Copenhagen Business School

Master's Thesis

cand.merc.(mat.)

Modeling of Stochastic Routing for On-Demand Waste Management Applications

Author:
Jakob Berg Lind, s110863

Supervisor:
Hans Keiding

May 17, 2021

Character count: 146.000
Page count: 67

Abstract

This thesis investigates the waste collection problem as to how to route vehicles for collection when the concept of emptying-as-needed is introduced. We propose a heuristic with quadratic time complexity that plans routes for the collection of waste from containers based on a reverse base stock model and parametric filling rates. This heuristic differs from a dynamic waste collection plan, as we additionally allow for the emptying of nearby non-needed containers. Our approach is illustrated using a theoretical upscaling of a real-life attempt at waste collection with emptying-as-needed in Glostrup municipality, which is located in Denmark. We clarify that the practice of On-Demand waste collection is more costly in terms of distance traveled compared to a static waste collection plan and a dynamic waste collection plan, respectively. The proposed heuristic is shown to be very competitive given the large flexibility of the parameters that consider the trade-off between cost and a balanced workload. Additionally, we provide a brief history of the literature along with a detailed description of vehicle routing and inventory management for waste collection purposes, which have been used to develop the proposed heuristic.

Resumé

Denne afhandling undersøger affaldsindsamlingsproblemet, herunder hvordan man ruteplanlægger køretøjer til indsamling når tømning-efter-behov introduceres. Vi foreslår en heuristisk med kvadratisk tidskompleksitet, der ruteplanlægger til indsamlingen af affald fra containere baseret på en omvendt lagermodel med sikkerhedslager og parametriske fyldningshastigheder. Denne heuristik adskiller sig fra en dynamisk affaldsindsamlingsplan, idet der tillades tømning af nærliggende ikke-nødvendige affaldscontainere. Vores tilgang illustreres ved hjælp af en teoretisk opskalering af et virkeligt forsøg på affaldsindsamling med tømning-efter-behov i Glostrup kommune, der ligger i Danmark. Vi afklarer, at brugen af On-Demand affaldsindsamling er dyrere med hensyn til tilbagelagte kørte afstande sammenlignet med henholdsvis en statisk og en dynamisk affaldsindsamlingsplan. Alligevel har den foreslåede heuristik vist sig at være meget konkurrencedygtig på baggrund af den store fleksibilitet af de parametre, der betragter afvejningen mellem omkostninger og en afbalanceret arbejdsbyrde. Derudover giver vi et kort resumé af litteraturen sammen med en detaljeret beskrivelse af ruteplanlægning af køretøjer og lagerstyring til affaldsindsamlingsformål, som er blevet brugt til at udvikle den foreslåede heuristik.

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

The collection of waste is a highly visible and important municipal service that involves large expenditures and contributes to environmental pollution. This type of reverse logistics received considerable attention in recent years due to increasing environmental and ecological concerns as well as economical benefits. In addition to saving from direct material costs such as fuel and man-hours, companies can also save from disposal and energy costs through dynamic reverse logistics. Being able to sort the waste at home in the simplest possible way is a prerequisite for one good environmental involvement. This is to e.g. meet the municipality's demands for increased participation among residents in waste minimization, recycling, and reuse. At the same time, the dense city leads to major challenges for efficient waste sorting. Not least in urban environments that are often planned according to other conditions than just waste collection vehicles in mind.

Waste collection is thus problematic, and often a source of local environmental problems if it is not carried out in such a way that the municipal residents become involved. If the local incentives are utilized in the right way, new technology can thus create new sustainable ones that instead are based on the needs of citizens.

The most effective method of collecting waste has until now been a set emptying day. However, hence the growing interest in vendor managed-inventory during the last decades, a combination with new technologies now make it possible to offer more personal collection schemes, so that the individual citizen decides when the container should be emptied. In such cases, replenishment frequencies do play an important role in integrated inventory models to reduce the total cost of supply chains.

In waste management, the ideal situation is to have the waste collectors avoid emptying half-filled containers and to have transport kept to a minimum for the sake of both the environment and financial. The concept of decide-yourself-emptying with digitized waste containers allows the customer to decide when they want the garbage truck to come by and empty their garbage and waste containers. The waste collector can thus provide customers with service when customers feel that they actually need emptying, i.e. On-Demand waste collection. This incentive against emptying in many cases half-full containers in connection with conventional emptying on predetermined schedules.

In most municipalities, waste collection is based on households being linked to standard subscriptions and the emptying of containers is taking place on predetermined scheduled routes. Such a traditional model does not open up for customized solutions. Further, the waste sector has previously been analogous and is often perceived as conventional in its own right development work. Thus, an On-Demand solution for the digitized waste sector of the future is lucrative, but not well examined or explored.

However, the municipal waste company NSR (Nordvästra Skånes Renhållnings AB) is looking for such On-Demand solution for the municipal of Helsinborg in Sweden. In 2018, NSR conducted a pilot test intending to assess customers' understanding and experience of demand-adapted container emptying, which in practice means emptying four-compartment containers close to the household after the customer's actual request. The experiment tests technical solutions that are intended to gain knowledge about how needs-adapted emptying actually works, and what results in the service offers in form of perceived customer satisfaction. A total of 250 households in Mariastanden in Helsingborg participated in the experiment. During the test, it was concluded that the number of empties decreased significantly. One of the other economic impacts that were pointed during the experiment was that some waste fractions experienced up to 40% fewer empties. This was due to 54% of all emptyings had an emptying interval longer than 28 days. Yet, 41% of all emptyings were performed on a Monday, and respectively 7%, 21%, 6%, and 25% for Tuesday through Friday. NSR concluded on the experiment that an On-Demand solution reduced wear on vehicles due to fewer emptyings, but they recognized the very skewed distribution of emptying days (NSR, 2019).

Further, 12 in-depth interviews were conducted. A study of these interviews by Michael Johansson at Lund University in Sweden showed that the households experience overall great customer satisfaction, hence the On-Demand service being simple and flexible while individuality is perceived to be important as the awareness that households have different needs and conditions for emptying containers (Johansson, 2019).

Also, interviews of NSR's own employees show indications of more efficient waste collection tours, increased perceived customer satisfaction of fast scheduling, and no need of driving any faster, despite longer distances between the ordering of container emptying. Lastly, these interviews claim that at least as many containers can be emptied per tour as before. However, with an increased geographical area, one could advantageously result in a larger emptying area during the same time period.

The study concluded by discussing the continuing need to inspect the effects of the experiment on transport, such as an examination of the vehicle routing planning, and whether such technology solution reduces or increases transportation costs. Thus, with a larger geographical case study area, the environmental effects associated with transport could be further deepened. The study clarifies that it would be relevant to highlight and to carry out the test linked to the waste rate per emptying, i.e. having an economic perspective clarify the degree of potential in the introduction of needs-adapted emptying.

1.1 Thesis Statement

The objective of this thesis is to examine the introduction of an On-Demand solution into waste collecting. This thesis will thereupon study how a new technology-driven solution in combination with vehicle routing can be used to model the otherwise cyclic predetermined routes for municipal's waste management. Thus, the main research question for this thesis is in which way and to what degree can the On-Demand waste collection planning be implemented using dynamic developed vehicle routing algorithms. For this, we study how parameters affect a designed and implemented heuristic for waste management through simulations as well as the performance of such dynamic vehicle routing model.

Specifically, we study how reverse inventory management can model waste containers for waste management, and how a proposed On-Demand technology-driven solution that utilizes this feature performs in a theoretical up-scaled environment for waste collection.

Chapter 2

Problem Background

Motivated by NSR's experiment in Helsinborg, Sweden, a similar experiment on On-Demand waste collecting takes place from September 2020 until March 2021 in Glostrup, Denmark. The experiment here is a collaboration between the current Norwegian waste collection company Urbaser A/S, the municipally owned waste company Vestforbrænding whose executing incineration at its plant, and Glostrup Forsyning which is the part of Glostrup municipal concerning drinking water, wastewater drainage, district heating, and the collection and disposal of waste in Glostrup.

This similar experiment forms the basis of the presented implementation for On-Demand waste collection in this thesis. That is, this experiment's presumption and premise provide the framework of On-Demand waste collection.

2.1 On-Demand Waste Collection in Glostrup Municipal

Glostrup, a suburb of Copenhagen, accommodates more than 4300 family homes including detached houses and terraced houses in 2020 (Danmarks Statistik, 2021). As of today, each household in Glostrup has a single waste container for ordinary household waste, which they only can fill with ordinary household waste. Each of these households' containers is to be included in a fixed determined waste collection schedule each week. Daily, the current renovator Urbaser is emptying around 730 containers on average every weekday, Monday to Friday, on a predetermined schedule. Without including allotment gardens and public accessible mini-containers, nearly 3650 containers containing household waste are collected each week.

This similar experiment taking place is however a small-scaled version, in which only the residents of a total of nearly 60 households in the two streets Ydergrænsen and Ejbyholm in Glostrup participates. The main goal of this experiment is to study a theoretical upscaled version of this small sample space, to contribute to the relatively unexplored area of On-Demand waste collection to whether such emptying-as-needed, gives greater flexibility, fewer empties, a more green waste collection, more waste for recycling, and overall greater satisfaction for the citizen and resident with an economical approach (Erlitz, 2021).

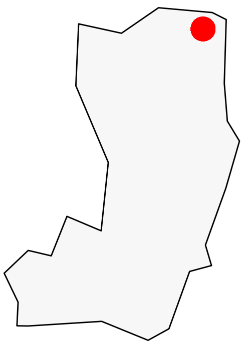

Figure 2.1: *Map of Glostrup municipality. The red dot at the top is Vestforbrænding waste deposit center.*

During this experiment period, the residents test two different technologies for On-Demand emptying of waste containers as alternatives to their fixed predetermined emptying schedule. The households at Ejbyholm will test a mobile-adapted self-service portal on the internet, while the households at Ydergrænsen will have a so-called slider mounted on their waste containers. Both technologies request works the same way, as in the residents will need to request waste emptying manually. Thus, these households have to request a waste collection themselves, when their waste containers are about to become full. Requesting an emptying is performed by the customer by navigating inside the self-service portal, or simply by sliding the mounted slider on their waste container. When using this slider, you must move the slider, whereby a sensor registers this slide and thereby sends the request for emptying to the renovator, namely Urbaser, who then will collect the waste. Figure 2.2 shows bintel AB's slider-solution, which was attached to the waste containers during NSR's experiment in Helsinborg.

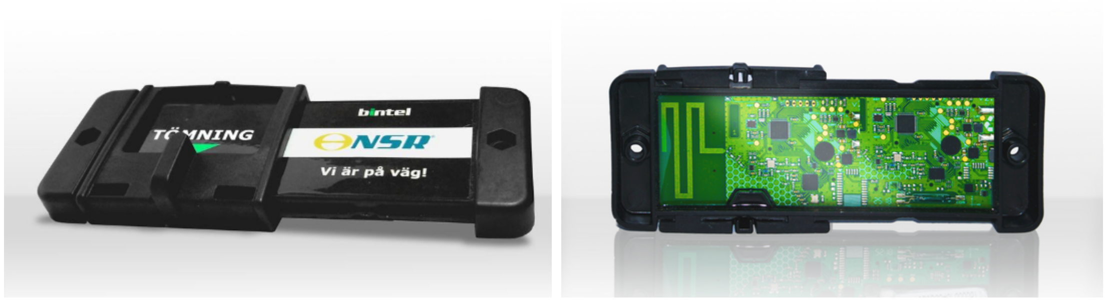

Figure 2.2: *The bintel SLIDER for a waste container. Left: front of the slider. Right: backside of the slider.*

A set of rules have been given for this experiment, namely within 2 days of the request of emptying, the container must be emptied. Given this, the ordinary rules of the waste collection still follow, such as the waste collector cannot take more waste, than there is inside the container, e.g. the container must not be filled more than the lid can be closed tightly and the waste must not be squeezed into the container. The residents must therefore request an emptying when their container is full. Moreover, if there may be an excess waste for the individual household of the ordinary amount, extra bags can be filled but they must be put next to the collection point on the day of collection (Glostrup Kommune, 2014).

Lastly, this experiment differs from the experiment which took place in Helsinborg, in that this pilot test investigates the use of a self-service portal versus the slider. Yet, the pilot test's main focus is directly linked to study the economical and environmental benefits achievable in an upscaled version of an On-Demand waste collection solution.

2.1.1 Problem Description

The theory of transportation management considers how to route the vehicles. That is, how to satisfy customer demand and minimize costs linked to a vehicle's cost. Moreover, the theory of inventory management considers how much to replenish the inventory. This is likewise how to satisfy customer demand and minimize costs associated with inventory. One way to achieve this is to use the principle of Just-In-Time since it accomplishes minimal costs associated with the inventory, by keeping a low, if any, inventory. This principle is in fact a replenishing policy where demand is only fulfilled the exact moment the customer demands. Therefore, the combined decision on when to refill the customers' inventories, both how much, when, and which customer to replenish, and how to route vehicles that execute the delivery, is also known as the Inventory Routing Problem (IRP) (Snyder and Shen, 2019).

One specific application of the IRP is the waste collection problem which includes a large number of customers, fluctuation in the demand, and a long planning horizon. However, the special case in which the renovator plans the emptying of containers with periodically routing, and thus bases planning on the amount of waste inside the containers, the waste collection problem becomes a special case of IRP. The waste collection problem is a reverse IRP since the purpose of visiting a "customer" is collecting rather than delivering something, and customers' inventories are emptied instead of being replenished. However, the decisions dealt with are similar in both of IRP's subproblems, regarding which customer to visit and how to route the vehicles. Solution methodologies for IRPs also work for waste collection problems as long as they support decisions for uncertain waste deposits and a large number of waste containers, which are usual settings for a waste collection problem (Mes et al., 2013).

Although the insights apply to the IRP with many customers, we consider a theoretical upscaling of Glostrup's pilot project's small sample space. In our On-Demand waste collection problem, the municipality's renovator has to plan periodically when and which waste containers to empty and how to route the vehicles that actually collect waste. This must be done subject to the principle of Just-In-Time such the renovator's collection costs and waste container overflows are minimized while residents' satisfaction is maximized simply due to requests being met.

In terms of On-Demand waste collection, the problem is simply a routing problem with both stochastic customers and demands. For our implementation, we additionally extend the waste collection problem by allowing containers that not necessarily are requested for emptying to be emptied. This asserts itself in the waste collection environment since time spent emptying a single container usually is minimal. Therefore, it comes naturally to empty related and nearby waste containers that may otherwise be requesting an empty within the subsequent period.

