each customer. This list is later adopted in the partial route rescheduling method to define what routes should be considered. When a failure occurs at a certain customer, instead of evaluating all routes, only the ones containing a neighbour of the failed-to-service customer are considered. If a customer's location is such that it has insufficient neighbours within the neighbourhood distance d, the distance is increased, for the customer in question solely, until a predefined sufficient number of neighbours is reached. The reason for this is that, if a failure occurs at a customer which has less than a certain amount of neighbours, the algorithm will encounter difficulty in finding a route which can service the customer instead and more often result in a detour to depot.

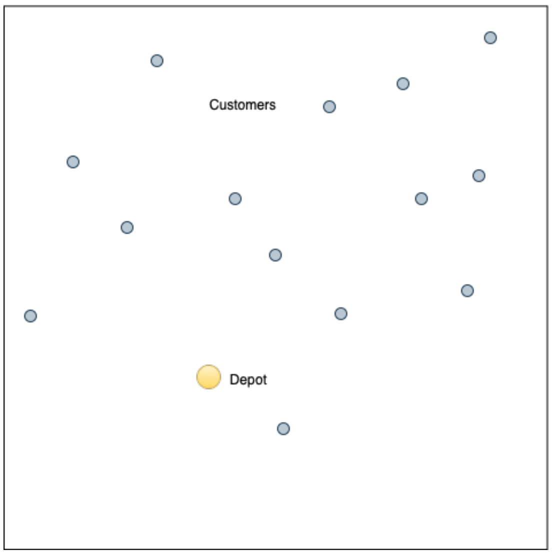
4.3 First stage

The problem in the first stage is a deterministic constrained VRP with homogeneous vehicles. After reviewing the existing literature, the Tabu Search is considered the most suitable approach to solve the first stage problem. Tabu Search is a metaheuristic and more specifically it is an improvement method as it searches through successive neighbours of a solution, allowing the objective value to deteriorate in order to avoid local minima (Gendreau et al., 1994). As mentioned above, Tabu Search is an improvement method thus it does not construct a solution but improves an existing one. Therefore, before applying Tabu Search an initial solution needs to be determined, by means of a constructive heuristic.

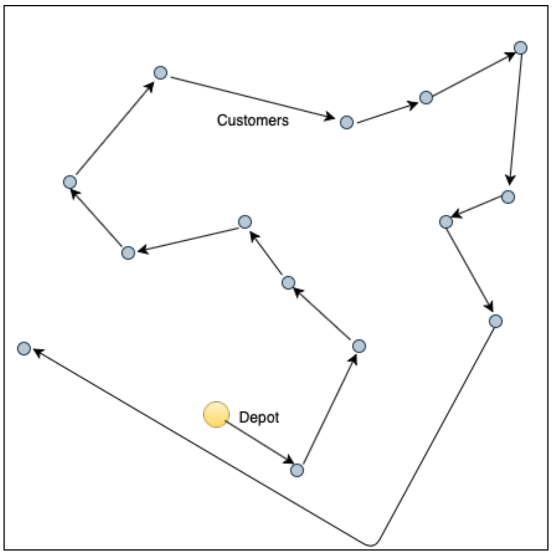
In the next sections we will cover the constructive heuristic, the Tabu Search in great detail, with all its parameters and discuss how it was adapted to suit the problem at hand. Finally, we discuss how we transition from first stage routes to the input solution for the second stage, via a final improvement heuristic.

4.3.1 Adapted Nearest Neighbourhood

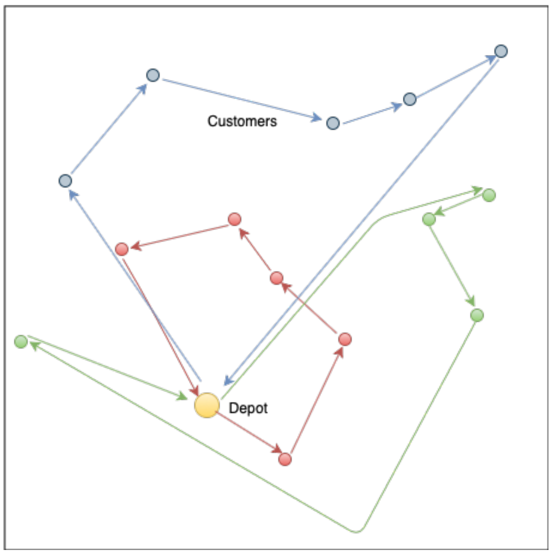
The constructive heuristic chosen to determine the initial set of routes, is an adapted Nearest Neighbourhood heuristic. The constructive heuristic begins with taking the initial distribution of the depot and customers, an example is given in Figure 1a. Secondly, starting from the depot a route is constructed by iteratively adding the closest customer to the current end of the route. The solution at this step resembles an initial solution for a TSP problem, as seen in Figure 1b. Thirdly, before splitting the route into as many smaller routes as vehicles available, the latter is improved via a quick local search heuristic which switches the position of two customers within the route to improve on the total length. Lastly, the single route is divided into as many smaller routes as there are vehicles. The number of customers, n, is divided by the number of vehicles, m, and the value obtained, v, is rounded up by excess and corresponds to the number of customers on each route, except the last route in case of an unequally divisible number of customers. After m segments with v customers are formed, the first and last node of each segment are connected to the depot to form m routes, as seen in Figure 1c, which gives us the initial routes to be improved on by the Tabu Search. The set of routes constructed need not be feasible, because the constraints of the vehicles are not considered in this stage.



(a) Depot and customers distribution.



(b) TSP route.



(c) Final routes.

Figure 1: The three steps of the constructive heuristic.

4.3.2 Tabu Search

Tabu Search meta-heuristics iteratively search the neighbourhood of the current solution for an alternative, not necessarily improved solution. One of the uniquenesses of the Tabu Search is that it includes a Tabu list, a list of solutions that are not allowed for a certain amount of iterations, to avoid the algorithm from choosing a preselected solution for a number of iterations. The Tabu list further helps the algorithm's pursuit of escaping local minima.

To initiate a Tabu Search method it is first necessary to define the neighbourhoods. Different neighbourhoods are selected as, if no alternative solution is found, the algorithm moves to the subsequent neighbourhood to search for solutions. During the Tabu Search, we allow for infeasible solutions to be considered, by including a penalty term in the objective function for when constraints are violated. Including infeasible solutions is common when applying local search heuristics as it can aid in evading local minima. The penalty incurred is proportional to the degree of violation of the constraint. This is done such that, between two solutions with equal number of violated constraints, the best solution, equivalently the closest to being a feasible solution in future iterations, is chosen. Furthermore the penalty incurred is higher when the duration of the route violates the feasibility constraint as there is no uncertainty that this will be a constraint violated in the second stage as well. Contrary to the capacity constraint for which, for different demand outcomes and due to the safety space, SS, and customer deviation, cd_i, buffer parameters, an infeasible route in the first stage may be feasible in the second stage.

4.3.2.1 Neighbourhoods

1. **Switch order of customers within a route**
2. **Move one customer from one route to another**
3. **Exchange two customers between two routes**
4. **Exchange two customers from one route with one customer from a different route**

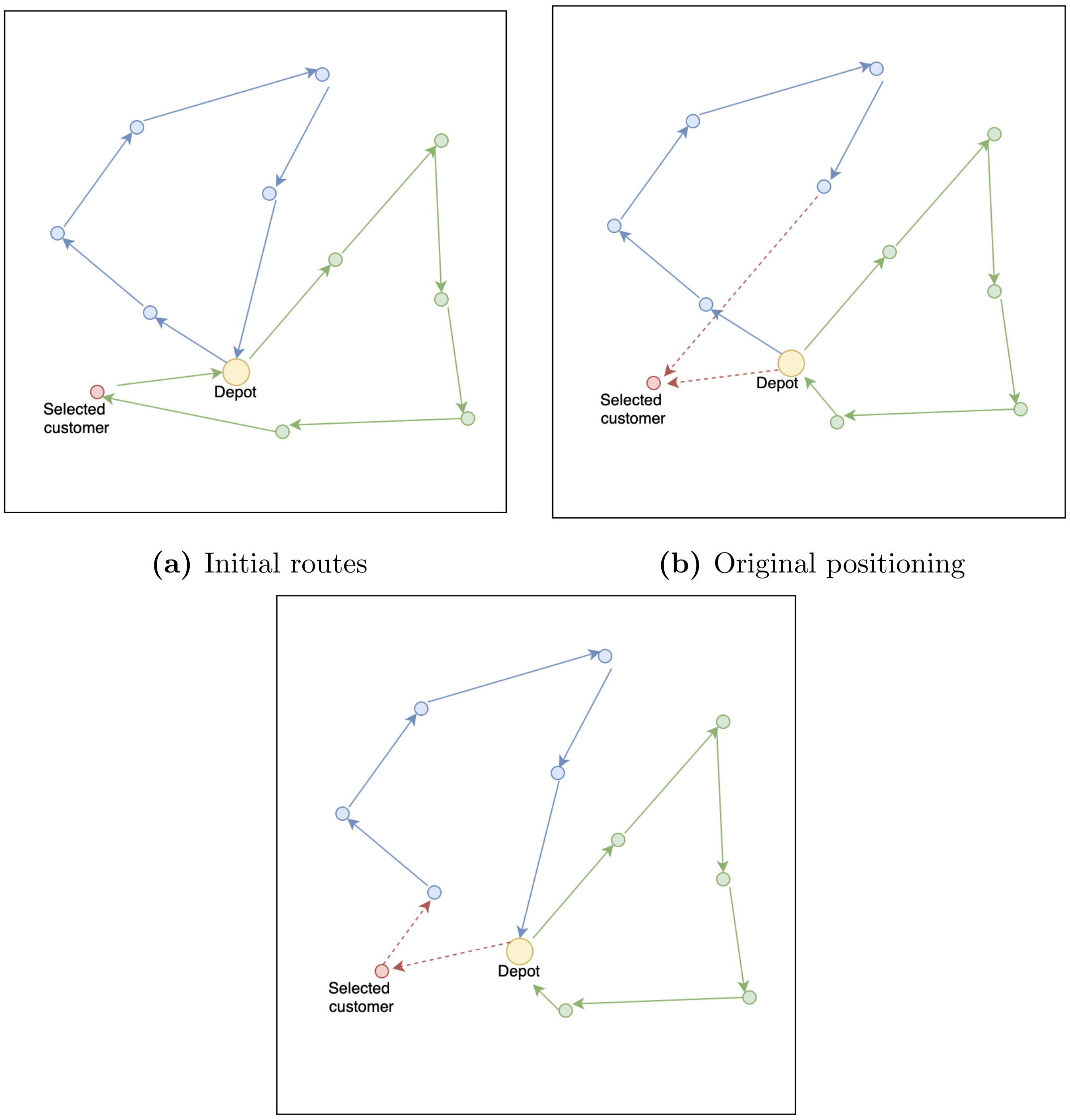
The first neighbourhood is unique as it improves the routes separately by selecting two customers within a route and swapping their position. Furthermore, as opposed to the other neighbourhoods where the best solution is outputted once all possible movements have been checked and the best one is incorporated, the first neighbourhood continues to implement multiple changes until it reaches a time limit. The algorithm implements any change in the route that leads to an improvement, regardless of it being the best possible change in the route, until the predetermined time limit on this neighbourhood is reached.