Now, Glostrup's framework is used to define some specific characteristics of this On-Demand waste collection problem:

- There are no time windows for emptying the containers, only a maximum number of locations to visit and a transportation cost (such as fuel and time) to satisfy.
- The capacity of containers can be exceeded with overflow waste. When emptying a container, it should be entirely emptied including the overflow waste.
- A large number of containers has to be emptied each working day.
- Household waste is the only waste fraction considered.
- The requests for emptying must be met within 2 working days.
- The vehicles' parking location and waste disposal site is the same since the disposal site must be visited as the last stop regardless.

The latter characteristic is reasonable, hence the parking space of Urbaser's vehicles is very closely located to the waste disposal site, meaning that only a minimal constant cost is to be added to each vehicle. Furthermore, the waste collectors will receive a route plan before working, meaning that route planning must be ready quickly. Waste deposit rates, as well as deposit volumes, are stochastic, and they might even fluctuate due to e.g. holidays and seasonal effects. These may yet only be estimated. Thus, we take this stochasticity into account as well.

We now draw the parallels with IRP formulations by firstly introducing the methodology to the stochastic nature of our problem, regarding which customers to visit, by presenting routing algorithms that handle large-scale instances and thus are ideal for waste collection problems. Thereafter, we provide a model in a time-discretized version of base stock modeling to cope with the stochastic filling in a waste collection problem together with a distinguishing of waste containers in a routing setting. Lastly, we present our proposed and implemented model in a case study of Glostrup municipality where On-Demand waste collection has been an initiated project.

Chapter 3

Vehicle Routing Modeling

The first objective within On-Demand waste management is to consider algorithms capable of efficiently solving large-scale networks. In this chapter, we present algorithms for solving waste management's routing problem.

3.1 Background

The Vehicle Routing Problem (VRP) is a combinatorial optimization and integer programming problem which seeks to find the optimal set of routes for a fleet of vehicles to deliver to a given set of customers. VRP was first formally introduced by Dantzig and Ramser in (Dantzig and Ramser, 1959). As of today, the VRP has remained an important problem within logistics and is one of the most studied combinatorial optimization problems. VRP is a type of scheduling, which fits the On-Demand waste collection experiment in Glostrup, hence the need of studying the economic aspects of such waste collection, when route optimization is the major research interest. A fixed predetermined waste collection schedule is a variant of VRP, where reverse logistics is introduced since the vehicle fleet's purpose is to collect rather than deliver. In the waste collection regime, each vehicle is a garbage truck, which is constrained in the sense of having limited capacity, thus only a certain amount of waste containers can be collected considering both weight and size of the waste before having to deposit at a waste disposal center.

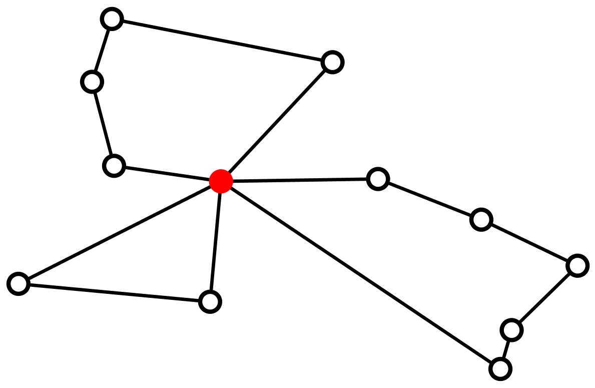

Figure 3.1: *example of customers being assigned to three different vehicle routes. The depot is the red dot in the center.*

The variant seen in waste collection is defined as the Capacitated VRP (CVRP) (Mes et al., 2013). We are here given a number of customers, each with its own demand, the distance between each pair of customers, and a number of capacitated vehicles. Each vehicle starts at the depot whereby it travels to serve the customers. The main objective is thus to minimize the total distance traveled while respecting the capacity constraints of the vehicles. This is achievable by splitting the deliveries between the vehicles and specifying an order in which each vehicle undertakes its task, such that the distance traveled by the vehicles is minimized and any pre-stated constraints are met (Toth and Vigo, 2014).

A solution to the CVRP thus consists of making each vehicle serve a subset of the customers, i.e. to traverse a route from the depot to a number of customers meeting each customer's demand and back to the depot. If the routes have been created so that all customers are served, while the accumulated demand of the customers has not exceeded the capacity of the vehicle, the problem has a feasible solution. It is considered irrelevant to traverse each route forward or backward since we make the fairly reasonable assumption of symmetrical distances in all directions.

VRP encircles many related problems. Alternative VRP formulations model other constraints. Some of the classic VRP formulations include: the VRP with Pickups and Deliveries (VRPPD) where each vehicle picks up items at various locations and drops them off at others; VRP with Time Windows (VRPTW), which is the introduction of time window constraints where each customer must be visited within a certain time frame; VRP with Multiple Depots (VRPMD) where multiple, yet different located, starting depots are present. However, these alternative formulations do not fit the specific problem that arises when waste collecting in the municipality of Glostrup, hence for instance no pickups or deliveries appear. Time Windows are not being the crucial incentive, more precisely because of the employment conditions in Glostrup. Lastly, only a single depot, namely the waste disposal site is existent. All routes implemented consider this assumption of both the parking location and waste disposal site is the depot.

However, one closely related VRP formulation in the context of waste management is the Periodic VRP (PVRP) (Toth and Vigo, 2014). In classic VRPs, the planning period is just a single day. In the case of PVRP, the VRP is generalized by extending the planning period to M days. Note, that if the planning period $M = 1$, then PVRP becomes an instance of the classical VRP.

The PVRP can also be seen as a multi-level combinatorial optimization problem, in which the objective is to generate a group of feasible combinations for each customer. Each customer in the PVRP must be visited k times, where $1 \leq k \leq M$. For example, if the planning period has $t = 2$ days $\{t_1, t_2\}$ then the $2^t - 1$ possible combinations for each customer are: $\{0, 1\}$, $\{1, 0\}$, $\{1, 1\}$. Further, a PVRP is solved in two stages. In the first stage, the selected combination for each customer must be subjected to the daily constraints, such that customers are to be visited each day. In the second stage, the PVRP must simply be solved for each day. This same procedure will be applied for our implementation of a heuristic for On-Demand waste collection, as our heuristic considers day-to-day planning.

The VRP in practical applications is not always deterministic, as uncertainty in the model's parameters exists. That includes both expected variations and unexpected events. The introduction of Stochastic VRP (SVRP) are models where one or several components of the problem are random. Such stochastic parameters are formulated using suitably defined random variables or through the use of scenarios. Often, the models that are considered to be stochastic are:

- stochastic customers: The demand d_i of each customer is a random variable.
- stochastic demands: Each customer v_i is present with probability p_i and absent with probability $1 - p_i$.
- stochastic times: Service times δ_i and travel times t_{ij} are random variables.

Likewise PVRP, the SVRP is also solved in a two-stage approach. In the first stage, an initial solution is determined before knowing the realizations of the random variables. That is, routes are firstly determined for all customers based on stochastic if demand is not known beforehand.

In the second stage, a recourse action may be performed on the solution when the true values of the random variables are known (Toth and Vigo, 2014). The customers having no demand on routes are then simply skipped. This skipping is either not servicing the customers without demand, or to skip visiting the customers. The difference here lies in when the demand is known, which is either after or before the planning has been made, respectively. The second stage typically happens if a vehicle has to return for more demand since the previous customers exceeded estimations.

The addition of stochastic is particularly appropriate for On-Demand waste collection as well, as both the customers and the demands may be assumed as random variables. The filling of a container in practice is non-constant, thus the filling is a random variable. Moreover, the customers are also present with a probability, which we assume is following their container's filling-level. Thus, the customer is present when their container is full since they are then requesting an emptying. The demand is therefore to some extent known, as we may assume demand of unit size. The applied SVRP strategy for On-Demand waste collection is thus to skip visiting customers that otherwise would be present, i.e. demand is assumed known before planning, since our heuristic also models the waste filling-levels.

3.2 Static Route Planning

Concerning waste management, the planning of this vehicle routing schedule is often considered to be static. Any static, i.e. fixed, routing schedule design routes that are repeated on a cycle. This cycle is for instance daily, weekly, or every second week which may be dependent on whenever there is sufficient demand (Mes, 2012). However, the static routes may be changed when the common customer base changes, such as the forecasted amount of waste.

One of the primary advantages to consider a static route schedule is the ease of execution due to driver familiarity. This is because the static route planning encourages a design for cycles related to routes with respect to truck capacity and driver hours. On the other hand, the disadvantages of a static route schedule are often seen in the inefficiency caused by the variability when forecasting the waste for the cycles, and the difficulty of efficient waste collection at individuals hence lack flexibility (Caplice, 2006). This is for instance the risk of overflowing the containers or the risk of emptying an e.g. only half full waste container. Exactly this is to be removed by converting to an On-Demand waste collection solution.

The current planning methodology in Glostrup is considered static. Specifically, the municipality's households are divided into five sectors. This sectorial division matches a weekly collection cycle such that each weekday, one sector is chosen for the collection. This means that no stochasticity is present as no uncertainty is involved when routes for such cycle in Glostrup were planned. The renovator thus takes advantage of static planning methodology, in which case the disadvantages also follow along.

3.3 Dynamic Route Planning

Any variable, i.e. dynamic, routing schedule design routes that change continually every day. The routes are therefore based on actual requirements, contrary to the deterministic nature of static schedules, which in waste management context is the filling-levels of the containers. The routes of dynamic methodology are typically designed for concrete vehicle and driver constraints (Caplice, 2006). This means that the dynamic route plannings often utilize all available capacity to provide the best possible solution.

The primary advantage of any dynamic planning methodology is thus the utilization of trucks and drivers. Additionally in dynamic planning, the flexibility of individual customer orders is present. This advantage is highly enhanced in an On-Demand waste collection, as the customers experience complete flexibility. This flexibility is the main driving force behind the On-Demand waste collection experiment in Glostrup as customers request emptying themselves. On the other hand, the primary disadvantages of any dynamic planning methodology are the difficulty of both determining optimal routes and maintaining a planning process, hence the $\mathcal{NP}$-hardness of a VRP[1]. However, the dynamic planning methodology's weakness in a real-world scenario is the strength of a static planning methodology, namely the execution may not match the plan, simply due to driver familiarity, seniority, and local knowledge.

Therefore, On-Demand waste collection includes the characteristics of dynamic planning methodology, thus the same strengths and weaknesses as any dynamic planning schedule, but with the enhanced advantage of complete flexibility, hence the residents must request an emptying themselves.

[1]See (Arora and Barak, 2009) for a complete discussion on non-deterministic polynomial-time hardness problems

3.4 The Origin of VRP

VRP generalizes the well-known traveling salesman problem (TSP) since we define a fleet of vehicles rather than a single unit. Informally, the TSP is defined as n points on a map, then a route is provided through each of the n points such that each point is only used once and the total traveled distance is minimized. The name originates from the classic real-world example, where a salesman is sent on a trip to n cities. Thereby, it is agreeably to select the trip such that the traveled distance is minimized (Lawler et al., 1985). Formally, the TSP is simply the determination of a minimum-cost simple circuit formulated using a complete graph $G = (V, E)$, with a node-set $V = \{1, 2, .., n\}$ corresponding to the cities and each edge $e_i \in E$ has an associated weight representing the distance between the nodes it connects. The goal is to create a Hamiltonian circuit, i.e., a circuit that visits all the nodes of a directed graph exactly once.

The problem is in fact $\mathcal{NP}$-hard, which is proved by showing that any instance of the TSP can be reduced to an instance of the Hamiltonian circuit. A Hamiltonian circuit is a vertex tour, i.e. cycle, that visits each vertex exactly once. Thus, if we let the graph G contain a Hamiltonian circuit, then traversing to all the vertices V of the graph will form a TSP. We now assume that a new, fully connected, graph G' contains a TSP, which also traverses all the vertices. Since none of the vertices are excluded from G' it necessarily uses all the edges of G to form a Hamiltonian circuit. Yet, G' is only extended by converting G to a complete graph, thus TSP can be reduced to an instance of the Hamiltonian circuit. The Hamiltonian circuit is thus simply an efficient solution to the TSP and VRP, since it visits each vertex exactly once while maintaining a short traverse on the edges, hence no avoidable edges are included in the solution. In figure 3.2, we illustrate this proof. Note that a Hamiltonian circuit does not necessarily have to be a peripheral route as in figure 3.2.

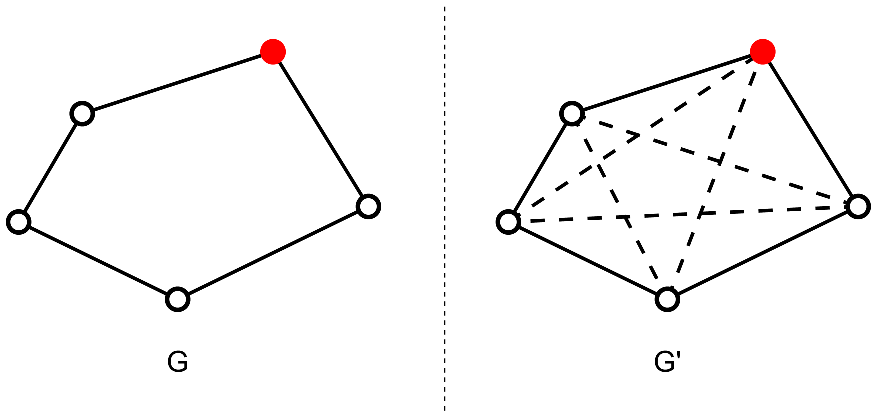

Figure 3.2: *Illustration of the proof of TSP being $\mathcal{NP}$-hard. Note that G is the Hamiltonian circuit and G' is a TSP. The top red dot is the depot.*

A generalization of the TSP is the Multiple Traveling Salesman Problem (mTSP), where multiple tours are constructed, thus multiple salesmen can be used to visit the cities. The purest mTSP can trivially be turned into a TSP by constructing a graph G with $n-1$ additional copies of the starting vertex and by forbidding traveling directly between the n starting vertices. However, this pure formulation of mTSP has no constraints on how the routes are constructed. Therefore, we may encounter tours of salesmen not even leaving the starting node, while others travel many cities (Lawler et al., 1985).

However, VRP and its family of related problems can be understood as being a generalization of mTSP that incorporates additional constraints. Some of these constraints, such as capacity limits, introduce additional dimensions to the problem that are in themselves hard combinatorial problems (Toth and Vigo, 2014).

Therefore, determining the optimal solution to the CVRP is $\mathcal{NP}$-hard in the strong sense since it contains the TSP as special cases when the fleet size is 1 (Toth and Vigo, 2014). VRPs being $\mathcal{NP}$-hard makes it very unlikely that an algorithm exists, with reasonable time performance, that can solve the problem exact. For instance, if all $n!$ combinations of a TSP should be determined, it would be requiring a running time of $\mathcal{O}(n!)$ which is rather unpractical (Lawler et al., 1985).