The reason behind this change in the first neighbourhood is that, checking each movement requires close to zero computation time, thus in the same time the algorithm checks all possible movements of any other neighbourhood it can loop through all possible changes of the first neighbourhood multiple times. When a movement that improves the current length of the route occurs, the swap between customers is implemented and the process continues, looping through the routes, up to the time limit on this neighbourhood is reached. The reasoning behind this is that this neighbourhood only affects one route at a time, thus any beneficial movement can be implemented without it hindering the search of the best solution within the neighbourhood.

In all other neighbourhoods, one, or more, customers are moved to a different route in order to search for an improved solution. Different insertion criteria are evaluated, such as the one adopted by Gendreau et al. (1994), in which a customer can be added to a route only if the latter contains at least one of the neighbouring customers of the customer in question. The benefit of this approach is that it limits the number of moves to evaluate within a neighbourhood and this is more tractable from a computation time stand point. However, limiting the size of the neighbourhood appeared to hinder the search for a good solution and thus was not implemented.

Whenever a customer is inserted in a new route, it is inserted in its optimal position in the new route. This is essential in finding a low cost solution as merely switching the customers in their respective position can lead to a costly route due to incorrect positioning of the new customer which can lead to erroneously disregarding good routes. An example is given in Figure 2, where an iteration of the second neighbourhood is shown. Figure 2a shows the initial distribution of customers within two routes. The customer in red, is the customer selected and is moved to the other route. In Figure 2b, the customer is inserted at the same position as in its original route and in Figure 2c, the customer is inserted at its optimal position in the new route. When comparing the costs of the original route, seen in Figure 2a, with the new route, it is clear how the positioning of the new customer is key to selecting the best neighbourhood solution.

Whenever a solution that meets the acceptance criteria is found, it is set as the current solution of the Tabu Search, and the process is resumed from the first neighbourhood. In the scenario that a neighbourhood's best solution does not meet the acceptance criteria, the subsequent neighbourhood is searched.



(a) Initial routes **(b)** Original positioning

(c) Optimal positioning

Figure 2: Determining optimal position to insert customer.

4.3.2.2 Tabu list For the Tabu list, a max length is predetermined at the beginning of the algorithm. As described by Salhi (2002), the length of the Tabu list can lead to different results. After each neighbourhood is evaluated, if the solution reported fulfills the acceptance criteria, it is added to the Tabu list. If the length of the Tabu list is above the capacity, the first in the list is removed, as it has been in the list for as many iterations as the capacity of the list. In this thesis we chose a maximum length for the tabu list of eight solutions, where a solution is a set of routes.

4.3.2.3 Acceptance criteria As mentioned previously, one of the benefits and uniquenesses of the Tabu Search is that it allows for a worse solution to be considered in order to better avoid local minima. Initially the acceptance criterion is such that a solution is accepted if and only if its cost is lower than the cost of the current solution. If all neighbourhoods have been evaluated and no viable solution is found, the acceptance criterion is slightly lowered, such that worse solutions are considered.

4.3.2.4 Stopping criteria There are different stopping criteria adopted commonly when applying Tabu Search metaheuristic such as a time limit or when a percentage of improvement from the initial solution has been reached. We assume that in a real life setting, the limit of the algorithm would be time. Thus we implement a time limit on the first stage. When the upper bound is reached, the algorithm stops and outputs the current best solution. We assume the time limit can be relatively high as the first stage is solved at the beginning of the day, before the vehicles are dispatched, therefore we take a time limit between 10 and 15 minutes, depending on the problem size, to find a good set of routes. The higher the limit is, the better the solution outputted by the Tabu Search.

4.3.3 Duration feasibility check

As previously mentioned, a higher penalty is incurred on a route if its duration exceeds the upper-bound, L, in comparison to an infeasible solution due to the route being assigned a total demand quantity higher than the capacity of a vehicle, Q'. The higher penalty is in place such that in the first stage a route which exceeds capacity limit rather than one which exceeds the duration limit is selected, as the quantity is uncertain and therefore an infeasible route may then become feasible with the real demand realizations while the duration of a route faces no uncertainty. The first stage routes are therefore unlikely to present routes which exceed duration but it may occur.

A route whose duration exceeds the upper limit L, will always be infeasible unless one or more customers are removed from the route. As we cannot have infeasible routes in the second stage and there are no recourse actions for lack of duration-feasibility, we shall ensure no route is duration-infeasible as we transition from first stage to second stage. Therefore, after the Tabu Search outputs its final routes, a final method is applied such that all routes have a feasible duration. This method checks each route, if its duration exceeds the upper-bound, it is added to a subset of routes. Subsequently, for each route in the subset, a customer is removed and inserted to a route, not in the subset, which can endure the additional duration.

4.4 Second stage

In the second stage we test the routes of the first stage via a Monte Carlo simulation which randomly selects a demand outcome for each customer within its demand range, $[a_i, b_i]$. As described in Section 2.2.1, to ensure all customers cannot take their highest possible demand outcome simultaneously, there is a limit on the cumulative demand.

Given that in the first stage we calculated routes given the nominal value plus a fixed customer deviation, and not the real demand, the routes' total demand is expected to deviate from what was assumed in the first stage. Therefore there will be customers whose demand is higher than that accounted for and some for which it is lower. Subsequently, there will be routes with higher total demand, and some with lower total demand, than expected. This deviation will most likely result in some failures, i.e. when a vehicle has insufficient capacity to service its current customer.

When a failure occurs, a real time decision is made on how to amend the failure. The algorithm evaluates two recourse actions: a detour to depot, such that the vehicle can fully restock and return to the unserviced customer, and a partial rescheduling action, where the failed customer can be reassigned to a different route. As mentioned above, the latter is a partial rescheduling, as only routes within a certain proximity from the failed customer are considered. The reason for restricting the set of routes considered is to decrease the required run time while not hindering the result of the search too much by defining a subset of routes which are most likely to be chosen.

The first stage methods are run at the start of the day, before dispatch, therefore there is a higher time limit than for the second stage. The second stage methodologies are considered multiple times during each simulation and require making a fast decision to amend the failure as vehicles are stalling, waiting for updates on their routes. Therefore the time limit is much more restrictive in this stage, as to minimize the time vehicles are idle.

As in many real life scenarios, the vehicles have a limit, L, on how long their shift can be. This prevents the problem from always having a feasible solution under any outcome, as the vehicle cannot return to depot at every failure, thus adding an obstacle to the problem which is often encountered in real life scenarios. Furthermore returning to depot is often a less efficient use of the vehicle than partially rescheduling the route, as it requires a futile trip, and therefore time in which the vehicle is not servicing customers. We therefore favor the partial rescheduling when possible.

The first recourse action is a detour to depot, this action is a simple way of adjusting a route to compensate for a failure. If a vehicle reaches a customer but does not have the capacity to meet the customer's demand, which is only known upon arrival, it may return to depot, restock and return to the customer who was left unserviced. This action affects solely the route that encountered a failure and additional costs are encountered relative to the detour. The distance traversed to reach, and return from, the depot are taken into account in the shift of the vehicle limited by L, as well as the time to restock. Furthermore preventive detours are considered in order to minimise the distance traversed. This will be discussed more specifically in the next sections.

The second recourse action is the partial rescheduling. When a failure occurs on a route, a subset of all routes is considered. All routes, which include a neighbour, specifically a customer which is close in location to the customer where the failure occurred, form a subset of options. Subsequently, by means of a heuristic, routes within the subset are adjusted such that the failed customer is serviced by a different route. It is important to take into account at what point of its route each vehicle is, and what its current capacity is in comparison to what was expected at that point, i.e. if its customers have been above or below the expected demand. The latter can play an important role in future failures, if the failed customer is relocated to a vehicle which is below the expected capacity, it is less likely to incur a failure on future customers than a vehicle which is above expected capacity.

In the next sections the second stage methods are described in detail. First the selection criteria between recourse actions are depicted, then the detour to depot action is described, subsequently the partial rescheduling is outlined and finally we consider preventive recourse actions.

4.4.1 Selection criteria for recourse actions

The criteria adopted to choose between the recourse actions were introduced in Section 2.2.5.1. Due to the limit on the duration of a route and the capacity of a vehicle, there is no one recourse action that is suitable in all failure scenarios. We therefore define some criteria that can be checked instantly in order to select the best recourse action to amend a current failure.

The features considered when a failure occurs on a route are: (i) the distance from the failed customer to the depot, (ii) the position of the customer within the route, (iii) the remaining capacity of the route, and (iv) the total expected remaining demand of the route. The first feature checks the distance from the depot. If the route's duration plus the time required to reach, restock and return from the depot, exceeds the upper limit L, then the only viable recourse action is the partial rescheduling. The second characteristic considered is the position within the route. If a failure occurs within the first 20% of the duration of the route, the recourse action undertaken is a detour to depot. The reason for this is that, given that a failure occurs so early in the route, it is very likely that a second failure will occur if only one customer is removed via partial rescheduling. The third and fourth features, namely the remaining capacity of the vehicle and the expected remaining demand, are considered together for the third and final criterion. The third and final criterion states that: if the remaining expected demand of the route minus the demand of the customer awaiting to be serviced, exceeds the vehicle's remaining capacity, the chosen recourse action is the detour to depot as an additional failure is very likely to occur. If, on the other hand, a vehicle's remaining expected demand exceeds the remaining capacity by less than the demand of the customer awaiting service, the recourse action undertaken is the partial rescheduling. The decision process is depicted in the flow diagram in Figure 3.