Lastly, time complexity thus becomes very important when comparing algorithms that handle $\mathcal{NP}$-hard problems. The time complexity of an algorithm is simply determined as the minimum of all possible scenarios for the algorithm, whereby an algorithm can be said to exhibit a certain growth rate following the order of growth functions. It is thus ideal to avoid algorithms with running times larger than polynomial since an algorithm with such running time of (n^2) can solve problems within a reasonable time (Arora and Barak, 2009).

3.5 Formal Definition

The underlying structure of the VRP is a complete, i.e. fully connected, directed graph $G(V, E)$. This structure and notations with respect to vehicles, routes, and the associated cost are outlined in the following definitions formulated in (Toth and Vigo, 2014).

A vertices set of $n + 1$ $(n \geq 1)$ vertices, $V = \{v_0, v_1, ..., v_n\}$, where node v_0 represents the depot and the remaining node set $V' = V \backslash \{0\}$ corresponds to the n customers. The set of edges, i.e. arcs, between the vertices is denoted $E = \{(v_i, v_j) \mid 0 \leq i, j \leq n, i \neq j\}$. A required supply of q_i units from the depot to each customer $i \in V'$ is given by $Q = q_i$. The demand is given by $d = (d_0, d_1, ..., d_n)$ where $\forall d_i > 0$ $(1 \leq i \leq n)$. Thus, d_i denotes the demand d_i of customer v_i as $d(v_i)$. Then $C = c_{ij}$ with $(c_{ij} > 0)$ is a cost matrix representing the cost of traversing edge (v_i, v_j), denoted $c_{ij} = c(v_i, v_j)$. We define $c_{ii} = 0$ $\forall i$, thus $c_{ij} \leq c_{ik} + c_{kj}$ $(0 \leq i, j, k \leq n)$. Therefore, if $c_{ij} = c_{ji}$ $\forall i, j$ then C is symmetric, otherwise if $c_{ij} \neq c_{ji}$ then C is asymmetric, meaning that the cost traversing $v_i, v_j \neq v_j, v_i$.

We have $M = \{1, .., m\}$, with $1 \leq m \leq n$, being the fleet size available at the depot, i.e. the number of vehicles. For each type $k \in M$, then m_k have capacity equal to Q_k. The vehicles are said to be homogeneous if $Q_k = Q_l$ $\forall k \neq l \in M$, otherwise heterogeneous. The vector of the route of vehicle i where $v_0^i = v_{k_i+1}^i = v_0$, $v_j^i \neq v_l^i$, $0 \leq j < l \leq k_i)$ is given by $R_i = (v_0^i, v_1^i, ..., v_{k_i}^i, v_{k_i+1}^i)$, thus each vehicle starts and ends at the depot. Note that k_i is the length of the route R_i. Then $S = \{R_1, R_2, ..., R_m\}$ is the set of routes representing the solution, where $r(S)$ corresponds to the minimum number of vehicles needed to serve set S. Lastly, the cost of route R_i is $C(R_i) = \sum_{j=0}^{k_i} c(v_j^i, v_{j+1}^i)$, thus $C(S) = \sum_{i=1}^{m} C(R_i)$ is the total cost of solution S, which satisfy $R_i \cap R_j = \{v_0\}\ \forall R_i$ and R_j where $(1 \leq i,\ j \leq m,\ i \neq j)$ and $\cup_{i=1}^{m} R_i = V$, so that each customer is served exactly once.

Now, we may formulate the VRP as an integer minimization problem using a two-index vehicle flow formulation as

$$\min \sum_i \sum_j c_{ij} x_{ij} \tag{3.1}$$

subject to

$$\sum_{i \in V} x_{ij} = 1 \quad \forall\ j \in V \backslash \{0\} \tag{3.2}$$

$$\sum_{j \in V} x_{ij} = 1 \quad \forall\ i \in V \backslash \{0\} \tag{3.3}$$

$$\sum_{i \in V} x_{i0} = M \tag{3.4}$$

$$\sum_{j \in V} x_{0j} = M \tag{3.5}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall\ S \subseteq V \backslash \{0\},\ S \neq \emptyset \tag{3.6}$$

$$x_{ij} \in \{0, 1\} \quad \forall\ i, j \in V. \tag{3.7}$$

Using this formulation, c_{ij} represents the cost of going from vertex i to vertex j, x_{ij} is a binary variable that values 1 if the edge going from i to j is considered as part of the solution and 0 otherwise.

The objective function is defined by (3.1). The constraints (3.2) and (3.3) explicitly state that exactly one edge enters and exactly one leaves each vertex, respectively. That is, each vertex is associated with a customer. The constraints (3.4) and (3.5) impose the degree requirements for the depot which states that the number of vehicles leaving the depot is the same as the number entering, i.e. each vehicle must also return. Constraint (3.6) is the capacity cut constraints, which impose connectivity of the solution routes, hence $G(V, E)$ being a directed graph, thus forbidding solutions consisting of several disconnected tours and that the demand on each route must not exceed the vehicle capacity. Finally, the constraint (3.7) is the integer constraint.

The second latter constraint imposes both connectivities of the solution and the vehicle capacity requirement. It requires that each cut $(V \backslash S, S)$ defined by some customer set S is crossed by a number of edges, that is not smaller than $r(S)$, which was the minimum number of vehicles needed to serve set S. The constraint, therefore, ensures that no small trips occur in a vehicle route and the number of vehicles for each route is sufficient (Toth and Vigo, 2014).

3.6 Exact Methods

Efforts have been made to provide exact solutions to the VRP. However, due to the complexity of VRP being $\mathcal{NP}$-hard, exact solutions are quite unconventional and search-based algorithms and heuristics are often preferred. Exact methods are more useful in facilitating the theoretical understanding of VRP than in providing methods to solve practical routing problems. This is mainly due to applied VRP instances usually involves hundreds of customers, while containing richer constraints than those modeled in classic VRP (Toth and Vigo, 2014).

One of the first instances of exact methods is due to Christofides and Eilon, who provided a branch and bound algorithm for exactly solving the VRP (Christofides and Eilon, 1969). However, the algorithm's computational time was so large that the algorithm only solved the problems of 13 customers. The algorithm was later improved upon in 1976 using a different branch and bound technique, allowing solving for up to 31 customers.

For CVRP, one of the most successful exact methods is Fisher's k-tree method, which successfully solved 71 customer problems from 1994 (Fisher, 1994). It uses the branch and bound procedure to partition problems by fixing the marginal incidence of a subset of selected cluster customers. The edge constraints are then dualized to obtain Langrangian relaxation, which is a k-tree problem with minimum degree constraints.

However, in recent years, Archetti et al. have been applying exact algorithms to the IRP, which incorporated inventory and holding costs, which allowed finding exact solutions within 2 hours of computation time for instances with 50 customers (Archetti et al., 2007). Even though the problem of On-Demand waste management is similar to the IRP, except being the direct reverse logistic version on an IRP, a similar computation time would not be preferred in any case. Moreover, a waste collection problem usually involves a much larger number of customers to serve, thus increasing the time computation time of such an algorithm intensively. Therefore, using an exact method for solving an On-Demand waste management schedule is not preferable.

3.7 Constructive Heuristics

The size of $\mathcal{NP}$-hard problems that can be solved, optimally, using mathematical programming or combinatorial optimization techniques may be limited. Therefore, heuristics that are able to scale to large problem instances are advantageous to compromise on optimality, in favor of fast running times, rather than solving using exact methods. Firstly, we look at the construction heuristics, where the starting point is an empty solution.

3.7.1 The Savings Heuristic

Clarke and Wright's Savings heuristic is one of the most well-known and studied heuristic algorithms in the capacitated VRP. The Savings heuristic belongs to the constructive methods, in which tours are built up by adding nodes to partial tours or combining sub tours, regarding restrictions on both capacities and costs.

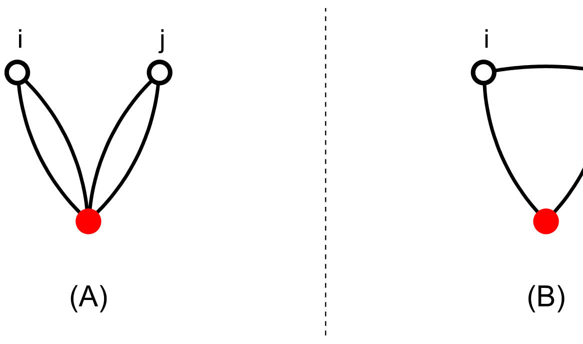

Figure 3.3: *Illustration of the savings concept. The bottom red dot is the depot.*

The heuristic is based on the simple proposition of iterative combining routes in order of those pairs that provide the largest saving (Clark and Wright, 1964). That is, the heuristic itself is based on the principle of savings. The idea is to have two different routes $(v_0 \rightarrow ... \rightarrow v_i \rightarrow v_0)$ and $= (v_0 \rightarrow v_j \rightarrow ... \rightarrow v_0)$ that are feasible for a merge in a new route $(v_0 \rightarrow ... \rightarrow v_i \rightarrow v_j \rightarrow ... \rightarrow v_0)$. The fusion of these routes generate a cost saving S_{ij} as illustrated in figure 3.3. In (A), two different routes will be driven to reach both i and j. An alternative route (B) would be to visit both i and j on the *same* route. The saved distance utilizing route (B) rather than (A) is simply the distance between these two. That is, we denote the transport costs of (A) as seen in figure 3.3 by

$$D_a = c_{0i} + c_{i0} + c_{0j} + c_{j0},$$

while similarly denoting the transport costs of (B) as

$$D_b = c_{0i} + c_{ij} + c_{j0}.$$

Thus, if we combine these two routes, we achieve a savings of

$$S_{ij} = D_a - D_b = c_{i0} + c_{0j} - c_{ij}. \tag{3.8}$$

Now, the relatively large cost-savings are attractive, as most transport costs can be saved by visiting both i and j on the same route. The heuristic, therefore, strives to put together a route consisting of the biggest savings it can find. Thus, the heuristic is characterized as greedy, hence the construction of routes depends solely on striving for the largest possible savings.

The Savings heuristic comes in two versions, namely the parallel version and the sequential version. However, both versions contain the same basic steps, as well as the computation of the savings for each $i, j \in V \backslash \{0\}$ using the same notation as equation (3.8). In the parallel version, more than one route will be built at a time. This means that the parallel version builds upon multiple routes each iteration using the best feasible merge. This iterative computation from the parallel version can be seen as pseudocode in algorithm 1.

The sequential version on the other hand ensures that only one route will be built at a time, hence extensions of the first route obtained by a feasible merge of the edges with the

Algorithm 1: Parallel version of Savings

```
Input: List of savings in decreasing order, $n$ initial routes
Output: Route set
1 foreach $i, j \in savingsList$ do
2     if $feasibleMerge(R_i,R_j)$ then
3         Merge($R_i$,$R_j$)
```

largest savings are made. The sequential version extends the first route found, which will be $(v_0 \rightarrow v_i \rightarrow v_j \rightarrow v_0)$ until it can no longer be extended before a new route is needed. This iterative computation is presented as a pseudocode in algorithm 2.

In order to warrant for legitimate routes, the feasible constraint in both versions of the Savings heuristic ensures that a merge is permissible if and only if: The two associated delivery locations are not already on the same route, the delivery locations are still directly connected to the depot, and either the capacity limit Q or the distance limit D have not been exceeded. Note that in terms of waste management, the distance limit D is often overlooked.

In both versions of the Savings heuristic, the number of vehicles required depends on how many routes are formed in total. Hence if the output's route set is only a single route, only one vehicle is needed. The slight difference between the versions is in the selection strategy and the number of routes built at each step of the algorithm. It is yet worth noting that the number of routes may be reduced during the iteration process of the parallel algorithm (Lysgaard, 1997).

Depending on the way the heuristic are implemented, the parallel version may involve more computational work and complexity, hence it is merging several routes at the same time (Lysgaard, 1997). Thus, it cannot be immediately concluded whether the sequential or parallel version is most appropriate. It should however be noted that there will often be found a more cost-effective solution in the parallel version (Toth and Vigo, 2014). Yet, both versions have time complex-

Algorithm 2: Sequential version of Savings

```
Input: List of savings in decreasing order, $n$ initial routes
Output: Route set
1 foreach $i, j \in savingsList$ do
2     if $feasibleMerge(R_i,R_j)$ then
3         Merge($R_i$,$R_j$)
4         foreach $k, l \in savingsList$ do
5             if $feasibleMerge(R_k,R_i)$ then
6                 Merge($R_k$,$R_i$)
7             else if $feasibleMerge(R_j,R_l)$ then
8                 Merge($R_j$,$R_l$)
```

ity of $\mathcal{O}(n^2)$ on the calculation of the savings, while sorting the list has a time complexity of $\mathcal{O}(n \cdot log(n))$, which results in overall time complexity of $\mathcal{O}(n^2)$.

3.7.2 Criticism and Modifications of the Savings

Modifications are made to improve the interaction by which the routes are made, thus minimizing the routes' sum of distances, but at the same time also reducing the time spent on the actual route planning.

Criticism of the Savings heuristics evolves since it tends to produce peripheral routes. One way to reduce this effect is to modify how the savings are calculated in equation (3.8) by adding a parameter λ which would affect the routes directly as suggested in (Gaskell, 1967). This simple extension is thus calculated by

$$S_{ij} = c_{i0} + c_{0j} - \lambda c_{ij}, \quad 0 \leq \lambda \leq 3. \tag{3.9}$$

Introducing a λ-weight to put more emphasis on the edge between two nodes makes the route itself more radial than circumferential, i.e. routes are being diverted from the depot. The reasoning behind this weight addition is that it is only more appropriate to consider outlying nodes first and thus avoid long routes to individual customers. The problem is however in what specific value the parameter in a practical study should assume. Yet, studies have shown that no value of the parameter could be found that is significantly better than others (Rand, 2009). However, grid search methods may simply be relevant for usage today if overfitting issues do not apply.

A further extension of (3.9) is introduced by Paessens in 1988 where its two-parameter approach to (3.8) provides better results. This modification have no clear motivation provided, yet the addition of a ν term promotes the relative distance between each node i and j from the depot (Paessens, 1988). This two-parameter savings is calculated as

$$S_{ij} = c_{i0} + c_{0j} - \lambda c_{ij} + \mu |c_{i0} - c_{0j}|, \quad 0 \leq \lambda \leq 3,\ 0 \leq \mu \leq 1. \tag{3.10}$$

However, Altinel and Öncan observed that the merging of routes with equal or nearly equal savings occurs frequently, particularly towards the end of the interaction (Altinel and Öncan, 2005). It was therefore proposed that a three-parameter variation of the savings as

$$S_{ij} = c_{i0} + c_{0j} - \lambda c_{ij} + \mu |c_{i0} - c_{0j}| + \frac{\nu(d_i + d_j)}{\bar{d}}, \quad 0 \leq \lambda \leq 3,\ 0 \leq \mu \leq 1, \tag{3.11}$$

is ideal since it further considers the customer demands when calculating savings. Here we have d_i being the demand of customer i, $\bar{d}$ is the average customer demand, and ν is a yet a new parameter. The new parameter ν weights the relative importance of those customers with large demands. That is, the higher the parameter value, the earlier the customer with large demands are merged to a route. In the context of waste management where the logistic is reverse, the large demands is equivalent with those households with a high filling level.