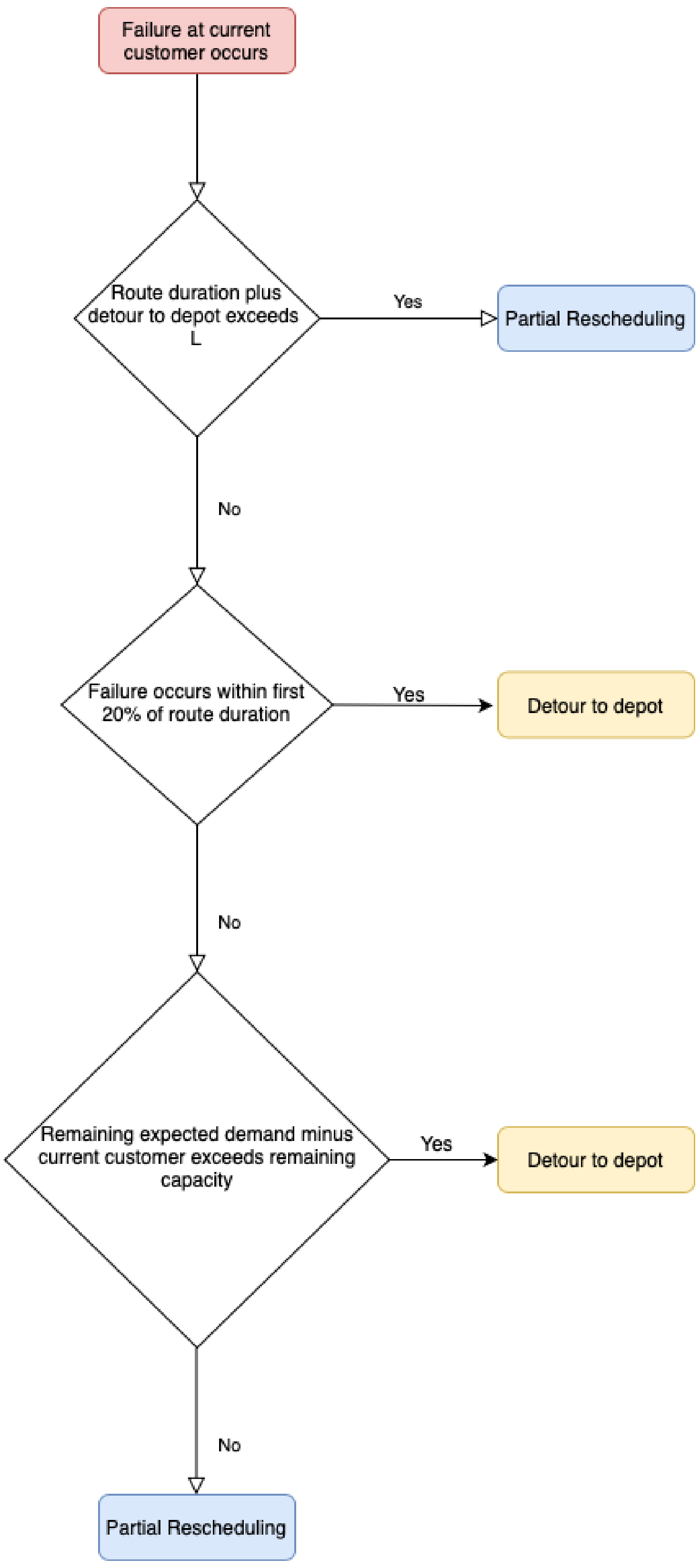

Failure at current customer occurs
Route duration plus detour to depot exceeds L
Yes
Partial Rescheduling
No
Failure occurs within first 20% of route duration
Yes
Detour to depot
No
Remaining expected demand minus current customer exceeds remaining capacity
Yes
Detour to depot
No
Partial Rescheduling

Figure 3: Flow diagram on recourse action selection criteria

4.4.2 Detour to depot

In this section we discuss the detour to depot recourse action. When this action is undertaken to amend a failure, the vehicle in consideration returns to the depot, restocks, and then returns to the customer, as seen in Figure 4. The advantage of this recourse action is that its required computation time is close to zero, as it involves solely the route where a failure occurred and simply requires the addition of two customers to the route, namely the depot and the customer at which the vehicle is currently located.

However, the disadvantage of this method is that it increases the duration of the route by two times the time required to cover the distance between the failed-to-service customer and the depot, plus the time spent at the depot. This can lead to a substantial increase in the route's length and thus duration. We assume that a visit to a customer where a failure occurs requires zero service time as it considers solely the time required for the customer to disclose its real demand, which can be assumed to be close to irrelevant. The customer's service time is considered when the vehicle returns and can service the previously failed-to-service customer. Additionally, we assume that a visit to the depot also requires a certain time for the vehicle to restock. In our problem we set the service time of the depot equal to that of visiting any customer. However, as service times are customer attributes, this is easily adaptable, as is the homogeneity of the service times among customers.

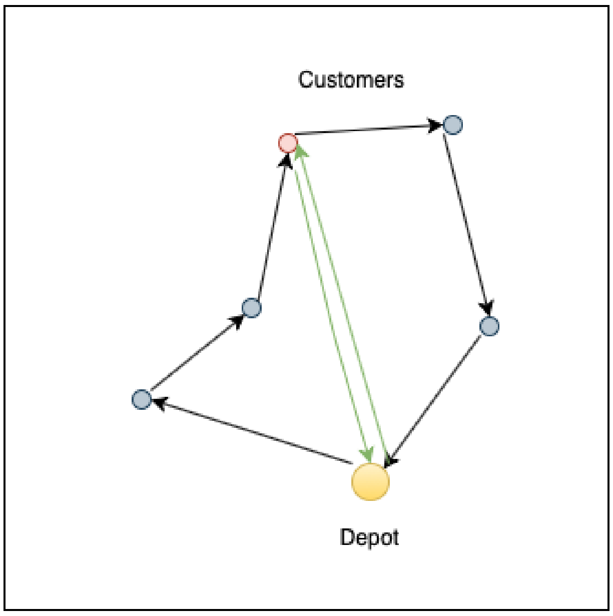


Figure 4: Detour to depot

4.4.3 Partial route rescheduling

The second recourse action that can be adopted to amend a failure is a partial rescheduling of the routes. Given a failure occurs on a route, and the recourse action adopted is chosen to be the partial rescheduling, a rerouting procedure begins to move the current unserviced customer to a route that has sufficient capacity. In order to save on computation time, not all routes are considered. A subset of possible routes is constructed by selecting all the routes which include at least one customer who is a neighbour of the failed-to-service customer. A failure can happen at any point of the route but is more likely to occur towards the end of the route, as the vehicle is expected to have less remaining capacity than at the beginning of its route. Given a customer has j neighbours within the network, we expect most of the neighbours to be within the same route. Furthermore, given a failure occurs in the second half of the route, some of the neighbours of the failed-to-service customer will most likely have already been visited. We decide to ignore whether a neighbour has been visited or not before adding the route to the subset, as we can assume, if a neighbour was included in a route, the latter can still be a viable option for insertion. Additionally, not considering the routes where a neighbouring customer is present but has already been visited would reduce the subset and increase the probability of the algorithm being unsuccessful in finding a route to insert the customer under consideration. Therefore it is important to choose a large enough value j of neighbours for each customer such that there is almost always a route in the constructed subset. In our problem we set a minimum of eight customers. The neighbours are found based on distance, a circle is created around each customer and all customers within the circle are set as the neighbours of the latter. The radius if a customer's circle is increased until the minimum amount of eight neighbours is found. Thus if there are many customers within a certain small distance of the customer in question, the latter might have more than eight neighbours. The neighbour search process is depicted below in Figure 5

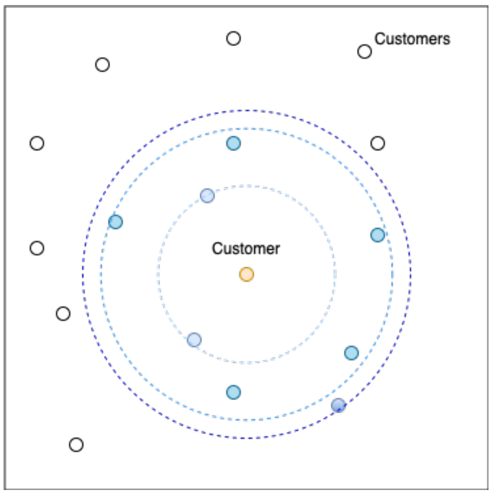
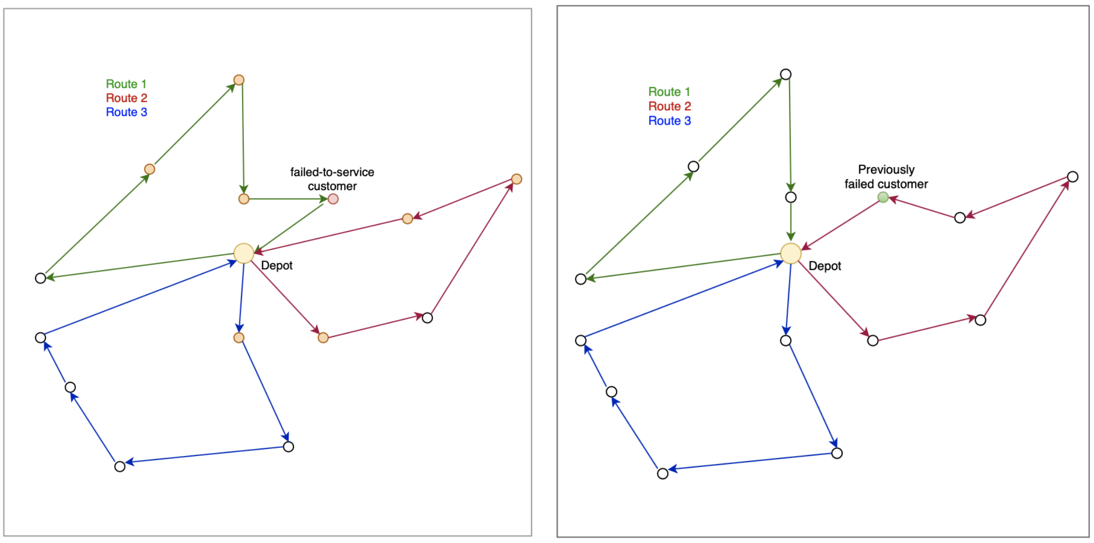


Figure 5: Neighbour search

Once the subset of options has been constructed, each option is checked to see if it remains feasible if the new customer is inserted. To do so we check the remaining capacity of the vehicle, the remaining expected demand of the vehicle and its expected duration, which does not take into account any possible future failures. If the vehicle can service the customer in question without violating any constraints, the vehicle remains a viable option. Then the customer is added to the cheapest feasible route, in its optimal position, starting from the position of the first unserviced customer on the route. If no route within the subset is a viable option, the algorithm begins to search among the other routes. If no route is still a viable solution, or the algorithm has been running for too long, the method resorts to implementing the detour to depot recourse action. In the unlikely scenario in which both recourse actions are unsuccessful in amending the failure, a penalty is given to the vehicle in question as a detour to depot action is selected leading to a violation of the shift length. This is a viable option in a realistic setting where a penalty is incurred by the company in the form of overtime pay.



(a) Initial routes with failure on Route 1. **(b)** After the partial rescheduling action.

Figure 6: An instance of partial rescheduling.

An example of the partial rescheduling being implemented is depicted in Figures 6a and 6b. The first figure represents a route which incurs a failure at a certain customer, shown by the customer in red. A subset is created with the routes that include at least one neighbour of the failed customer, here depicted in orange. All routes appear to contain at least one neighbour, then each route's feasibility is analysed if the failed customer were to be added to the route, and the most efficient option is found. In Figure 6b, the previously failed-to-service customer has been added to Route 2, in its optimal position, towards the end of Route 2, as the failure occurred at the end of Route 1.

4.4.4 Preventive recourse actions

The above recourse actions are considered only when a failure occurs and the recourse actions are necessary to amend the situation. However it is easy to imagine that a preventive recourse action could be more efficient as it would avoid an idle trip to a customer that cannot be serviced. The uncertainty of the demands of not-yet-visited customers hinders the ease of selecting when to adopt preventive recourse actions as the chance of future failures is also uncertain. The first decision we made is to only consider detour to depot as possible preventive recourse actions. The reason is that this action only alters the route in question, without bringing changes to other routes. As the future demands are uncertain, if a preventive partial rescheduling were adopted, a customer could be added to a route that is about to incur a failure, thus leading to multiple consecutive failures.

The second decision to make is when to adopt a preventive detour to depot, given there can be no certainty of a subsequent failure. During the Monte Carlo simulation, vehicles are continuously monitored for failures. Furthermore, after each successfully serviced customer a vehicle has visited or is visiting, a preventive recourse action is considered for the next customer. When a customer is successfully being serviced, the following customer is taken into consideration for a possible preventive recourse action. In our problem structure we assume that for each customer, we are given a lower bound and an upper bound on their demand, $[a_i, b_i]$, from which we deduce the nominal demand value and the expected demand value. If the ranges were not given, a lower bound could be determined by the maximum deviation a customer is expected to have and their nominal value. Given the lower bound of the subsequent customer's demand, a_i, and the remaining capacity of the vehicle after servicing its current customer, we can deduce whether a failure would occur with certainty. If the customer's minimum possible demand exceeds the remaining capacity of the vehicle, a preventive detour to depot is adopted, as illustrated in Figure 7. Subsequently, if the sum of minimum possible demand of all unserviced customers on route exceeds the remaining capacity of the vehicle and the vehicle is within its first half of the route, a preventive detour to depot is implemented.

The benefit of implementing preventive recourse actions is evident, and can be seen by comparing Figures 4 and 7. Preventive recourse actions could lead to more efficient routes if considered more often, but at the risk of implementing actions to avoid a failure that would not occur regardless. Thus, in this thesis, we focus on implementing a preventive detour to deposit only when a subsequent failure is certain to occur.

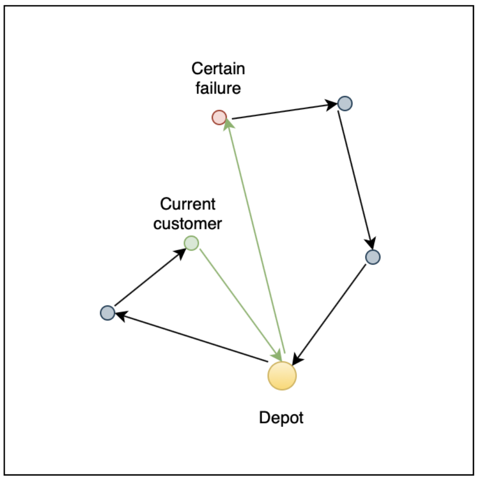


Figure 7: Preventive detour to depot

4.5 Monte Carlo simulation

As mentioned when introducing the second stage problem, the routes outputted by the first stage methods, are tested for feasibility via a Monte Carlo simulation in which random demand vectors are generated. The demand vectors are generated by taking a random demand value for each customer within its possible demand range, $[a_i, b_i]$. If the cumulative demand over all customers exceeds the predetermined value B, from Section 2.2.1, the demand vector is regenerated until its sum does not exceed the limit B. The worst-case scenario always occurs on a vector which does not present the highest cumulative demand tested, hence we cannot generate worst-case demand vectors purposely. We instead assume that over the 100 iterations tested, the worst-case scenario obtained is close to, if not already, the real worst-case scenario.

During the simulation, the vehicles' capacity is monitored for current or future failures as described in the above sections. When a failure occurs or is certain to occur, a recourse action is implemented. Given that all customers have a positive service time, as well a positive required time to travel from one customer to another, we do not implement a continuous simulation but rather a discrete one. A discrete simulation refrains the program from continuously checking the current situation of all vehicles. We refrain from this and implement a discrete time simulation which checks every minute the current position of a vehicle and it's last serviced or currently visiting customer.

If a vehicle is still busy servicing the same customer as in the previous check, it is unnecessary to perform the same checks twice and thus the simulation program quickly moves to the next vehicle. If a customer is in a new position compared to that of the previous iteration, however is not visiting any customer as it is on route between destinations, the program moves on to the next vehicle. If, on the other hand, a customer is visiting a different customer, first a failure check is performed, to verify whether the vehicle has sufficient capacity to service the customer. If there is a failure, a recourse action is undertaken, otherwise the next customer is evaluated to analyse whether a preventive recourse action is beneficial.

The simulation continues until all vehicles have returned to depot. If for some larger instances there were to be no feasible way of amending a failure, a penalty on the length of the route is set, as to ensure all customers are serviced by the end of the day, with no exception. Alternatively a penalty could be set for each lost customer, as to ensure that all vehicles return to depot before the end of the shift.

5 Data

5.1 Fisher instance set

The approach presented in this thesis is tested on a CVRP set, namely the F-instances, introduced by Fisher in (1994). There are many different sets of instances available, each differing in problem size, demand values and proximity among customers. Some sets were criticised for seeming too artificial or for locating the customers within an unrealistic, geometrical shape (Uchoa et al., 2017). The Fisher set presents three instances, ranging from 44 to 134 customers, and with 4 to 7 vehicles. The instances are constructed from real-world data, two instances from data collected in a grocery store in Ontario and one from the tire delivery by Exxon (Fisher, 1994). The F-instances were selected over the other available sets for three reasons: i) it is constructed around real data, ii) all test instances have been solved optimally and thus we have an optimal value to compare our first stage results to, and iii) it provides the opportunity to test the method of this thesis on an instance with over 100 customers, as the majority of the test instances are small problems which do not accurately represent real life scenarios.

5.2 Problem instance

For each problem instance in the set we are given n customers, m vehicles each with capacity Q. For each customer, $i \in I$, we are given a demand, q_i, and its location, given by two coordinates (x_i, y_i). The depot is located at (0,0) and the distances between locations are calculated by euclidean distance. For each customer we take the given demand, q_i, as its nominal demand value and the range of possible demand outcomes, [a_i , b_i], where :

$$a_i = q_i \times 0.8, \tag{3}$$

$$b_i = q_i \times 1.2, \tag{4}$$

and the values are then rounded down to the nearest integer. Thus there is a maximum of 20% deviation from its nominal value. The values [a_i , b_i] are only necessary for the simulation, such that the demand realization for each customer is within the predefined range. As stated in Section 2, we do not allow for all customers to reach their worst realisation, but limit the total possible demand. Only the nominal value, q_i, is known and taken into account when determining a set of routes in the first stage of the problem.

6 Results

In the following section we discuss the results found when testing our approach on real life problem instances. The methodology is tested on three different problem instances, which vary on number of customers, number of vehicles and capacity of the latter. First the results of each individual instance are discussed, then the performance of the algorithm across different instances is analysed. The first stage results of each instance, are compared with the optimal value of the deterministic version of our problem. For this analysis all buffer parameters are set to zero, such that the algorithm takes no safety measures towards creating a safe solution for the second stage while determining the cheapest set of routes. That is, it does not account for safety space on the vehicle nor does it account for a higher customer demand than expected. Furthermore, the results of the optimal solution are obtained by setting the upper bound on capacity as the only constraint. We disregard the duration constraint when reporting the first stage results as this limitation was not included when calculating the optimal routes and including it modifies the nature of the problem, thus yielding a solution which would not be comparable to the optimal.

Subsequently the results of the overall algorithm are discussed, setting the buffer parameters to their predetermined values which are later specified and incorporating the duration constraint. Including the duration constraint already results in a change in the solutions obtained, which present a higher costs. The routes found when the duration constraint is not in place, may be infeasible when the constraint is included. Due to the penalty incurred when constraints are violated, the routes resulting in the first stage solution are likely to be different when the duration constraint is included, resulting in a higher objective value. Furthermore, depending on the values chosen for the buffer parameters, the demand considered per customer can be higher than their nominal value, and the vehicles' capacity in the first stage can be considered to be lower than the real capacity. As this restricts the problem, it is clear how the results of the first stage, with the added parameters and limitations, will most likely be higher than the result obtained in the deterministic variant.

The results of the overall algorithm cannot be compared to the optimal results of the deterministic setting, as the uncertainty in the demand, and the possibility for higher demand values, makes the comparison meaningless. Therefore we will discuss the overall performance, the increase in costs due to failures in the second stage, and their respective recourse actions. The results of the second stage are obtained by running the simulation 100 times, each with different random demand realizations for the customers. The worst outcome is then taken as the basis for evaluating the performance of the applied methodologies, in order to maintain a robust approach. Finally, we compare the performance of the first stage and overall methods across the different instances.

The buffer parameters are chosen as follows: the safety space on a vehicle, SS, is set to 10%, leading to a vehicle capacity of 90% of the real capacity. Recall that the adapted capacity, Q', is only taken as the vehicle's capacity in the first stage. The customer deviation, cd_i, is set to 0, thus, given the nominal value of customer i, q_i, the expected demand is then equal to the nominal value (q_i), for each customer. The values for the buffer parameters were selected such that they provide the best performing algorithm over all instances. Different parameters are later discussed in the sensitivity analysis in Section 7. The service time of all customers, including the depot, when a detour to depot is undertaken, is set to 20 minutes. Given the difference in size between problem instances, the time limit on the Tabu Search is increased to respect the increase in possible solutions.

6.1 Instance 1

The first, and smallest, instance tested is constructed around 44 customers plus the depot, with a cumulative demand of 7220. There are four vehicles available, each with a capacity of 2010. The time limit for the overall first stage methods, including constructive heuristic and Tabu Search, is set to 10 minutes.