Another way of modifying the Savings heuristic to neglect its greediness to determine peripheral routes is to iterative search for more cost-efficient solutions than the first obtained. Holmes and Parker proposed an algorithm that iterative suppressed the largest saving, in which it is possible in some cases to gain more cost-efficient routes (Holmes and Parker, 1976). Simply by suppressing the largest saving on the savingslist, i.e. by letting $S_{ij} = 0$, and then determine routes as usual. The first merge will thus be the otherwise second-largest saving obtained and a new route combination will be chosen if it is more cost-effective than initially. This suppressing method can be generalized by repetitively suppressing the next largest saving on the savingslist. It is, therefore, possible to obtain various solutions, and even more if not just a single saving is suppressed but instead the $2, 3, ..., n$ largest savings.

Further criticism of the Savings heuristic is based on the premise that customers who are to be added to a route can only be inserted between the depot and the first and last node on each route. Thus, Mole and Jameson considers whether a customer can be inserted between already routed customers (Mole and Jameson, 1976). That is, for each customer that is not already routed, the insertion cost of k between i and j must be calculated as

$$S_k(i, j) = \lambda c_{0k} - (c_{ik} + c_{kj} - \mu c_{ij}). \tag{3.12}$$

If λ and μ is for instance assumed to be respectively 2 and 1, we obtain the same modification as suggested by Gaskell in (Gaskell, 1967). Yet, this savings is the difference in visiting k separately and visiting i and j in sequence. This modification of savings however includes huge computation times since it leads to many iterations determining which i and j that each insertion of k is the most cost-efficient.

Conclusively, despite various criticism and modifications as suggested, the Savings heuristic is still preferred to generate initial routes for a large number of nodes very fast, making this heuristic desirable considering its simple configuration both with or without modifications. The Savings heuristic outperforms other heuristics for initial solutions even at large-scale data sets as seen in (Bard et al., 1998). Problems of large customer sizes are indeed relevant for waste collecting as the sizes of these problems are proportionately large in practice as well. For our usage of the Savings heuristic, we stick to the parallel version without modifications, as it already is proven to fast provide competitive initial routes, while peripheral routes are attractive considering On-Demand waste collection.

3.7.3 Nearest Neighbor Heuristic

A trivial but intuitive constructive heuristic is the Nearest Neighbor principle. In the VRP context where the route begins at the depot, each step of the heuristic ensures the customer nearest to the latest routed customer is chosen, thus routes are built sequentially. This continues until the route reaches its maximum capacity, at which point a new route is started. The number of vehicles for this heuristic, therefore, depends on the number of routes constructed.

Yet, the Nearest Neighbor heuristic tends to provide poor results and is rarely used in practice.

Algorithm 3: Nearest Neighbor

```
   Input: A weighted graph $G = (V, E)$
   Output: Route set
1  foreach $i \in V,\ i \notin R_m$ do
2      $c_{0,i} = \arg\min\{c(0, i) : i \in V\}$
3      if feasibleMerge($R_m, i$) then
4          $R_m$ = Merge (i) after (0)
5          foreach $j \in V,\ j \notin R_m,\ i \neq j$ do
6              $c_{i,j} = \arg\min\{c(i, j) : i, j \in V\}$
7              if feasibleMerge($R_m, j$) then
8                  $R_m$ = Merge (j) with (i)
```

Its greediness often results in just locally optimized route connections, which may provide long detours that otherwise would be avoided if it optimized globally. By globally optimizing in VRP context means that there may be a need for a poor link choice before the very good link choices are built afterward. The Nearest Neighbor tends to route the first few vehicles to closely located nodes. In the case of problems with a large number of nodes, the last routed vehicles may be routed to remote customers and thereby very periphery routes are built. The time complexity of this algorithm is however $\mathcal{O}(n^2)$, making it desirable for utilizing in a large-scale network. In algorithm 3, we present the pseudocode for a VRP orientated Nearest Neighbor heuristic.

The Nearest Neighbor heuristic can be adapted to handle insertion, in which case it transforms into an iterative improvement heuristic instead. Rather than letting the heuristic iterative expand its route to the next nearest node, it will then insert the nearest node k, which is not in a route, nearest to a node i already in a route. Thus, insertion based on minimizing $c(i, k)$. The insert will either result in (i, k) or (k, i) depending on which option is the most cost-efficient. In algorithm 4, we present the pseudocode for a VRP orientated Nearest Insertion heuristic.

Algorithm 4: Nearest Insertion

```
   Input: A weighted graph $G = (V, E)$
   Output: Route set
1  foreach $i \in R_m,\ k \notin R_m$ do
2      $c_k^{insert} = \arg\min\{c(i, k) : i \in V\}$
3      if feasibleMerge($R_m, k$) then
4          $R_m$ = Insert $k$ next to $(i)$
```

3.7.4 Cheapest Insertion Heuristic

The Cheapest Insertion heuristic also belongs to the construction methods and its strategy is also based on pure greediness. The heuristic is similar to the Nearest Insertion, however, instead of finding the node k closest to any node i already in a route that minimizes $c(i, k)$, the

Cheapest Insertion heuristic seeks to minimize a detour in between node i and j, similar to the concept of savings.

```
Algorithm 5: Cheapest Insertion
   Input: A weighted graph $G = (V, E)$
   Output: Route set
1  foreach $i, j \in R_m,\ k \notin R_m$ do
2      $c_k^{insert} = \arg\min\{c(i,k)+c(k,j)-c(i,j) : i,j \in V,\ i \neq j\}$
3      if feasibleMerge($R_m, k$) then
4          $R_m$ = Insert $k$ between $(i, j)$
```

Originated from TSP, this heuristic starts by finding the node among all nodes not inserted so far, which insertion in the route causes the smallest increase in cost. Within the VRP context, this is replaced by finding the node which insertion in any route feasible for a merge causes the smallest increase in cost. This insertion's cost is thus based on the same idea as Saving's (3.8), however, the depot is no longer necessarily the main node. The heuristic then continues to insert all nodes until have been visited (Rosenkrantz et al., 1977). Nevertheless, the time complexity of this algorithm is also $\mathcal{O}(n^2)$, making it desirable for utilizing in a large-scale network. This heuristic's pseudocode for a VRP-setup is seen in algorithm 5.

3.7.5 Farthest Insertion Heuristic

Another heuristic also belonging to the constructive methods is the Farthest Insertion heuristic. It is the opposite of the Nearest Insertion heuristic in which it will select the farthest node k, not in any route to node i already in a route for insertion. This selection is thus searching for the least cost-efficient detour. However, the insertion will then be the edge (i, j) in a route that maximizes a detour in between node i and j, similar to the concept of savings too.

```
Algorithm 6: Farthest Insertion
   Input: A weighted graph $G = (V, E)$
   Output: Route set
1  foreach $i, j \in R_m,\ k \notin R_m$ do
2      $c_k^{insert} = \arg\max\{c(i,k) + c(k,j) - c(i,j) : i,j \in V,$
3      i $\neq j\}$
4      if feasibleMerge($R_m, k$) then
5          $R_m$ = Insert $k$ between $(i, j)$
```

This Farthest Insertion heuristic seems a bit counterintuitive at first since it is choosing the node maximizing the detour, thus encouraging peripheral routes. However, the Farthest Insertion works well in practice as seen for instance in the paper of Rosenkrantz et al., where multiple 50-node problems are tested. Here the Farthest Insertion achieves up to 22% and 7% better results compared to the Nearest and Cheapest Insertion heuristic respectively (Rosenkrantz et al., 1977). The time complexity of this algorithm is likewise $\mathcal{O}(n^2)$, also making it desirable for utilizing in a large-scale network. Lastly, the pseudocode for a Farthest Insertion heuristic in a VRP is presented in 6.

Conclusively, these three insertion heuristics may look similar in terms of the selection process of container k. However, they do all perform differently in terms of end results. It is therefore difficult to point out which one is the better performer in the waste management context. In figure 3.4, we provide an example of how a route will merge, entirely depending on how the node k is located relative to node i and j. Yet, the three insertion heuristics all serve the same purpose of inserting a container into the route.

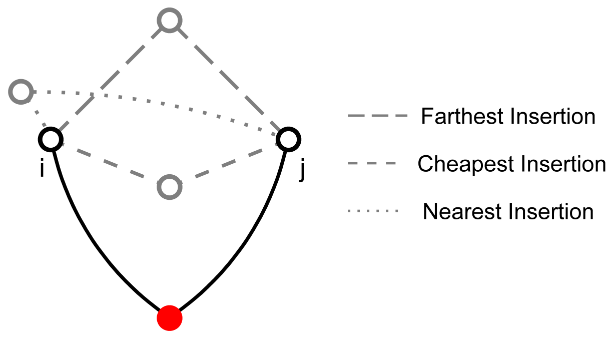

Figure 3.4: *Illustration of the difference between three insertion heuristic's choice of a grey-colored node. Only one grey-colored node is to be inserted in which the respective route is built. The depot is the red dot at the bottom.*

3.8 Two-phase Heuristics

We next look at two-phase heuristics, where particularly two subfamilies emerge. Firstly, the cluster first, route second-subfamily is a method where the points are first divided into a series of clusters. That is, points that are geographically reasonably close. Then, for each cluster, a route must be determined where all of the cluster's nodes are visited.

The second subfamily is the route first, cluster second-method, which starts differently by first determining one grand route, where all nodes are visited, and then dividing this route into smaller trips. However, such methods are generally thought to be less competitive than methods within the cluster first subfamily (Laporte et al., 2000).

3.8.1 The Sweep Algorithm

One of the profound algorithms for the cluster first, route second subfamily is introduced by Gillett and Miller who provided a new approach called the Sweep algorithm (Gillett and Miller, 1974). Based on polar coordinates, one may define a sweeping-line, e.g. a ray, that is horizontal in relation to the depot. Feasible clusters are initialized by enclosing nodes by rotating the sweeping-line centered at the depot clockwise using the nodes' angle from the nodes to the depot. That is, the angle in polar coordinate between the depot and each node is defined as

$$\theta = \arctan\left(\frac{y}{x}\right) + V \cdot 2\pi, \quad x \neq 0, \tag{3.13}$$

where V may assume 0 if $x > 0$ and $y > 0$, 1 if $x > 0$ and $y < 0$, or 1/2 if $x < 0$, to ensure $\theta \in \{0, 2\pi\}$. The enclosing seen in waste management context is often the CVRP variant, where the sweeping-line may rotate among the household's containers until the garbage truck simply can not carry any more waste from the enclosed containers.

When all nodes have been divided into clusters as seen in figure 3.5, the routes within each cluster are treated as separate TSP instances and optimized accordingly. This two-phase approach does not prescribe a method for how the individual TSPs are solved. Instead, it assumes that already developed TSP methods can be used.

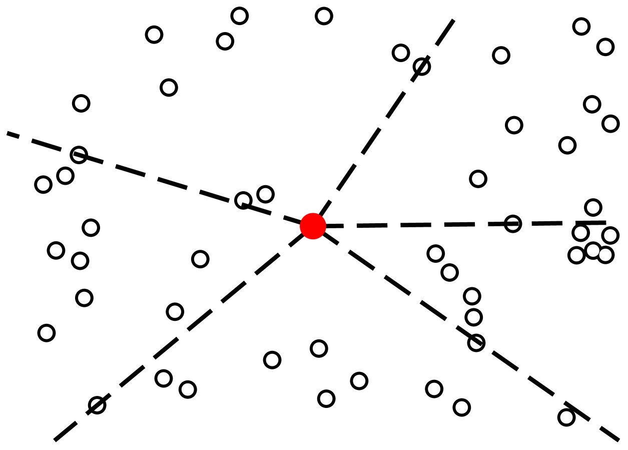

Figure 3.5: *Example of a clusterformation for five vehicle routes. Each cluster is represented by the dashed black sweeping-lines' divisions. The depot is the red dot in the center.*

The Sweep method is surprisingly effective and has been shown to solve several benchmark VRP problems to up to 2% of best-known solutions (Toth and Vigo, 2014). However, the Sweep algorithm is rather unpractical when a large number of customers are introduced. This is due to the algorithm's negligence of distances, where clusters initialized are based on greediness, since the clustering is made by radial rays. Some nodes that are indeed located very close to each other, must be visited by different vehicles due to the rays' division. Thus, the creation of multiple peripheral routes could be avoided simply by merging closely located nodes that are split into different clusters.

The Sweep algorithm may still be an ideal choice as a heuristic for waste management hence the deterministic nature of traditional waste collection in municipalities. The municipality is usually divided into several areas, so the collectors empty within an area every day, which is the direct relation to the PVRP. The Sweep is therefore attractive since it may divide the area based on geometric foundations. Yet, issues and obstacles arise in an On-Demand setup where requests of emptying must be fulfilled within a certain period provided by a policy. The division made by the Sweep algorithm should consider this when applied. It would otherwise risk making vehicles traverse far from its area to comply with the request of a single household in another area. The Sweep algorithm thus lacks the ability to generate reasonable routes for On-Demand waste management.

Fisher and Jaikumar build upon the cluster first, route second approach by contributing with a General Assignment Problem (Fisher and Jaikumar, 1981). This assignment problem in the VRP context chooses m nodes to the m vehicles available and thereafter assigns nodes to whichever cluster whereby the traveling costs associated with insertion are minimal. However, one limitation of this method is that the number of vehicles needed to traverse the routes must be fixed in the beginning to initialize clusters. Still, this method produces results that are up to 2% better compared to the regular Sweep algorithm. The advantage is found in the fact that cluster division thus considers transport costs, i.e. the distances, opposite to the Sweep algorithm.

One instinctive extension to the Sweep algorithm is the Petal algorithm (Ryan et al., 1993). The strategy is here to produce a collection of overlapping candidate routes, which then are called petals, and then to solve the partitioning problem to produce a feasible solution. Yet, the petal method has produced competitive results for small problems but quickly becomes impractical where the set of candidate routes that must be considered is large since the computational requirements needed to emerge (Toth and Vigo, 2014).

Today, clustering methods have been suggested as cluster first route second-approaches. These methods often consider centroid-based clusterings such as k-means and k-medoids, which origins from the main task of exploratory statistical data analysis seen in data science and machine learning. Cluster analysis is based on the principle of maximizing the similarity of groups among themselves and minimizing group similarities. Thus, the idea is in VRP context to group nodes based on e.g. distances and density. This evidently helps the Sweep algorithm's disadvantage of creating multiple peripheral routes. Case studies show that the k-medoids clustering techniques where the center of sets are specified as the closest point to the center of the sets are superior compared to k-means, where the centers are specified as the center of the sets themselves, despite k-medoids being more sensitive to noisy data (Comert et al., 2018).

3.9 Iterative Improvement Heuristics

Iterative improvement heuristics are often used in combination with other heuristics. In this case, they are run on the candidate solution after the initial heuristic. Iterative improvement heuristics are left operating on the proposed candidate solution persistently in the search for a better solution.

3.9.1 Two-Opt Algorithm

Solutions that are obtainable from the current candidate solution, S, can be found within S's neighborhood. That is, operators deployed onto S using iterative improvement heuristics may find new feasible solutions S' by searching the entire neighborhood of S. One of the best known improvement operators is Two-Opt. This operator takes two edges $(i, j), (k, l) \in T$, where T are the edges traversed by $R_m = \{v_0, ..., v_i, v_j, ..., v_k, v_l, ..., v_0\}$, and removes these from the candidate solution. This separates the route into two disconnected components, $D_1 = \{v_j, ..., v_k\}$ and $D_2 = \{v_0, ..., v_i, v_l, ..., v_0\}$. These disconnected components are also called chains. A new candidate solution is produced by reconnecting D_1 to D_2 using the same vertices i, j, l, k but with different edges such that $(i, k), (j, l) \in T$ as seen in figure 3.6.

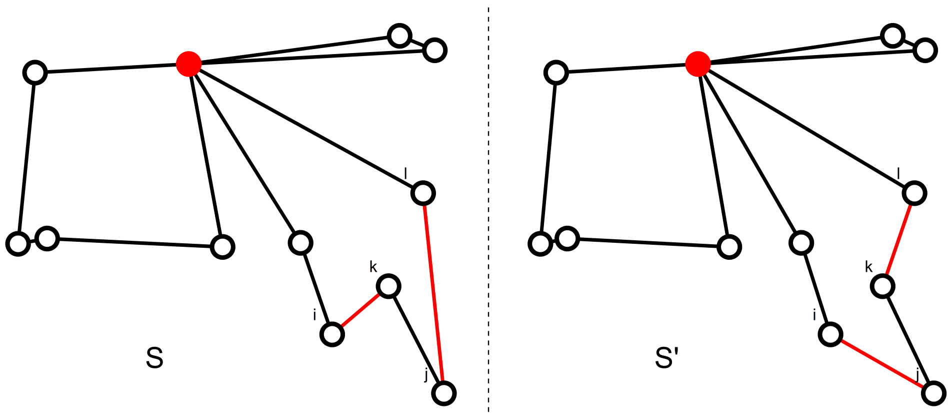

Figure 3.6: *An example of Two-Opt applied to a candidate solution S on the left. The new candidate solution S' on the right exchanges edges $(i, j), (k, l)$ with edges $(i, k), (j, l)$. The depot is the red dot in the center.*

Note that figure 3.6 shows an example of Two-Opt being performed on the same route. This means that no feasible constraint has to be checked, since the operation takes place within the same route. The time complexity for this algorithm is $\mathcal{O}(n^2)$ since to consider all pairs of edges is $\mathcal{O}(n^2)$, then reversing edges is $\mathcal{O}(n)$, thus resulting in overall time complexity of $\mathcal{O}(n^2)$. The pseudocode for this Two-Opt algorithm with application to a VRP-setup is provided in algorithm 7.

The Two-Opt algorithm applies to any VRP context, where two components from different routes may be connected as seen in figure 3.7. In practice, such components are often connected by splitting two routes into four disconnected components and then reconnecting them two-by-two with one component in order and the other component in reverse order. When applying the Two-Opt algorithm to different routes, then the feasibility constraint must be checked since the operation will not be applied if the newly suggested solution is not feasible. Yet, the motivation behind this Two-Opt is that if two edges in a tour cross, you may often decrease the cost of the tour simply by replacing the pair that crosses with a pair that does not cross. The operations made thus aims to detangle a route since routes with edges that cross themselves are very unlikely to be optimal. This is the case of the example in figure 3.7.

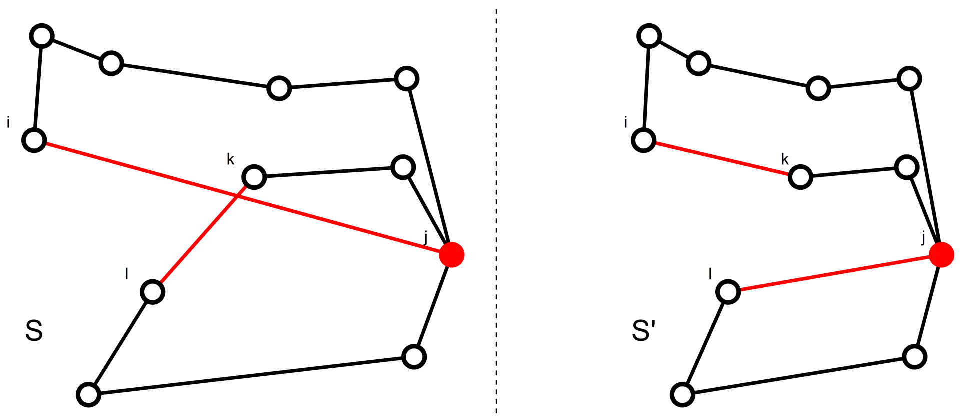

Figure 3.7: *Example of Two-Opt applied to different routes in a candidate solution S on the left. The new candidate solution S' is seen on the right. This operation may only be applied if the new candidate solution S' is feasible. The depot is the red dot on the right.*

One must note that this operation is indeed computational expensive. Thus, limits to only searching in neighborhoods have been suggested e.g. by Osman, where a λ-Opt is called together with a simulated annealing process to ensure that the algorithm converges (Osman, 1993). The idea behind such simulated annealing is that a search-based meta-heuristic accepts solutions with a probability that are both better and worse. Thus, the advantage is non-greediness since a candidate solution may not get stuck in local optima. The algorithm may then converge, and not accept any worse solutions, whenever a probability function as a function of temperature and time is exceeded.

Another algorithm that also limits the search to the neighborhoods is proposed by Gendreau et al. for VRP applications, where two functions are called and combined to form a TabuSearch (Gendreau et al., 1994). The first function is a generalized insertion procedure, which simply limits the search for nearby nodes in the neighborhood defined by distance. The second function is the Two-Opt algorithm, which may only apply to swapping nodes on different routes. Since it is a search-based meta-heuristic it will also accept solutions that are both better and worse. However, in this algorithm, feasible solutions are stored for θ amount of iterations on a *TabuList*, such that it may keep track of local optima.

However, we combine the two techniques of Osman and Gendreau et al. for our implementation of the Two-Opt. Our implementation will continue to search for improvements until no improvements are possible. Note that for this explicitly, the algorithm converges when no improvement is found in both single-node swaps and chain swaps. Yet, it cannot be neglected that other specific parameter settings may become useful in terms of concrete time complexity in some instances.

Lastly, many other operations have been suggested such as the generalization of Two-Opt to Three-Opt by Christofides and Eilon which produced exceeding results compared to Two-Opt (Christofides and Eilon, 1969). In a Three-Opt algorithm, the number of edges that are removed

Algorithm 7: Two-Opt algorithm

```
   Input: A weighted graph $G = (V, E)$
   Output: Route set
1  improvement = False
2  while improvement is False do
3      restart:
4      bestDistance = totalRouteDistance(S)
5      foreach $i \in V - 1$ do
6          foreach $k \in V - 1,\ k = i + 1$ do
7              newRoute = 2optSwap($S, i, k$)
8              newDistance = totalRouteDistance($newRoute$)
               if newDistance < bestDistance and
               feasibleMerge(newRoute) then
9                  S = newRoute
10                 bestDistance = newDistance
11                 improvement = True
12                 goto restart
```

and reconnected increases from two to three. However, the downsides of generalizing to such multiple Opts-strategy is that producing these operations leads to large neighborhood searches and thus large time complexity, which is strictly poor for waste collection problems. Ideas to reduce this have been presented in terms of e.g. Or-Opt, which is a restricted Three-Opt where relocation of all sets of 3, 2, and 1, consecutive vertices, i.e. nearby nodes, are used as a local neighborhood instead (Or, 1976). Yet, such a method does not provide the same accuracy for best-known solutions on samples compared to non-restricted Opts.

Chapter 4

Waste Container Modeling

The second objective within On-Demand waste management is to consider the refilling of inventory decisions that are based on various inventory levels and supply chain policies. In this chapter, we present a reverse base stock model that models the fillings of waste containers to effectively minimize the frequency of emptying.

4.1 Filling of Waste Containers

The inventory level is in reality not constant but fluctuates over time (Snyder and Shen, 2019). Therefore, we introduce a reverse base stock model to model the filling of a waste container, since the filling is stochastic, hence it may vary over time. An example of a waste container's fill-level is seen as a stochastic process is seen in figure 4.1. The waste container's fill-level is here seen as a function of time, where it returns to a level of 0 throughout the session with the same interval. This means that the waste container is being emptied on a static predetermined waste collection schedule.

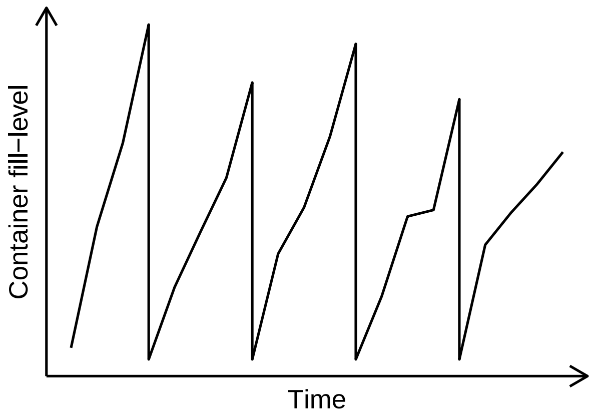

Figure 4.1: *Example of a waste container's fill-level curve.*

Note that the slope of each spike is different, i.e. the curve has some fluctuations and inconsistency, yet always positive. Each spike's height is also not the same, hence the waste fillings are most likely different from day-to-day. We say that the container has a continuous flow of waste which is similar to a stochastic process.

For estimation of the waste filling curves, we turn to Glostrup's pilot test. Figure 4.2 shows the 866 observations of intervals between requests that occurred during Glostrup's pilot project.

Note that the two groups of residential can be united since they indeed look similar in terms of intervals between requests. An unpaired two-sample Wilcoxon rank-sum test as a non-parametric approach is used to accurately compare these two independent groups each with a sample space. The test statistic of 90895 with a p-value of 0.4519 suggests that the null hypothesis is not to be rejected even at higher significance levels, meaning that the two groups can not be said to be different from each other. This means that it is in fact responsible to merge the two groups to form a single larger sample space, which we will utilize going forward.

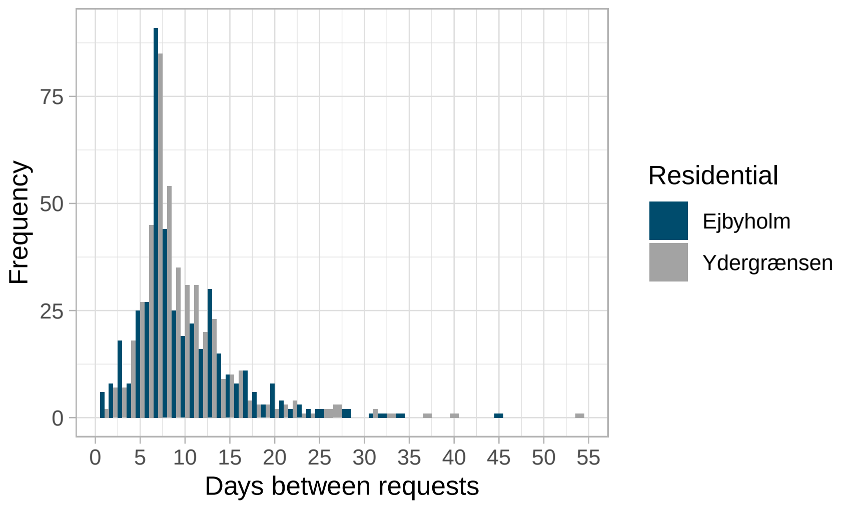

Figure 4.2: *Histogram of intervals between request for emptying in Glostrup.*

During this pilot test in Glostrup, 20.3% of the observations kept a time-to-request on 7 days. Meanwhile, 22.9% of all time-to-request occurred before 7 days and 56.8% of all time-to-request occurred after 7 days. This explains both groups' tendency to show right-skewness as seen in the figure 4.2. Note that the frequencies also do vary a lot. In fact, they are varying from 1 to 54 days. This means that the rate of filling differs a lot throughout the residential.

Further, the mean days before a request for emptying occur is at 9.688 days. This corresponds to the interval between emptying is 38.4% longer when On-Demand waste collection is present compared to the static waste collection, where emptyings took place on a specific day once every week in this residential. This also indicates that a large number of residents do follow the static waste collection schedule, where containers are emptied each week regularly as it fits their specific needs.

4.2 Non-parametric Estimation of Time-to-Request

The duration of a waste container's lifespan, i.e. time-to-request, can be modeled using survival analysis. We may estimate the survival function

$$\begin{aligned} S(t) &= P(T > t) \\ &= 1 - F(t) \end{aligned} \tag{4.1}$$

where T is a random variable representing the death with cumulative distribution function $F(t)$ on the interval $[0; \infty)$ (Harrell, 2001). We do not observe the T_i's but rather the censored variables $T_i = \min\{T_i, C_i\}$ with censoring times $C_1, ..., C_n$. These are assumed identically distributed and $T_1, ..., T_n, C_1, ..., C_n$ are independent.

In the case of On-Demand waste management, the censoring condition is when the measurement of a container is only partially known, in which such a situation could occur if the container was emptied without actually being requested beforehand. However, the setup of the pilot test in Glostrup illustrated in figure 4.2 ensures no censoring, since containers were only emptied coherent with a request. For our implemented heuristic, we do allow for emptyings of containers that have not requested an emptying. Thus, using this definition of censoring is attractive.