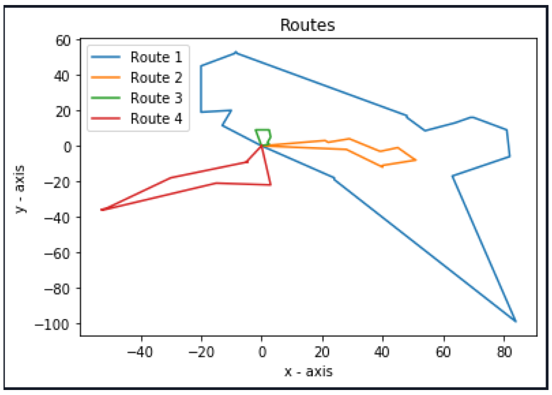
6.1.1 First stage performance

Now we discuss the performance of the first stage methods, i.e. the constructive and improvement heuristics. Below the costs obtained, compared to the optimal costs, as well as the identified and optimal routes are displayed. We do so by setting all buffer parameters to zero, as well as not incorporating the duration constraint.

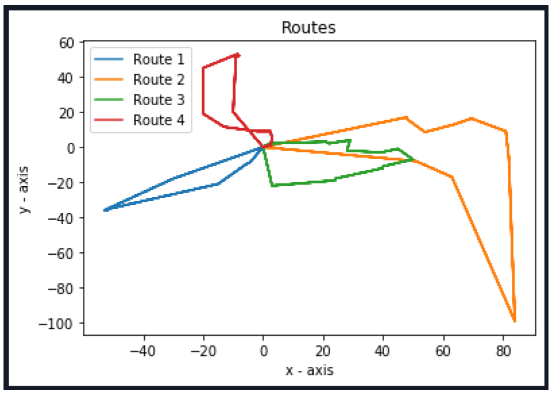
Optimal cost: 724

First stage cost: 751.51

Time: 565 seconds



(a) Optimal routes.



(b) First stage results.

Figure 8: Instance 1: First stage results.

Firstly, the cost obtained by the first stage heuristic returns a solution which is very close to the optimal result cost wise, achieving a value less than 4% higher. However, it is very interesting to see that the heuristic reaches such a close value while having noticeably different routes. As can be seen by comparing Figures 8a and 8b, in which the optimal routes, and the outputted routes of the first stage method are presented. There are some similarities in the grouping of customers, yet some big differences both in the ordering and in the clustering of locations. The different ordering of the routes can be seen in Appendix A.2 where the list of customers per route is presented. Route 4 in Figure 8a and route 1 in Figure 8b are quite similar, they only differ by a couple of customers which are present in one but not the other route. However this is the only route which is very close to its optimal counterpart, the other routes are quite different. The customers on route 1 in the optimal setting are split over routes 2 and 4 in the obtained solution, while the customers serviced by routes 2 and 3 in the optimal scenario, are combined into route 3 in the heuristic's output. Overall the heuristic performs extremely well in the small problem instance, reaching a near optimal solution in under ten minutes. The Tabu search converges to its obtained solution before reaching the time limit, thus ensuring that a better solution was not on the verge of being found.

6.1.2 The overall performance

Now we analyse the overall performance of the algorithm. To do so we set the buffer parameters as defined in Section 6. As the cumulative demand of customers can vary from the cumulative nominal demand, we present the expected demand as well as the real demand outcome of the worst iteration.

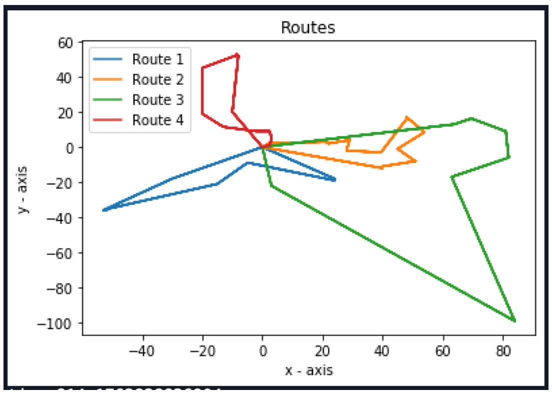
Expected demand: 7220

First Stage cost: 812.69

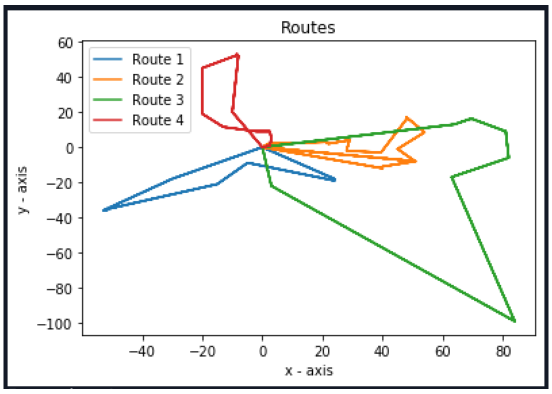
Demand outcome: 7328

Overall cost: 1015.95

In the following section we discuss the performance of both stages combined. As previously mentioned, we set all buffer parameters to their predetermined value and reinstate the constraint on a vehicle's shift duration. In Figure 9a the first stage routes with the updated parameters are shown. Comparing Figures 9a and 8b, the change induced by the buffer parameters on the routes is visible, specifically on routes 2 and 3. The cost incurred by new routes is 812.69, an increase of about 8% in comparison to the routes calculated without buffer parameters.



(a) Initial routes.



(b) Final routes.

Figure 9: Instance 1.

As mentioned above, we have run 100 iterations of the simulation, given the same initial routes provided by the first stage problem, and shown in Figure 9a. The Figure shown in 9b corresponds to the worst iteration, which results in a final cost of 915.95, plus a penalty of 100 due to one of the routes' duration exceeding the limit, namely route 2 where a failure occurred and a detour to depot recourse action was undertaken to amend the failure. The worst iteration presented a cumulative demand of 7328, 108 units more than the deterministic counterpart. However there were iterations with higher cumulative demand which led to lower objective values. This is an interesting, and partly unexpected result.

Preventive recourse actions worked very well in other iterations, hindering routes from exceeding the duration limit, therefore a more insightful analysis into when to apply the latter may result in a lower overall cost. Furthermore, not more than one failure occurred in each iteration, deeming adequate the choice of the buffer parameters.

Table 1: Failure analysis instance 1.

	Instance 1
Number of Failures	0.32
DTD	0
Partial Rescheduling	0.32
Partial rescheduling ended on DTD	0.17
Preventive DTD	0.58

In Table 1, we list the number of failures and what recourse actions we mostly adopted over the 100 iterations. The values reported are calculated as the number of times an event occurred, divided by total number of iterations tested, i.e. 100 iterations. In each iteration there is at most one failure, in total over the 100 iterations we incurred only 32 failures. In all these 32 failures, the chosen recourse action was partial rescheduling. However, in 17 iterations, the partial rescheduling was unsuccessful and ended on the detour to depot (DTD) recourse action. Furthermore, on 58 iterations, where a failure did not occur, a preventive DTD was adopted. And finally, in 10 iterations, no failure occurred and no preventive recourse action was adopted.

6.2 Instance 2

The second instance is comprised of 71 customers plus the depot, with a cumulative demand of 114840. There are four vehicles, each with capacity 30000. The customers are collectively closer to the depot than in the first problem instance as can bee seen in the figures below where the axis present a much smaller dimension range than the first instance. Given the increase in the number of customers, the time limit of the first stage is set to 13 minutes.

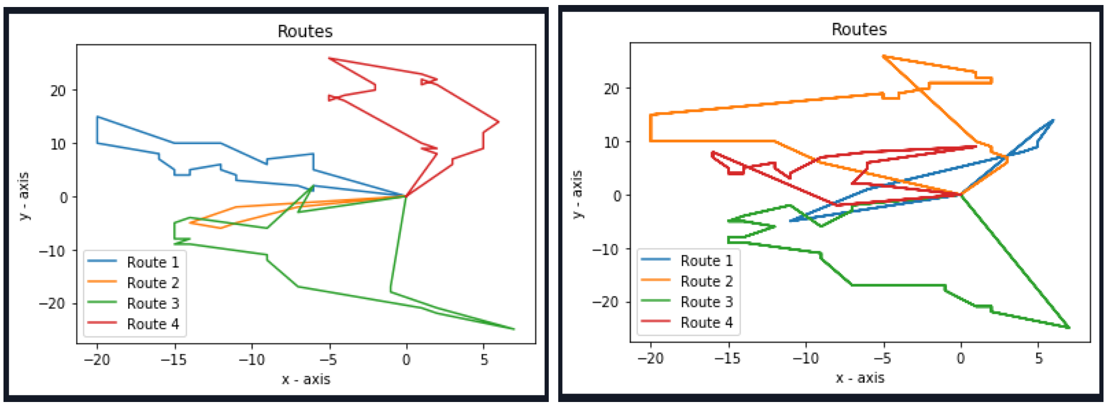
6.2.1 First stage performance

Now we discuss the performance of the first stage methods with all buffer parameters set to zero.

Optimal cost: 237

First stage cost: 300.09

Time: 776.25 seconds



(a) Optimal routes. **(b)** First stage results

Figure 10: Instance 2: First stage results.

In the second instance the results of the heuristic are not as close to optimal as in the first problem instance. We achieve a cost of 300, resulting in a 26% increase in objective value in comparison to the optimal value. The outputted routes are quite different from the optimal routes as can be seen in Figure 10. A 26% increase in objective value is not a bad result however, a better solution is believed possible with more complex neighbourhoods available in the Tabu Search. The main obstacle in reaching a better result in the first stage for this instance was in the initial route constructed via a Nearest Neighbourhood heuristic. This heuristic can work very well in many instances however in this case it created a very bad starting point for the Tabu Search. The initial solution of the Nearest Neighbourhood heuristic is presented below in Figure 11. The chaotic initial solution is due to the density in which the customers are located in the small area, in comparison to that of instance 1 which presented almost half as many customers in an area which is more than double that of the current problem.

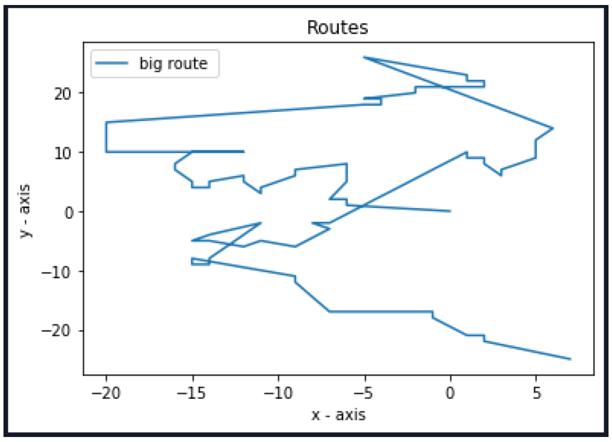


Figure 11: Nearest Neighbourhood result

6.2.2 The overall performance

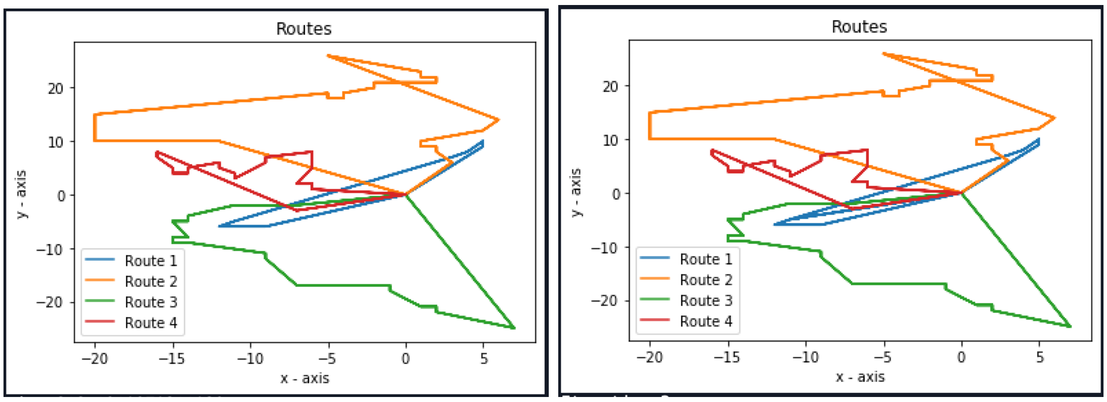
Now we analyse the overall performance of the algorithm. As before, we reset the buffer parameters defined in Section 6.

Expected demand: 114840

First Stage cost: 285.58

Demand outcome: 116426

Overall cost: 306.75



(a) Initial routes. **(b)** Final routes

Figure 12: Instance 2: First stage results.

Before we analyse the second stage results we must discuss the result of the first stage methods under the new parameters. Unexpectedly, with the capacity of the vehicle set to 90% of the real capacity, the cost obtained for the first stage problem is lower than that obtained when the real capacity is taken. This result was tested many times and consistently returned these costs. This difference can be seen by comparing Figures 10b and 12a. First, we note that this result is possible as routes which were feasible with capacity Q might be infeasible under capacity Q', leading the algorithm to making different moves from the start of the Tabu Search. This shows the need for more complex neighbourhoods in this problem instance, as the Tabu Search is not able to improve on the initial solution sufficiently.

As before, the final routes of Figure 12b corresponds to the worst iteration over the 100 tested. On almost all 100 iterations tested, a failure occurs or is prevented from occurring on route 1 at the same customer, namely customer 12. Depending on the demand outcomes the algorithms prevents the failure with the preventive return to depot or incurs a failure, and applies the detour to depot recourse action. The customer has a very high demand, hence the partial rescheduling is never considered. The total demand of the customers varies from 107990 to 120582 over the 100 iterations, yet the failure occurs always at customer 12 on route 1. The nominal demand of this customer is of 21611 units. It is the customer with the highest nominal demand, by more than twice that of any other customer's demand, thus it is understandable how it becomes a vulnerable point to failures.

In a real world setting this customer could be removed from the set of routes calculation and handled separately, for example assigning it to the vehicle that terminates its route first, as to avoid a certain failure.

Table 2: Failure analysis instance 2.

	Instance 2
Number of Failures	0.57
DTD	0
Partial Rescheduling	0.57
Partial rescheduling ended on DTD	0.57
Preventive DTD	0.43

In Table 2 we analyse the failures that occurred over all 100 iterations. Again, the values reported are the averages: the number of times an event occurred divided by the total number of iterations tested, i.e. 100 iterations. As in instance 1, in every iteration of instance 2 at most one failures occurs or is prevented from occurring. Important to note that in all iterations, if a failure occurred or a preventive recourse action was implemented to prevent a failure, the latter was on the same customer. Namely customer 12, mentioned before, which presents the highest demand. Depending on the demand of the other customers visited before customer 12 on route 1, the algorithm implements a preventive recourse action, which occurs in 43 iterations out of 100, or reaches the customer, incurs a failure, and attempts a partial rescheduling in the remaining 57 iterations. In all iterations in which a partial rescheduling action is attempted, the program fails to find an other vehicle which has sufficient capacity to service the customer in question, and thus implements a DTD recourse action. Overall, customer 12 is the only customer that leads to a failure or leads to a preventive recourse action being adopted over all 100 iterations.

6.3 Instance 3

The third and final instance tested in this thesis is the largest problem instance considered. A network of 134 customers is given, with a cumulative demand of 14620, as well as 7 vehicles, each with a capacity of 2210. Given the increase of the network, the time limit on the first stage methods is set to 16 minutes.

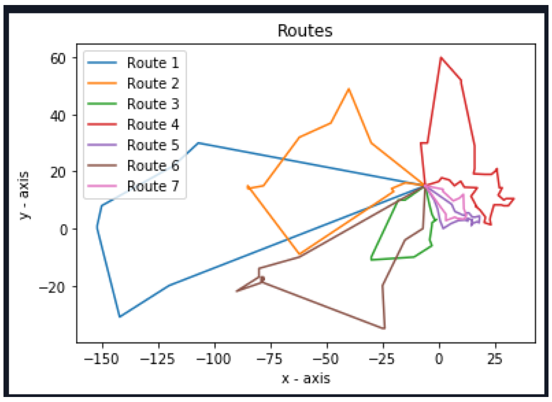
6.3.1 First stage performance

Again, we first discuss the performance of the first stage methods only by comparing the results obtained to the optimal ones, when setting all buffer parameters to zero and disregarding the duration constraint.

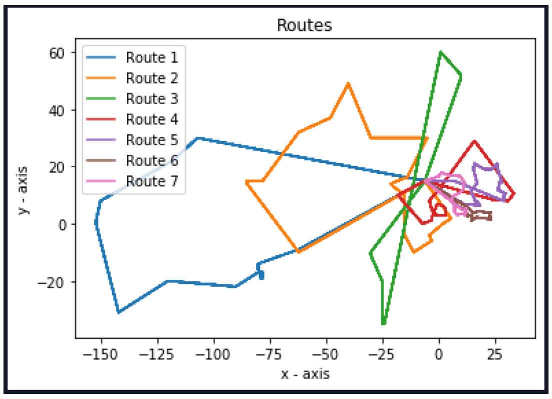
Optimal cost: 1162

First stage cost: 1248.59

Time: 944.42 seconds



(a) Optimal routes.



(b) First stage results

Figure 13: Instance 3: First stage results.

The first stage problem is solved in just over 15 minutes, achieving a result of 1248.59, which is about 7.5% above the optimal value. Given the size of the problem this is an excellent result. By comparing Figure 13a and Figure 13b, it is clear how few routes in the heuristic were constructed in a similar way to the optimal ones. Route 1 is the only extremely similar route to its optimal counterpart. Route 2 of our solution includes all customers visited by route 2 in the optimal solution, as well as customers of route 3 of the optimal solution. Route 3 through 7 of the heuristic cover the same customers as their optimal counterparts with some differences in grouping, especially on route 3 and 4. The costs difference between the optimal solution and the one found by the heuristics is so small as the differences between the routes are mainly among the routes to the right hand side of the depot, where all the customers are very closely located, thus leading to a small difference in route cost when assigned to a different route, as can be seen in routes 4 though 7 of both Figure 13a and Figure 13b.

It is unexpected in many ways that the performance of the third, and largest, instance is so close to the optimal value when the second instance achieved at best a result which was 26% higher. One reasoning behind this is that the outputted route of the Nearest Neighbourhood heuristic was a much better starting point for the Tabu search in this instance, as shown in Figure 14.

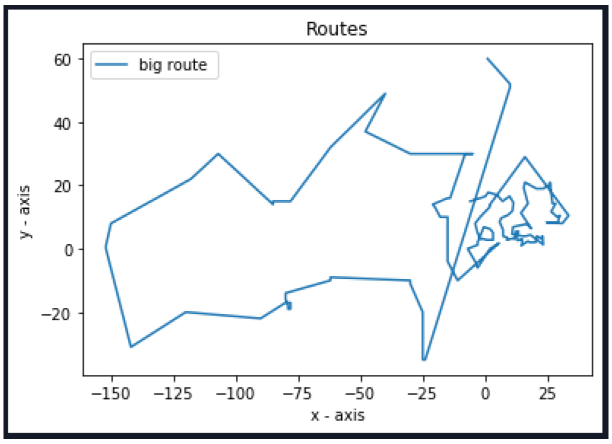


Figure 14: Nearest Neighbourhood result

6.3.2 The overall performance

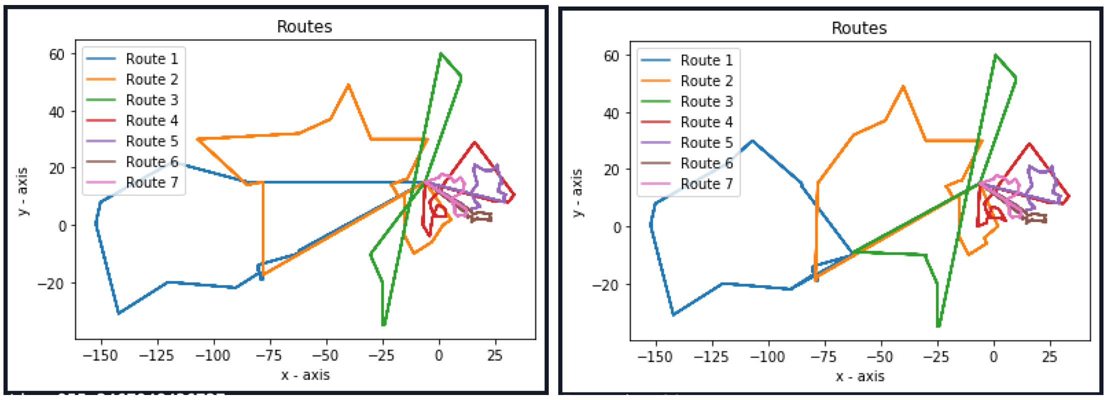
Now we analyse the overall performance of the algorithm by resetting the buffer parameters to their predetermined values and reinstating the duration constraint.

Expected demand: 14620

First Stage cost: 1256.26

Demand outcome: 15034

Overall cost: 1557.19



(a) First stage routes.

(b) Final routes.

Figure 15: Instance 3: First stage results.

Firstly, we notice that the lower considered capacity for each vehicle only leads to an increase in objective value of the first stage routes of only 0.6%, although the routes figured in 13b and 15a present clear differences, especially on route 2. Notice that route 1 returns to depot although the line is covered by routes 2 and 3 which traverse the same path. The change in path in Figure 15b in comparison to its initial route in Figure 15a, is due to a new customer which was assigned to route 1 in a partial rescheduling action.