Now, the observed event time corresponds directly to the day of emptying a waste container in waste management. We determine the survival function, i.e. the probability of having a container not being requested for another day as a function. This is achievable by applying the product limit estimator as a non-parametric frequency-based estimator. That is, given k fully observed event times, we assume that emptying of waste containers may only take place at fully observed event times $T_1 < T_2 < ... < T_k$. With the definition of $T_0 = 0$, d_i is the number of emptyings at T_i, and $Y(T_i)$ is the number at risk. In the case of On-Demand waste management, the number at risk is the number of containers left for potential requesting just before T_i. Then, the hazard rate, i.e. the requesting rate, is estimated by maximum likelihood thus

$$P(\text{emptying at } T_i | \text{not emptying until } T_{i-1}) = \frac{d_i}{Y(T_i)}. \tag{4.2}$$

This rate provides us the conditional survival function estimate

$$\hat{S}(T_i | T_{i-1}) = 1 - \frac{d_i}{Y(T_i)}, \tag{4.3}$$

which for each candidate time gives us

$$\begin{aligned} \hat{S}(t) &= \prod_{T_i \leq t} \hat{S}(T_i | T_{i-1}) \\ &= \prod_{T_i \leq t} \left(1 - \frac{d_i}{Y(T_i)}\right). \end{aligned} \tag{4.4}$$

Note that this maximum likelihood estimation of the hazard rate is nothing but the empirical estimate of the survival function in the case of no censoring, which is the format of Glostrup's pilot test. However, we may utilize this non-parametric estimation from equation (4.4) to consider the non-parametric survival function of time-to-request in figure 4.3.

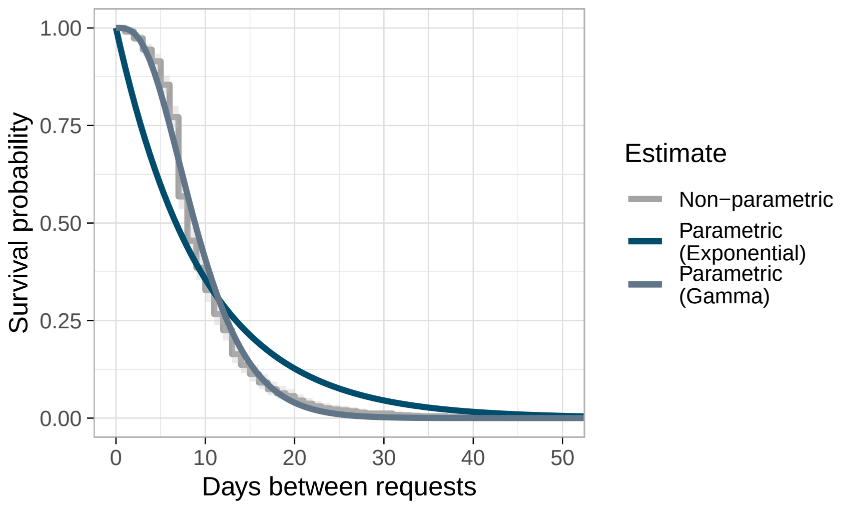

Figure 4.3: *Product limit estimator and parametric estimates modeled to the pilot test in Glostrup.*

In figure 4.3, we see how survival curves estimate the time-to-request in days. The non-parametric estimate includes grey-shaded confidence intervals. This product limit estimator claims a median of 8 days and a mean at 9.688 days before a waste container is emptied. This is exactly what we would expect since this is nothing but an empirical estimate. Lastly, the low survival probability for this non-parametric estimate's container seen after 20 days. This is indicating a fat right tail which is an issue for any probability density function.

4.3 Parametric Estimation of Time-to-Request

The Poisson process becomes relevant for modeling the time that passes between the need of emptying, i.e. the time-to-request. For this Poisson process modeling of the waste container-filling, we assume that the number of events in an interval is Poisson distributed with independent increments, a constant intensity for an event, and intervals between events are exponentially distributed, yet independence assumptions in all cases. Thus, we define a stochastic process as $N(t)$ where $\{N(t), t \geq 0\}$ with the assumptions above. Then, the point process is given by

$$P\{N(t) = n\} = \frac{(\lambda t)^n}{n!} e^{-\lambda t}. \tag{4.5}$$

Now, the following model used to encircle a container's waste-filling is based on the Poisson distribution similarly to (M.Gutierreza et al., 2015). However, in our model, we have that each waste container i is assigned an average time to become full, denoted F_i, where the Poisson average of the distribution is to be estimated. Furthermore, to estimate the waiting times, each container i is assigned a discrete daily filling volume denoted DF_i^d which follows a Poisson distribution with $\lambda = \frac{1}{F_i}$ where d corresponds to each concrete day of the week.

This Poisson point process's properties ensure that waiting times will be modeled as an exponential random variable with a mean of $1/\lambda$. The properties also imply that the points in the process have the loss of memory property, where the existence of one point in a finite interval does not affect the probability of other points existing (Lawler, 2006). Within waste management, if a waste container has not had any filling in s time units, the chance of receiving any filling in the next t time units is the same as if there were fillings before. Mathematically, we formulate this memorylessness-property as

$$P\{T_i \geq s+t \mid T_i \geq s\} = P\{T_i \geq t\}. \tag{4.6}$$

This modeling of a stochastic process allows us to study some variance in terms of how each of the waste containers is being filled, hence each container will follow its own stochastic process.

Now, as we already empirically estimated the λ-parameter of such Poisson process to 9.688 using the pilot test's data in figure 4.2, we may derive that the daily filling volume of each container be 1/9.688. This thus corresponds to an average of 9.688 days before the waste container is said to be full and thus in need of being emptied, else highly increasing the risk of overflow.

By approaching the parametric filling of a waste container in the same manner as the non-parametric, we may conclude that if the survival theory's approach suggests that the hazard rate is constant, then the distribution of the lifetime of a waste container is the exponential distribution with density

$$f(t;\lambda) = \lambda \cdot e^{(-\lambda t)}, \tag{4.7}$$

and survivor function

$$S(t;\lambda) = e^{(-\lambda t)}. \tag{4.8}$$

We may therefore construct a Poisson parametric survival curve from the empirical data as seen in figure 4.3 with λ equal to the mean of 9.688 days. This modeling, however, reveals underestimation in the small intervals, while also managing to overestimate in the larger intervals, due to the assumption of constant hazard rate as a yet simple and direct approach. Therefore, we say that the Poisson distribution is not accurate to describe the distribution's right skewness, hence the huge variance in time-to-requests seen in figure 4.2.

Instead, a Gamma distribution is suggested as a parametric alternative to the Poisson distribution that models the intervals of time to request. An equivalent suggestion is used in (Mes, 2012). Here, a Gamma distribution is computed and applied as a parametric estimation of the arrival process for the number of deposits per day in containers, since this is dependent on the wastes' weight, which is proven by being a significant regressor when modeled in a regression analysis. The problem described by Mes considers underground containers that are equipped with motion sensors, yet with the planning of container emptyings based on static waste collection schedules. This is different from this On-Demand waste management since Glostrup does not employ sensors in containers and the sizes of containers are much smaller. However, these two problems share similar difficulties of parametric estimating the fillings modeled as stochastic processes and thus are comparable.

Now, the Gamma distribution has a density

$$f(t;\alpha,\beta) = \frac{\beta^{\alpha} t^{\alpha-1} e^{-\beta t}}{\Gamma(\alpha)}, \quad t,\alpha,\beta > 0, \tag{4.9}$$

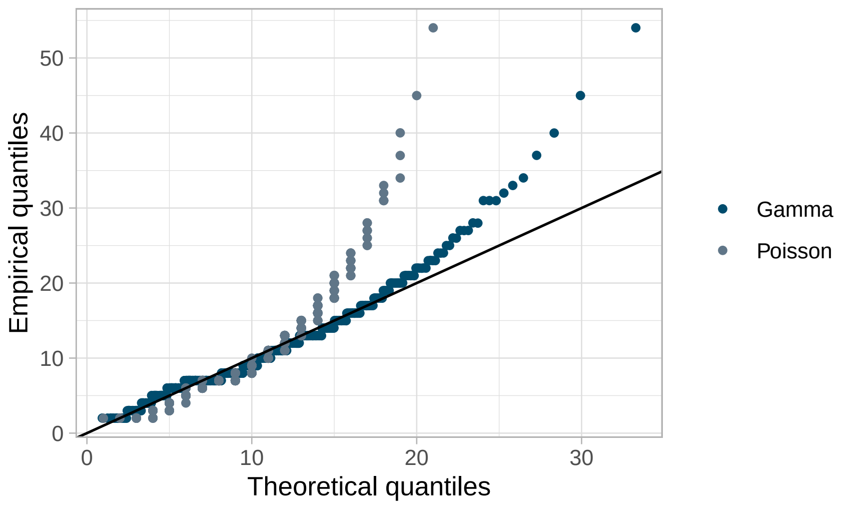

Figure 4.4: *QQ plot of the parametric estimates modeling the intervals between requests.*

using shape α and rate β. The survivor function is

$$S(t;\alpha,\beta) = 1 - \frac{I(\alpha,\beta t)}{\Gamma(\alpha)} \tag{4.10}$$

where $I_k(x)$ is the incomplete Gamma function, which is defined as

$$I(x;\alpha) = \int_0^x z^{\alpha-1}e^{-z}dz, \tag{4.11}$$

in which no closed-form expression exists. Note, if $\alpha = 1$ then the corresponding Gamma distribution is given by the exponential distribution.

Using this, we may construct a Gamma parametric survival curve from the empirical data as seen in figure 4.3. The fitted Gamma distribution is empirically estimated to include the shape parameter of 4.474 and a rate parameter of 0.452 using maximum likelihood estimation consisting in maximizing the log-likelihood.

As a parametric alternative to the Poisson survival curve, this Gamma survival curve is more preferable to accurately describe the empirical data. This is seen in figure 4.4 where both the Poisson and Gamma distribution modeling the time-to-requests are compared in a QQ-plot. This figure illustrates in the upper half of the plot that the Gamma distribution is much better at fitting the previous exposed fat right tail. However, note that either distribution models the fat right tail very well.

Again, using the pilot test's data in figure 4.2, we may compare these two empirically estimated models of respectively Poisson and Gamma using a likelihood-ratio test. The test static of 101.05 evaluated in a χ^2 distribution with 1 degrees of freedom suggests rejection of the null hypothesis of both models being equal. Thus, the Poisson distribution is overlooked since the Gamma distribution may provide a more accurate parametric survivor function while also fitting the theoretical quantiles of the time-to-requests better.

Therefore, whenever we model the estimated filling-level of a container, we turn to the Gamma parametric estimation of the survival curve. In practice, this means that we simply generate a random deviate of this estimation for each container to simulate the individual stochastic process's filling rate.

4.4 Distinction of Waste Containers

Within the dynamic planning of waste management, we daily select containers based on their estimated fill-level. To approach this day-to-day modeling of which waste containers to collect, we make a distinction between the different containers. This distinction is similar related to what is presented in (Mes et al., 2013) and (Omara et al., 2018). Firstly, Mes et al. builds a dynamic inventory routing model, yet it differs from On-Demand waste management in the sense of having residents requesting an emptying compared to only consider the filling-level of the waste containers. Secondly, Omara et al. also presents a dynamic inventory routing model, yet it differs from On-Demand waste management as it considers waste containers equipped with IoT-sensor-based to review the filling-level is introduced. Both models presented nevertheless deals with stochastic routing hence stochastic customers with stochastic demands which is comparable to On-Demand waste management.

A suitable representation distinguishing of waste containers for On-Demand waste management includes the definition of a "MustGo" as a container that has to be planned into a route as a request for emptying has occurred. In addition, a "MayGo" is defined as a container that might be planned into an existing route of MustGo's if it is convenient to do so. The MayGo's that are planned into the routes of MustGo's are of interest in On-Demand waste management, as the MustGo containers must be emptied within a certain number of days. Finally, a "NoGo" is defined as a container that should not be planned to be emptied during the current planning day.

Using these definitions, we do note that the actual size of waste is varying among the customers in On-Demand waste collection, yet it is often very close being of unit size for the MustGo's and lower for the MayGo's. Further note that there is no guarantee that a container is categorized as a MayGo before becoming a MustGo. Now, we may introduce waste containers into a routing schedule based on fill-levels estimated from stochastic processes.

4.4.1 Probability-driven Requesting

Firstly, to determine whether a container is designated to becoming a MustGo container within a simulation setup, we utilize the probability density function of an exponential distribution. This is due to the waiting times of fillings being modeled as random variables since the point process's loss of memory allows us to model events that occur continuously and independently at a constant average rate. That is, we define that any container with a high enough filling-level may be relabeled into a MustGo container. This is reasonable as one might expect a request for emptying to happen whenever the waste container is nearly full. We let the probability of container i being relabeled to a MustGo, i.e. being requested, follow the exponential distribution as a function of the filling-level in percent as

$$f(x; A) = \exp(-A \cdot x), \quad 0 \leq x \leq 1, \; A \in \mathbb{R}_+. \tag{4.12}$$

We may thus evaluate the probability of being relabeled with an estimate of A-parameter. That is, the probability of requesting an emptying in a simulated instance depends on A, where having a large A, means that only almost full containers are relabeled to MustGo. A large A-parameter is attractive for the On-Demand simulation setup since it will incorporate that households may forget to request and thereby increase the chances of overflows. On the other hand, a small A-parameter is attractive to prevent overflows of containers that otherwise would be requested late since the risk of overflows increases as A increases. However, an A-parameter equal to 0 will define a linear dependence between the estimated filling-level and the probability of having the container being requested being 1 which is not rational or authentic.

For simulation purposes, we define a fixed A-parameter that simply integrates all households' preference of when to request based on the fill-level. In figure 4.5, we show how different values of A affect this probability of requesting as a function of the fill-level.

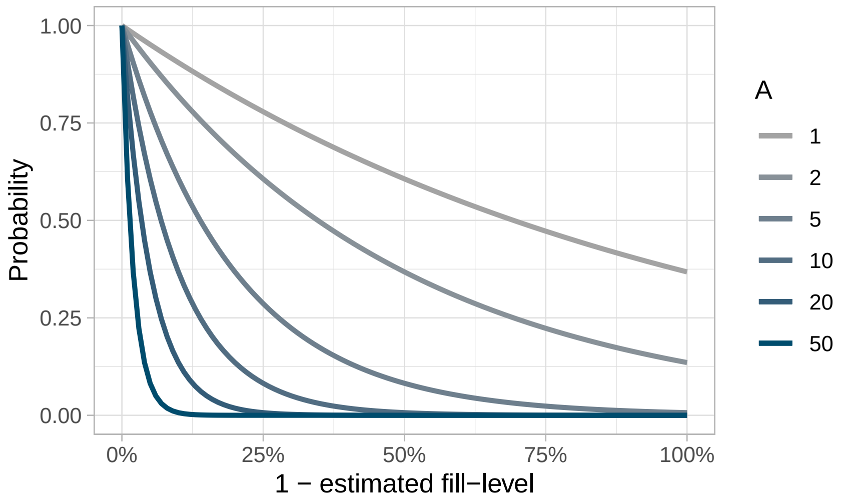

Figure 4.5: *Probability of requesting an emptying in simulation using the exponential distribution's density function for different A.*

4.4.2 The Threshold Level

Secondly, to determine whether a container is convenient to collect, we turn to the reverse base stock model. That is when to label a container as a MayGo in On-Demand waste management. In a stochastic base stock model with a base stock policy, the threshold can be seen as the safety stock level. This level defines the extra inventory that is kept on hand to buffer against demand uncertainty.