Figure 15b shows the worst iteration over the 100 customer demand simulations tested. There are some noticeable differences between the routes shown in Figures 15a and 15b. These changes are due to failures and recourse actions adopted to amend and prevent the latter. In the iteration presented, one preventive detour to depot was undertaken, as well as one detour to depot and one partial rescheduling to amend two failures. The cost of the final routes is 1357.19, plus 200 related to penalties incurred for exceeded duration on two routes.

Table 3: Failure analysis instance 3.

	Instance 3
Number of Failures	6
DTD	0
Partial Rescheduling	6
Partial rescheduling ended on DTD	1
Preventive DTD	1

As opposed to instance 1 and 2, where at most one failure occurred per iteration, in instance 3 we incur multiple failures on every iteration. Given the size of this problem instance this is not a bad result. We depict a failure analysis in Table 3, showing average number of failures and which recourse actions are mostly adopted. In every iteration a preventive recourse action is adopted on route 7 before servicing customer 71. The customer's demand is not very large but appears to be a vulnerable point of the route as the customers before all have high expected demands. In all iterations six failures occur, all occur on route 6 but not always at the same customer. Regardless, in each iteration the first 5 failures are amended via partial rescheduling while for the last failure the algorithm attempts to amend via partial rescheduling, but ends on DTD. Route 6 is one of the longest routes, as can be seen in Appendix A.4 in the first stage routes, thus the other vehicles have most likely returned to depot or have insufficient capacity to service the customer which was last failed to service. Overall, given the problem size of this instance, a constant amount of 6 failures is not a bad result, especially considering that all except one are amended via partial rescheduling.

6.4 Overall methodology performance

The performance of the first stage methods is overall very satisfying. Two out of the three instances analysed perform extremely well while one performs satisfyingly well. The higher increment in objective value of the second problem instance is due to the result of the Nearest Neighbourhood heuristic. This algorithm does not always perform poorly as can be seen in the results of the first instance and especially in the results of the last instance, however it is very sensitive to the disposition of customers. By comparing Figures 11 and 14, this is clear. On one hand it is nice that the algorithm's performance is not dependent on the size of the network and thus in a realistic setting if, for a given network, the algorithm does not perform so well, a new heuristic can be chosen to initiate a route or additional, more complex, neighbourhoods can be added to the Tabu Search in order to find better solutions. In all instances the first stage solutions converge before reaching the time limit and thus providing a higher time limit would not suffice to determine a better solution.

In all problem instances the buffer parameters selected lead to few failures occurring, furthermore all failures are easily amended via the recourse actions. In some iterations a penalty is incurred due to excess duration of one or more routes, however it is important to note that often in two iterations the cumulative demand of customers was much higher than that expected, as we allow customers not only to deviate from their nominal values but also for the total demand to deviate from the total expected demand by up to 5%, which can lead to a noticeable increase. Given that the problem solved was constructed from a deterministic VRP, in which there is no limit on duration, it is possible that, under the additional constraints and uncertainty in demand, there is no solution which does not incur any penalties. This is the case for instance 3.

Overall both the first and second stage methods are believed to perform well. The first stage methods identify satisfying routes, especially under problem instances 1 and 3, in a limited time schedule in order to appeal to real life situations. The second stage methods are able to make instant decisions, analysing the current situation of all other vehicles, in order to amend and prevent failures. It is very pleasing to see that the partial rescheduling recourse action was always considered, as no failures occurred within the first visits on any route, and was often successful in finding an alternative vehicle to service the customer in need, especially in instance 3. The partial rescheduling action was not so successful in the first and second instances as it was in the third instance, but this can easily be connected to the number of vehicles, and to the difficult customer of instance 2.

The algorithm can be easily adapted to include more complex neighbourhoods for the first stage heuristics, or more complex decision criteria in the second stage method in order to better accommodate specific scenarios but is generally a well performing, fast algorithm for solving robust vehicle routing problems in a two-stage manner.

7 Sensitivity analysis

In this section we analyse the effects of different buffer parameters on the different instances. As mentioned in the previous section, we examined the performance of the algorithm on different instances, by taking the same buffer parameters. We selected the values that lead to the best performance overall for the result Section. Now we will identify the best parameters per instance.

7.1 Safety space

First we analyse the effect of different safety space values, i.e. which percentage of the vehicle's capacity to leave idle in the first stage. Each scenario is tested 100 times on each problem instance, and the best parameter is chosen by comparing the worst iterations of each case. We review the following four cases:

Case 1: SS=1, cd_i=0 $\forall i$

Case 2: SS=0.95, cd_i=0 $\forall i$

Case 3: SS=0.90, cd_i=0 $\forall i$

Case 4: SS=0.85, cd_i=0 $\forall i$

Each case is tested 100 times, and the best parameter is chosen by comparing the worst iterations of each case. By analysing the table below, we chose Case 3 as our base parameters for the result section.

Table 4: Safety space sensitivity analysis.

	Case 1	**Case 2**	**Case 3**	**Case 4**
Instance 1	1204.47	1211.40	1015.94	1015.39
Instance 2	330.79	458.88	306.74	310.02
Instance 3	2036.59	2325.74	1557.19	2383.55

Table 5: Safety space sensitivity analysis without penalties.

	Case 1	**Case 2**	**Case 3**	**Case 4**
Instance 1	1104.47	1011.40	915.94	915.39
Instance 2	330.79	358.88	306.74	310.02
Instance 3	1536.59	1625.74	1357.19	1683.55

Case 3 corresponds to the chosen parameters of which the results were discussed in Section 6. As shown in Table 4 instance 1 performs best under case 4 and case 3, achieving a better result under scenario 4 by only decimals. Instances 2 and 3 perform best under case 3. The best scenario remains the same for all instances if duration penalties are not considered, as can be seen in Table 5. As expected, the second best parameter choice overall is that of Case 4, however this is not the case for instance 3, where the second best choice of parameters is actually to leave no safety space on the vehicle in the first stage.

Furthermore, by comparing Tables 4 and 5, we can see how many routes incurred penalties, by dividing the difference in values by 100. In the first stage we set penalties to be proportional to the degree of violation, while in the second stage we just set a fixed penalty of 100, whenever a route violates the duration limit. In instance 1 there is always one route which incurs a penalty except for case 2 where two routes exceed the limit on duration. In instance 2, there is only one case in which a penalty is incurred, namely case 2. Finally in instance 3, there are higher penalties, incurred almost on every route on the worst iteration, which is the one presented. The penalties range from 200 in case 3 to 700 in case 2 and 4. It is interesting to see that instance 2, which achieved a first stage costs which was furthest from the optimal value in its deterministic counterpart, incurs the least number of penalties in its worst iteration. Overall the difference in performance in the different cases among instances is very interesting. It shows how each problem instance has its own best choice of parameters, depending on problem size and customer distribution, rather than demand outcomes.

7.2 Customer deviation

Now we analyse the effect of different customer deviation parameters. This sets the expected demand of a customer to its nominal value plus some percentage given by the customer deviation. The higher the percentage, the closer the demand of each customer is taken to its largest possible value which is set to 1.2 times the nominal value. Seeing as we have assumed that not all customers can achieve their worst realisation simultaneously, we analyse the four following cases:

Case 1: SS=0.9, cd_i=0 $\forall i$

Case 2: SS=0.9, cd_i=0.05 $\forall i$

Case 3: SS=0.9, cd_i=0.1 $\forall i$

Case 4: SS=0.9, cd_i=0.15 $\forall i$

Table 6: Customer deviation sensitivity analysis.

	Case 1	**Case 2**	**Case 3**	**Case 4**
Instance 1	1015.94	1016.68	999.92	927.91
Instance 2	306.74	310.02	304.12	314.83
Instance 3	1557.19	2365.27	2595.02	2224.07

Table 7: Customer deviation sensitivity analysis without penalties.

	Case 1	**Case 2**	**Case 3**	**Case 4**
Instance 1	915.94	916.68	899.92	927.91
Instance 2	306.74	310.02	304.12	314.83
Instance 3	1357.19	1665.27	1895.02	1524.07

From Table 6, we see that instances 1 and 2 perform best under scenario 3, while for instance 3 this results in the highest cost over all cases. The best performing case for instance 3 is, unexpectedly, case 1 where no customer deviation is considered. Case 1 was taken as base case for Section 6, as it is the best for instance 1 3 and second best for instance 2 while not leading to an excessive increase in objective value for instance 1. As in the safety space analysis, we outputted two tables, Table 6 and Table 7 where the values differ by the penalties incurred. In Table 6 the true costs incurred are shown while in Table 7, the values without penalties are presented. Instance 1 incurs one penalty in all cases except the last, instance 2 incurs no penalties in any scenario while instance 3 incurs one penalty in case 1 but 7 in all other cases, i.e. all routes exceed their duration limit.

While the Safety Space parameter involves each vehicle in the same manner, the effect of the customer deviation buffer parameter is highly dependent on the demand of each customer. The improvements achieved in case 4 for instance 1, in case 3 for instance 2 and case 1 for instance 3, show that the buffer parameter can have a positive effect on overall performance, but can have different effects on different problem instances.

As mentioned before in Section 2.2.6, we introduced buffer parameters to avoid an over-conservative solution, which would lead to high first stage costs, or a stringent solution, which could lead to an excessive amount of failures in the second stage, leading to high costs again. Thus the buffer parameters were meant to avoid the two extremes. We expected that all cases in between these two extremes would result in lower costs for all instances. However, as seen in the sensitivity analysis in Tables 4 and 6, this is not the case. In both tables and in all instances, the costs are higher in the intermediate case 2 than in either extreme cases 1 and 4. This is an unexpected result, but the benefit of including the buffer parameters to some degree is still clear, as can be seen in Table 4 in case 3 for all instances.

The three instances analysed vary in number of customers, number of vehicles and size of area where the customers are distributed, however they all have in common the fact that among the customers there is a great difference in demand. Therefore it may be more efficient for the customer deviation parameter to differ based on the customer's expected demand. Of course, we cannot expect a specific customer deviation parameter for each customer in a real world setting however a partition-like system can be put in place as to have a higher percentage for the customers with a low demand, and a smaller percentage for customers in higher brackets of demand, in order to avoid the disparity currently present. We believe this would improve the performance of the algorithm over all instances, and gives insight for future research on identifying favorable partitions.

8 Conclusion

In this thesis we considered a robust vehicle routing problem with capacity constraints on the vehicles and uncertainty in customer demand. This problem is of great relevance in the logistics and transportation world as the scale at which goods are transported has only increased in the past decades, and with it the importance of efficient routing. In this thesis we tackle the vehicle routing problem with uncertainty in customer demand, assuming no information regarding the probability of a customer's demand is known. The robust vehicle routing problem remains one of the least studied variants and thus offers the opportunity for new solutions to be studied.

The goal of this thesis was to formulate a solution which is applicable and tractable in real life situations. For this reason, real life data is used to test the constructed algorithm, as well as adding service times and duration constraints to mimic a real world shift length limitation. We formulate the problem as a two stage robust optimisation problem. In the first stage we solve a deterministic vehicle routing problem, taking customer demand as given. Then, in the second stage, via a Monte Carlo simulation, we identify possible failure points and amend or prevent them with recourse actions. The choice of solving the problem in this manner was driven by the consistent objective of creating a solution which would be appealing in the real world, in which minor changes can be undertaken while routes are in progress thanks to real-time communication and availability of computationally tractable heuristics.

We included buffer parameters to ensure all vehicles have some additional space accounted for when they leave the depot, in order to service some extra demand across their route. This revealed to be a positive choice, as shown in Section 7, leading to lower overall costs on all instances. The results reported in Section 6 are very satisfying. In all instances few failures occur in the second stage while defining efficient routes in the first stage. Given a specific instance, and the costs shown in Section 7, the buffer parameters can be adopted to improve the costs even further. Furthermore, we already identified how the overall algorithm could be improved, for example by partitioning on the assignment of customer deviation and by including more complex neighbourhoods in the Tabu Search.

We believe we achieved our goal of presenting a well performing algorithm which tackles a RVRP as it can be done in many real world settings. Hopefully some new insight on solving RVRP was brought, as well as initiatives for further research.

A Appendix

A.1 Parameters

- n customers,
- m vehicles,
- $SS \in [0,1]$: Safety Space, percentage of vehicle capacity to keep idle as buffer parameter,
- Q: capacity of each vehicle
- $Q' = Q \cdot (1 - SS)$, capacity considered when solving the first stage problem,
- q_i: the nominal demand value for each customer $i \in I$ given by its demand range $[a_i, b_i]$,
- $cd_i \in [0,1]$: customer deviation parameter for customer $i \in I$ given as a percentage,
- $q'_i = q_i \cdot (1 + cd_i)$, the expected demand of customer $i \in I$ for the first stage problem,
- L : limit on length of route for all vehicles set to an 8-hour work shift,
- s_i: service time of customer $i \in I$,
- d: neighbourhood distance.

A.2 Routes Instance 1

Optimal routes:

Route 1 : 55, 56, 42, 58, 57, 40, 69, 41, 39, 38, 70, 68, 67, 66, 65, 63, 64, 60, 59, 62, 61, 35, 32, 33,
Route 2: 37, 12, 2, 16, 15
Route 3: 36, 33, 19, 20, 3, 18, 14, 17, 13, 72, 7, 11, 9, 8, 10, 5, 4, 6
Route 4: 24, 25, 27, 26, 50, 71, 52, 51, 48, 49, 53, 46, 54, 44, 45, 47, 43, 28, 29, 23, 22, 31, 30, 21

First stage results:

Route 1: 33, 35, 55, 56, 25, 42, 58, 62, 61, 59, 60, 64, 63, 65, 66, 67, 68, 34
Route 2: 57, 40, 69, 70, 38, 39, 41, 71, 52, 50, 51, 48, 49, 45, 43, 44, 47, 54, 46, 53, 26, 27, 24, 30, 21
Route 3: 29, 28, 23, 22, 31, 32, 12
Route 4: 37, 36, 19, 15, 20, 3, 16, 2, 14, 18, 17, 13, 72, 7, 11, 6, 4, 9, 5, 8, 10

A.3 Routes Instance 2

Optimal routes:

Route 1 : 55, 56, 42, 58, 57, 40, 69, 41, 39, 38, 70, 68, 67, 66, 65, 63, 64, 60, 59, 62, 61, 35, 32, 33,

Route 2: 37, 12, 2, 16, 15

Route 3: 36, 33, 19, 20, 3, 18, 14, 17, 13, 72, 7, 11, 9, 8, 10, 5, 4, 6

Route 4: 24, 25, 27, 26, 50, 71, 52, 51, 48, 49, 53, 46, 54, 44, 45, 47, 43, 28, 29, 23, 22, 31, 30, 21

First stage results:

Route 1: 33, 35, 55, 56, 25, 42, 58, 62, 61, 59, 60, 64, 63, 65, 66, 67, 68, 34

Route 2: 57, 40, 69, 70, 38, 39, 41, 71, 52, 50, 51, 48, 49, 45, 43, 44, 47, 54, 46, 53, 26, 27, 24, 30, 21

Route 3: 29, 28, 23, 22, 31, 32, 12

Route 4: 37, 36, 19, 15, 20, 3, 16, 2, 14, 18, 17, 13, 72, 7, 11, 6, 4, 9, 5, 8, 10

A.4 Routes Instance 3

Optimal routes:

Route 1: 116, 115, 107, 108, 109, 110, 121

Route 2: 47, 119, 19, 18, 133, 132, 117, 118, 120, 131, 66, 20

Route 3: 74, 75, 77, 135, 78, 65, 64, 80, 68, 81, 34, 72, 67

Route 4: 92, 22, 26, 27, 28, 29, 93, 30, 95, 94, 46, 44, 45, 41, 4, 42, 43, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 89, 16, 14, 17, 91, 90, 88, 87, 86, 85, 84, 21, 83

Route 5: 61, 62, 55, 56, 58, 106, 98, 97, 39, 40, 96, 38, 37, 36, 100, 101, 99, 105, 102, 103, 51, 50, 35, 33, 48, 73

Route 6: 82, 114, 130, 129, 128, 122, 123, 124, 126 112, 113, 127, 125, 111, 70, 71 69, 134, 79

Route 7: 23, 25, 24, 60, 32, 31, 59, 57, 104, 54, 53, 52, 63, 49, 2, 76

First stage routes:

Route 1: 92, 22, 26, 28, 29, 31, 32, 60, 61, 62, 63, 51, 52, 53, 54, 103, 104, 57, 58, 55,

Route 2: 56, 105, 102, 101, 99, 100, 37, 36, 38, 96, 41, 4, 42, 43, 44, 45, 40, 39, 97, 98, 106

Route 3: 59, 46, 95, 94, 30, 93, 91, 90, 17, 14, 16, 89, 15, 12, 13, 11, 9, 10, 6, 5, 27,

Route 4: 3, 7, 8, 88, 23, 25, 24, 73, 74, 75, 77, 135, 33, 49, 2, 76, 48, 78, 65, 79, 72

Route 5: 50, 35, 64, 80, 68, 134, 67, 18, 19, 119, 47, 21, 83, 20, 66, 131, 120, 118, 132, 117, 82

Route 6 116, 115, 107, 108, 109, 110, 121, 122, 128, 127, 113, 126, 112, 111, 123, 124, 125, 129, 130, 114, 133

Route 7: 34, 81, 69, 70, 71, 84, 85, 86, 87,

A.5 Code

This thesis was solved by programming in Python 3.7 on a computer with 2.3 GHz Intel Core i5 processor. The code is too extensive to include so we just discuss the code briefly.

The problem was solved via object oriented programming, which is a flexible and effective way for problem solving. Objects such as customer, route and vehicle were created each with their own properties, such as capacity, demand, location, etcetera. Additionally, for each object we define a set of functions in order to facilitate the calling of these objects and their attributes in other parts of the code.

Once the structure is created, i.e. all objects are defined, and the reader code is set to read in the problem instance files, the methods for solving the problem are defined. First we defined all first stage methods, defining initial routes via the construction heuristic, and then improving them via Tabu Search. Subsequently, we created the Monte Carlo simulation framework as well as the decision criteria for the recourse actions and the partial rescheduling action. The detour to depot action is constructed under the vehicle object as this action only entails one route. Finally, everything is combined in the Main framework, where the parameters, such as the buffer parameters, are set and the various functions are called in the required sequence. We call the first stage methods once, and then for the outputted solution, we test the routes by calling the simulation function 100 times.

References

Ak, A. and Erera, A. L. (2007). A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation Science*, 41(2):222–237.

Baldacci, R., Toth, P., and Vigo, D. (2010). Exact algorithms for routing problems under vehicle capacity constraints. *Annals of Operations Research*, 175(1):213–245.

Ben-Tal, A., El Ghaoui, L., and Nemirovski, A. (2009). *Robust optimization*, volume 28. Princeton University Press.

Bullnheimer, B., Hartl, R. F., and Strauss, C. (1999). An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*, 89:319–328.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1):80–91.

Ekinci, Y., Serban, N., and Duman, E. (2019). Optimal atm replenishment policies under demand uncertainty. *Operational Research*, pages 1–31.

Fisher, M. L. (1994). Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42(4):626–642.

Gendreau, M., Hertz, A., and Laporte, G. (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research*, 40(6):1086–1094.

Gendreau, M., Hertz, A., and Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290.

Gendreau, M., Laporte, G., and Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research*, 88(1):3–12.

Golden, B. L., Wasil, E. A., Kelly, J. P., and Chao, I.-M. (1998). The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In *Fleet management and logistics*, pages 33–56. Springer.

Gounaris, C. E., Wiesemann, W., and Floudas, C. A. (2013). The robust capacitated vehicle routing problem under demand uncertainty. *Operations Research*, 61(3):677–693.

Joerss, M., Schröder, J., Neuhaus, F., Klink, C., and Mann, F. (2016). Parcel delivery. the future of last mile, travel. *Transport and Logistics*.

Laporte, G. and Louveaux, F. V. (1993). The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142.

Laporte, G., Louveaux, F. V., and Van Hamme, L. (2002). An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423.

Lee, C., Lee, K., and Park, S. (2012). Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *Journal of the Operational Research Society*, 63(9):1294–1306.

Maggioni, F., Bertocchi, M., and Potra, F. A. (2015). Stochastic versus robust optimization for a transportation problem. In *ODYSSEUS 2015-Sixth International Workshop on Freight Transportation and Logistics*.

Oyola, J., Arntzen, H., and Woodruff, D. L. (2018). The stochastic vehicle routing problem, a literature review, part i: models. *EURO Journal on Transportation and Logistics*, 7(3):193–221.

Salhi, S. (2002). Defining tabu list size and aspiration criterion within tabu search methods. *Computers & Operations Research*, 29(1):67–86.

Secomandi, N. and Margot, F. (2009). Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230.

Singh, A. (2019). Managing the uncertainty problems of municipal solid waste disposal. *Journal of environmental management*, 240:259–265.

Solomon, M. M. and Desrosiers, J. (1988). Survey paper—time window constrained routing and scheduling problems. *Transportation Science*, 22(1):1–13.

Spliet, R. (2013). *Vehicle routing with uncertain demand*. Number EPS-2013-293-LIS.

Sungur, I., Ordóñez, F., and Dessouky, M. (2008). A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions*, 40(5):509–523.

Toth, P. and Vigo, D. (2002). *The vehicle routing problem*. SIAM.

Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., and Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858.

Yang, W.-H., Mathur, K., and Ballou, R. H. (2000). Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(1):99–112.