In waste management where logistics is reversed, this threshold is the level when a container should be considered for collection. That is, the threshold level determines when the container is said to be convenient to collect. A container with a waste level above this threshold is thereby said to be a MayGo-container. The threshold is thus the level right before we expect a household's request will happen.

If the threshold level is set at 100%, then no MayGo's are considered neither emptied on the routes since the containers will be requested instead. Note that this is equivalent to only considering requested containers, which is the core of the dynamic waste collection schedule and the basis of On-Demand waste collection. On the other hand, if the threshold level is set at 0%, then all containers are considered for collection in an On-Demand planning schedule. However, this opposes the idea of On-Demand waste management, as we would then be collecting many MayGo's instead of actual MustGo's. Yet, the lower the threshold level, the more likely a container is to be emptied before getting requesting, which undeniably reduces the number of MustGo's.

Our heuristic does indeed consider MayGo's for collection hence the introduction of a base stock model. For applications in waste management, we may estimate this threshold to be the average of how full the waste container was when a request was performed by the household. For our proposal, we simply specify a threshold level at 60% since we consider household waste in relatively small containers. This means that we may find it convenient to collect the waste container when we estimate its filling-level of more than 60%.

4.5 Distinction of Costs

As the On-Demand waste management considers the MustGo containers, since they are the ones to be emptied, there may be vehicles that can still load more waste from other containers, once all MustGo's containers have been planned for routing. We then add the MayGo containers to the vehicles' routes to improve the capacity utilization of these vehicles. However, there may also be too many, in which case not all MustGo containers can be emptied.

4.5.1 Penalty Cost

Firstly, we turn to the instance of having too many MustGo containers considered in one day for complete capacity utilization. In such case, it is not possible to empty all requests on that day. Yet, as there is a policy of maximum waiting days before having a request met, there is room to ensure emptying the subsequent days. To ensure this policy is withheld, we add penalties to the routes containing MustGo's that have been waiting for too long. This way, the longer the time the container has been waiting, the more likely the container will be selected as a part of the day's schedule.

If a delay of emptying should occur and the policy is violated, a penalty cost will take place based on a penalty function. We first define the linear penalty function as an expansion to the

VRP's objective function defined in (3.1). This new objective function reflects on the overall costs and may be considered as a trade-off for how willingly the planner is to violate the policy of maximum days waiting, and how efficient the planned routes are. This is because the penalty cost only arises if a route overrules the policy of maximum waiting days for the container to be emptied after a request occurred.

Now, to capture the effect of the longer the days waited, the more penalty should occur. We introduce a polynomial term on the penalty cost P and redefine our objective function as

$$\min \sum_i \sum_j c_{ij} x_{ij} + \sum_{k,t} P^t \cdot v_{k,t}. \tag{4.13}$$

This way, we extended (3.1) by defining a polynomial penalty function that ensures high costs by neglecting the collection of requested containers for longer periods.

4.5.2 Capacity Utilization Cost

Secondly, we turn to the instance of having too few MustGo containers considered in one day for complete capacity utilization. Within the IRP literature, a common way is to define a different cost for a potential customer, since we might add a collection of MayGo containers to an existing route (Mes et al., 2013). To do so, we calculate a cheapest insertion cost c^{insert}_{k,i,j,R_m} equivalent to algorithm 5, of container k between location i and j per sub-route as

$$c^{insert}_{k,i,j,R_m} = c^t_{i,k} + c^t_{k,j} - c^t_{i,j}, \quad \forall\ R_m \in S \wedge i, j \in R_m \mid R_m \subseteq S. \tag{4.14}$$

Now, we also take into consideration the capacity of a vehicle in each sub-route R_m, when deciding where to insert container k. If the additional volume from container k exceeds the maximum capacity of the vehicle on route R_m, then the container is not inserted on the route, since any additional visits to the disposal are not planned. That is, container k will simply not be available for emptying on route R_m, because the waste collectors are limited to a single visit at the disposal area every day.

Although adding MayGo containers increases vehicle utilization, it might also lead to unnecessarily long routes. This is because the set of MayGo containers is not defined by location, but rather the cheapness on a particular day. Also, scheduling of MayGo containers may provide unnecessary long routes if emptyings could be postponed to a subsequent day that is still within the maximum day policy for more efficient waste collection routes. We refer to this cost as the cost of capacity utilization. Suggestions such as bounds and adjustable limits on the MayGo-categorized containers have been proposed as ideas, but not implemented, to control for the inefficiency of unnecessarily costly routes in (Mes et al., 2013). However, since we already consider the threshold level of each container as evidence of when the container is sufficiently efficient enough to empty in On-Demand waste management. We thus proceed and continue to consider the actual cost of when to collect additional waste.

To consider both additional distances traveled for the vehicle and the estimated volume of the container k, we define a cost ratio for a potential collection as

$$c^{ratio}_{k,R_m,t} = \frac{\min_{i,j \in R_m | R_m \subseteq S} \; c^{insert}_{k,i,j,R_m}}{\hat{v}_{k,t}}, \quad \forall \; k \in V \backslash \{0\}, \; R_m \in S. \tag{4.15}$$

We consider this cost ratio that is low if the container is relatively closely located to a nearby route, or if the container does have much waste. On the opposite, the cost ratio will be high if the container is relatively remotely located to a nearby route, or if the container does not have much waste. However, in this method of defining the cost ratio for a potential collection, we encounter the problem of only choosing the most adjacent containers to the route. The more distant containers will never be chosen, simply because the minimum insertion costs are more likely to be higher by default.

4.5.3 Simulated On-Demand Schedule for Potential Collections

To illustrate this situation, we consider smoothed $c^{ratio}_{k,R_m,t}$ for three additional containers in a simulated On-Demand routing schedule with Gamma distribution filling rates as well as periodic requests based on the distinguishing of waste containers in an On-Demand waste collection schedule. We simulate a small system of 500 containers for 30 days, each following a Gamma distribution with parameters of scale 5 and rate 0.5, which is close to the beforehand parametric estimation of a container's time-to-request in Glostrup's pilot test. That is, we simulate a system where 500 containers are simply placed as random points within a unit square, in which they are located throughout the entire simulation. Each day of the simulation, each container is either labeled NoGo or MustGo with respect to the definitions, where the respective action is carried out using a single vehicle. A container is therefore selected for collection if it holds the probabilities condition. In this setup, we specify A to 5. Further, we model the three additional containers as static MayGos in order to study the effects of locations on cost. By equation (4.15), we fix $\hat{v}_{k,t}$ to 90 for all days in order to avoid $c^{ratio}_{k,R_m,t} \to \infty$ for $\hat{v}_{k,t} \sim 0$ when the container just was emptied the earlier simulation day.

Now, the three MayGo containers are located differently. Container 1 is a container closely located to the existing containers on a collection route, Container 3 is a remotely distant located container to the existing containers on a collection route, and Container 2 is located in between Container 1 and Container 3. We use an approach to determining the minimum insertion costs using the Cheapest Insertion-heuristic, which is also used in (Mes et al., 2013). This heuristic has an advantage in that it only causes the smallest possible increase in the length of the tour, however, it is yet greedy. Therefore, it fits a schedule for waste collection, where the adjacent collection point to a route often is chosen.

The result of this simulation is seen in figure 4.6. We see that the cost-ratio of C_1 is the consequent lowest of the three, hence seen in the numerator of (4.15), its distance of minimum insertion will always be smaller. Thus, if we only would use the cost-ratio as it is, we will always favor C_1 in comparison to both C_2 and C_3. It is therefore important to note that containers at more distant and remote locations would never be selected, even if it would be better to include them now than in a new route another day. This makes sense since containers at a location with

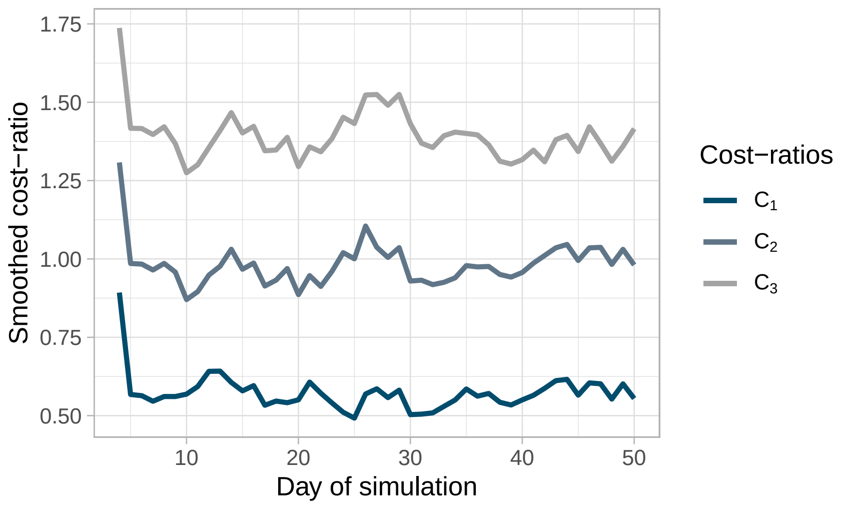

Figure 4.6: *Cost-ratios for three fixed MayGo-defined containers.*

more containers in a close neighborhood require less additional driving than containers at remote locations in a distant neighborhood. This obviously results automatically in smaller ratios.

Further, because of the On-Demand system where containers are selected as MayGo's and MustGo's and the stochastic filling of waste containers within the simulation days, only a small burn-in period is needed. This is seen by having all three cost-ratios sloping downward before stabilizing into a more stationary process. Yet, some containers will be requested even at the beginning of a simulation, meaning that a waste collection will always be relevant. Only a small number of requests have occurred in the simulations beginning and so that only a few collections are to be made, thus highly increasing insertions cost for fixed placed MayGo containers. The beginning of the curve thus illustrates this need for a simulation to begin after a burn-in phase of On-Demand initialization, in which a period of 10 days is reasonable.

4.6 Relative Improvement Criterion

To overcome the situation with containers located at different distances which never will be collected, we use a relative improvement criterion similar to (Mes et al., 2013). Thus, we may compare (4.15) with a smoothed historical average cost. We define the relative improvement criterion as:

$$\Delta^{ratio}_{k,R_m,t} = \frac{c^{ratio}_{k,R_m,t}}{\bar{c}^{ratio}_{k}}. \tag{4.16}$$

We consider (4.16) that is large if $c^{ratio}_{k,R_m,t}$ is high with respect to the historical level, indicating a small amount of waste compared to the additional time spend collecting container k. This means that waste container k is at time t remotely located in consideration of how the routes historically were. In other words, the container k is historically considered to be adjacent to the executed routes. It is, therefore, more cost-efficient to skip the collection of k at time t.

Likewise, (4.16) that is small if $c^{ratio}_{k,R_m,t}$ is low with respect to the historical level, indicating a large amount of waste compared to the additional time spend collecting container k. This means that waste container k is at time t adjacent located in consideration of how the routes historically were. In other words, the container k is historically considered to be remote to the executed routes. Thus, it is an opportunity that we should take. Therefore, it is indeed cost-efficient to insert a MayGo on a route whenever the container's relative improvement criterion is smaller than 1.

However, the smaller the ratio, the better, and thus instead of inserting the MayGo's on the routes as their relative improvement criterion is calculated to be smaller than 1 through the iterative procedure, we turn to a more cost-efficient strategy. That is, as a MayGo may be collected by multiple vehicles, multiple relative improvement criteria for the same container k are to be calculated too. To account for the smaller the ratio the better, we sort all relative improvement criteria by their relative improvement. We are indeed attracted by the highest relative improvements, which are the most cost-effective insertions of the MayGo's on routes. The MayGo's are thus instead inserted on the routes in order by their relative improvement for the On-Demand waste collection.

Lastly, to consider this criterion in practice for the On-Demand waste collection, we diverge from the simple historical smoothed mean and consider the historical smoothed geometric mean instead. The geometric mean is very useful since the costs of container insertion are not independent of each other while the costs tend to make large fluctuations. For instance, if a container was very expensive to collect in just a single day, it would heavily increase the mean for several periods as the average cost is based on history. Such a situation is concealed when applying a geometric mean.

Chapter 5

Proposed Implementation

In this section, we propose an integrated approach for construction of a new routing schedule based on the integration of On-Demand into waste collection applications. The heuristic we present is in itself used for an operational day-to-day planning of both requested and nearby containers to be emptied. We first explain the design and implementation of the heuristic to be used for solving the On-Demand waste collection setup.

5.1 Integrated Approach

The heuristic has two types of input. The first input is of the general characteristics of the system that includes the locations of the households' container, waste disposal rates, threshold levels, the policy of maximum waiting days, probability of a container being requested, available vehicles and their capacities, and penalty costs. The second input is of general characteristics of the waste collection system and aspects relating to the planning methodologies of an On-Demand waste collection schedule, which includes the types of different heuristics being used as input. Thus, the integrated approach of routing and waste container modeling works iterative and interactively in which the routing module creates routes dependent on the parametric modeling of waste containers' fillings. The goal of this heuristic is to create a more flexible heuristic that balances workload. This flexibility is accessible in terms of specific parameter adjustments and thus defines the heuristic's robustness.

5.1.1 Design of the Heuristic

Explicitly, the two introduced modules of routing vehicles and modeling of waste containers can be incorporated into one single heuristic, which is then able to route vehicles to the selective categorized containers each day. That is, the design of this heuristic for efficient solving of the On-Demand waste collecting problem is purely based on the two introduced modules. Further, since our heuristic is solely built upon this theory of these two modules, we even ensure that the time complexity of this heuristic is $\mathcal{O}(n^2)$, making it able to solve problems within a reasonable time.

The structure of the heuristic as an integrated approach is given as a visual overview in figure 5.1. The heuristic considers at any decisive moment the categorization of the containers accordingly to parameters of the first input to finally propose a route of the second input for the day's collection. Each decisive moment is each iteration in a loop that corresponds to each new day where planning has to be made. Yet, this heuristic uses a set of adjustable parameters for initialization, which then regulate the immediate outcome on the routes. These parameters will therefore have a direct impact on the objective function for the long-term horizon. We now further explain each step of the heuristic.

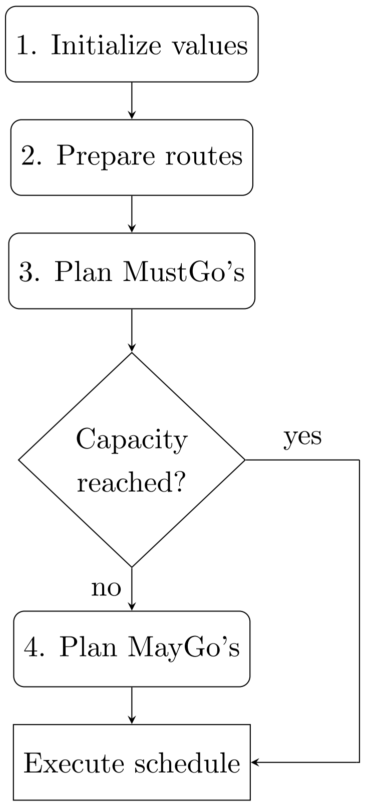

Figure 5.1: *Illustration of the proposed heuristic's iteration.*

Step 1: For each decision moment, we initialize values and estimate new entrant waste for each container based on the parametric estimation of time-to-request. Specifically, the heuristic adds a randomly generated deviate from the Gamma distribution. This means that there is variation between the individual containers' filling to each container. Each container thus follows its own stochastic process. The heuristic indeed considers weekends, in which new entrant waste is also being estimated.

Step 2: The preparation of routes is done by categorizing the containers according to the initialized values from step 1. That is, we use the probability of requesting to calculate whenever a container's request is occurring within a simulation environment, and the threshold-level to determine whenever a container should be considered for collection. The actual request in terms of simulation thus occurs when the stochastic process' filling level tipped the probability to more than a half, which is equivalent to our assumptions of a container's fill level is large enough. Therefore, the initial seed for the vehicle routing is these newly requested emptyings. However, the requests that have occurred within the days beforehand and are yet to be met, will also be a part of the initial seed.

Step 3: All MustGo's from the initial seed from step 2 are planned onto routes using the Savings algorithm as an initial construction heuristic. All MustGo's are assigned to any route. However, not all MustGo's are guaranteed to be fulfilled within the same day due to vehicle and capacity constraints. Yet, hence the allowance to leave out a requested container for another day, a penalty cost will arise on those containers that are left out too many times. In fact, the longer the container is left out, the more penalty cost is produced due to the t-polynomial penalty function. This penalty cost thus ensures that MustGo containers that exceed the policy of maximum days are prioritized, to minimize the amount of MustGo's not being emptied within the policy's time frame.

Step 4: The MayGo's are planned according to the relative improvement criterion, where the higher relative improvement, the better. These MayGo's are inserted on the route based on the Cheapest Insertion cost, which ensures minimal additional cost while utilizing all vehicles' capacities. Then, routes are independently optimized using the Farthest Insertion heuristic before being iterative improved by the Two-Opt iterative heuristic, to detangle potential crossings both within and across routes which improve the solution. Lastly, for the route execution, each container is always fully emptied and additional waste near the container, such as overflows, is also collected.

Now, the intuition behind the integrated approach is therefore to utilize the strength of dynamic VRP methodology while utilizing the weakness of static planning methodology. That is, to

Algorithm 8: Proposed heuristic

```
   Input: A weighted graph $G = (V, E)$
   Output: Route set
1  initializeValues()
2  foreach $day \in totalDays$ do
3      estimateEntrantWaste()
4      if $weekDay$ then
5          MustGos = findMustGos()
6          $S$ = newRoutes(MustGos)
7          if $r(S) > M$ then
8              $S$ = selectBestRoutes()
9          if $Q < Q_k \ \forall k \in M$ then
10             MayGos = findMayGos()
11             foreach $i \notin R_m$ do
12                 if $\min \Delta^{ratio}_{i,R_m,t}$ & $feasibleMerge(i, R_m)$
                    then
13                     $R_m$ = cheapestInsertion($i, R_m$)
14         Optimize($S$)
15     else
16         estimateEntrantWaste()
```

take advantage of customers' complete flexibility of when collection should take place, such that the amount of half-full containers that are being collected is minimized. This integrated approach is designed to handle the dynamic of the On-Demand setup as it models the intervals between emptyings which makes up for the inefficiency caused by individual and variability when forecasting waste for the cycles of a static planning schedule. Moreover, the large-scale routing algorithms ensure competitive vehicle routing for real-life applications where a quality solution within an acceptable time frame is key (Toth and Vigo, 2014). Lastly, we provide the full heuristic for a simulation purpose in a pseudocode in algorithm 8.

Chapter 6

Simulation

We verify our proposal by simulation. Computational results are expressed by analysis of the long-term performance to support both the strengths and weaknesses of this proposed heuristic that may solve the On-Demand waste collection problem with a focus on the financial implications. Therefore, we give insight into the effects of tuning the heuristic's parameters and thereby evaluating by comparing different planning methodologies. Implementation of the proposed heuristic is done in `R`, in which all simulations and results were computed on a system running Windows 10-64 bit, Intel Core i5-4670 CPU @ 3.40Ghz with 16GB DDR3 RAM available.

6.1 Test Instances and Parameter Setting

The purpose of this simulation is to demonstrate that our implementation is the most acceptable cheapest solution and thus closest to optimal. This is achievable by analysis of the heuristic carried throughout discrete event simulation. That is modeling of the operations of a system as a discrete sequence of events in time. The event in the simulation thus advances at a particular instant and thereby marks a change of state. In the time between two events, the state of the system does not change. The only two events that may trigger this time-controller that change the system's state is waste deposited into the container and the emptying of containers.

Specifically, the heuristic's implementation is set with a realistic parameter setting where the default framework is estimated based on Glostrup municipality's pilot project. These specific parameters are referred to as the default parameter setting. Now, we explain these default settings in detail.

6.1.1 Realistic Parameters

In Glostrup, each household accommodates a single container of 240 liters, which is purely designated for household waste. The current renovator, Urbaser A/S, retain two vehicles for household waste collection. A total of 3650 containers of household waste is to be emptied each week for the static schedule in Glostrup. Each vehicle, a diesel-powered garbage truck, may carry up to 600 full containers of household waste, hence all waste is simply compressed inside the vehicle. For simulation purposes, this same capacity ratio for the two vehicles is used as a default. Therefore, the two available vehicles each have a capacity corresponding to 1/6 of the total number of containers that are on a static schedule.

All waste containers each follow a stochastic process, and thus not all containers carry the same filling-level at time t, defined by the Gamma distribution with the same parameters estimated beforehand. Thus, this Gamma$(4.474, 0.452)$ is used to estimate the fill-level each day, hence it was proven to provide more accurate results. Note that since the fillings each follow a stochastic process, the variance between each filling will occur. The individual container's filling may produce very low waste one day, while the subsequent days' filling is high. This gives the fluctuations between the individual containers. Nonetheless, all containers start at the same filling-level of 0 at the beginning of the simulation. When the container is emptied, either because it was requested or simply because it was cheap to empty, it will reset to a fill-level of 0.

The other parameters that are used as a default input to the system are equivalent to the external factors of Glostrup's environment. That is, the A-parameter determining the probability of time-to-request is specified at 5, where it is reasonable to assume that an emptying will be requested when the capacity is almost reached cf. figure 4.5. The threshold is specified at 60%, thus containers may only be categorized as MayGo's if they have an estimated filling-level above 60%. The relative improvement criterion used to insert MayGo's on the routes are assessed by a smoothed geometric average of information of the specific container's 5 previous insertion costs. The policy of maximum days waiting for a container to be emptied after a request occurred is specified at 2 weekdays, meaning that requests occurring Friday afternoon must be emptied no later than Tuesday. Thus, if the same request is fulfilled on Wednesday, a penalty cost on the objective function arises. Within the simulation, this is equivalent to a container becoming a MustGo the following Monday morning.

The heuristic considers which route that minimizes the total distance traversed plus the sum of penalties for all the previously dropped customers, and which containers to empty concerning which containers are the most important to empty. For our default setting, we use a penalty cost of 1000, which is considered very high. Yet, when the penalty cost is this arbitrarily high, then the earliest requested containers will always be chosen and then little to no violations of the maximum waiting days' policy will occur.

Moreover, the default setting for the number of working days is 100, meaning we let the heuristic run for 100 days before evaluating results. However, since the first 10 days are neglected, only 90 specific working days are in use. This was clarified the need for a burn-in phase in figure 4.6. Since the burn-in phase is 10 days, it means that everything before 10 days of simulation is erased. Routes executed within the burn-in phase are thus neglected and not taking into consideration when comparing for multiple simulation days. Lastly, to ensure cost minimization when iterative improving the routes found, the default setup applies the Two-Opt heuristic to every customer and every chain to search for new optima.

6.1.2 Virtual Networks

Specific GPS locations of each of the households' containers in Glostrup that is to be emptied on regular basis are not used. Instead, we simulate the households' containers by representing all containers in a virtual unit square. That is, we randomly place the customers within the unit square. We refer to this virtual unit square as a network, and an example of such network is seen in appendix A.1. All networks' length consists of 10 by 10 minutes driving time, which reasonably reflects Glostrup municipality in terms of travel times. The depot, however, is not placed randomly in the network. The depot is always placed 1/5 length above the top right corner of the network to replicate Vestforbrænding's placement in Glostrup municipality.

We mainly turn to two networks with 100 and 500 customers. These networks are seeded to be the same for every simulation. That is, the exact same network with randomly placed containers and the exact same stochastic process for filling estimation is used for comparing the pairs of instances. The implementation of this proposed heuristic may solve these two networks as a waste collection problem for 100 days of simulation within less than 10 seconds and 600 seconds, respectively.

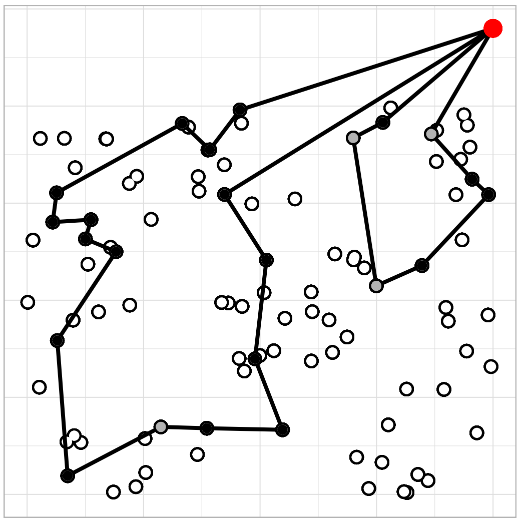

Figure 6.1: *Example of a 100-node replica of Glostrup showing routes for two vehicles in a single day of waste collecting. Each dot is a customer in the network, while the black and gray colored dots are respectively MustGo and MayGo containers visited. The waste disposal center is the red dot in the top right corner.*

Lastly, travel distances between the n customers are formulated as Manhattan distance[1], i.e. the absolute difference between coordinates. The Manhattan distance is often thought of as the distance along the edges of the blocks on a grid. For real-world problems, this measure serves as a better alternative to Euclidean distance[2] which is the direct length between coordinates. These distances are used as ordinal numbers meaning that they serve the purpose of ranking.

[1]That is: $c_{ij} = (x_j - x_i) + (y_j - y_i)$

[2]That is: $c_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$

6.1.3 Proposed Heuristic in Application

For application, we consider the proposed heuristic for On-Demand scheduling in a network that replica waste collecting in Glostrup municipality. This network consists of 100 customers and thus is a small-scale version of Glostrup. Now, the heuristic for On-Demand waste collection will indeed plan routes through the network each day. In figure 6.1 we visualize the network and the ongoing waste collection schedule, i.e. the planned routes, in application for a single day. Note that the route is simply the order in which the customers should be visited.

6.2 The Alternative Static Schedule

We consider our proposed heuristic in the instance of On-Demand waste collection in comparison to the static waste collection on a weekly cycle. This static schedule is equivalent to the current planning methodology in Glostrup.

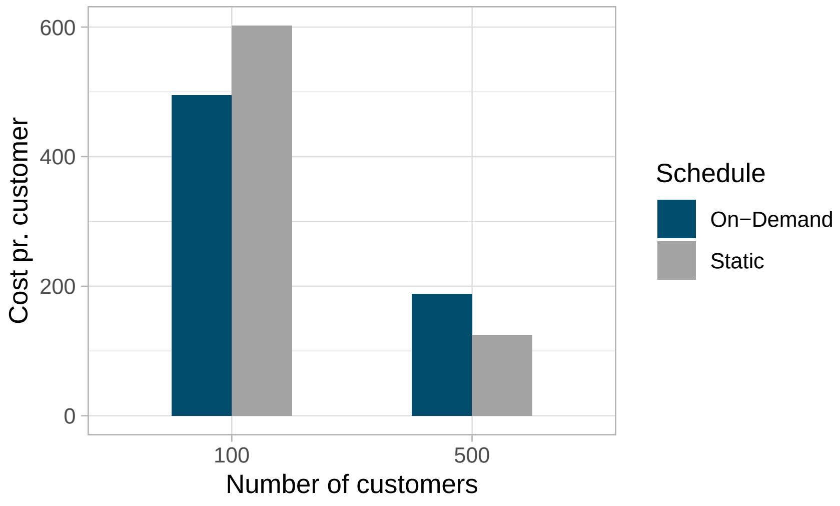

Figure 6.2: *Cost of the proposed heuristic and the static waste collection schedule.*

This current static schedule in Glostrup is based on five neighborhoods, i.e. geometric foundations, and emptying in a single neighborhood takes place each weekday using the two available vehicles, which will split the emptying in the neighborhood equally. Now, the Sweep algorithm is applied hence it being effective for traditional waste collection as in clustering the network into neighborhoods. The afterward applied algorithm for routing each TSP is the Farthest Insertion heuristic. Note that no clusterformation is suggested to apply better than the Sweep algorithm, simply due to these containers' random placement in continuation with the depot's placement within the network, thus no particular density-clustering can be applied efficiently.

The static schedule outperforms our proposal when the network is very dense in terms of cost as seen in figure 6.2. Our proposal, where the waste collection is of On-Demand, makes the vehicles in fact travel distances 21.6% less and 50.8% more compared to a static waste collection schedule in the instances of 100 and 500 customers, respectively. Further, our heuristic averages 90 daily emptyings, opposed to the static schedule's 100 daily emptyings in the instances of 500 customers. Thus, despite fewer emptyings, our heuristic is more costly. This is explained by the characteristic called *star-driving*. That is, as the requested containers should be emptied, the collectors may have to traverse a long route each day to fulfill these requests throughout the entire network. Sometimes, the collectors might even traverse near the same residential multiple times throughout a week, which is inefficient.

Further, the scale at which the waste collection operates matters, as there are large economies of scale in retrieval for both schedules. It is reasonable to argue that the cost reduces as the number of customers within the network increases. This is simply due to a larger number of customers means more densely located customers. The collection area is the same between the two networks of 100 and 500 customers, so the difference is in how many customers that are placed. The collectors do not have to traverse far between each emptying when customers are densely located.

It is also notable that the Sweep algorithm's potential is not fully exploited since the containers are located randomly within the network, and not in clusters while having the depot's placement in the top right corner of the network. This situation makes every route rather radial, which is the pure disadvantage of Sweep. Yet, this disadvantage is not big enough to offset the disadvantage of On-Demand waste collection's star-driving.

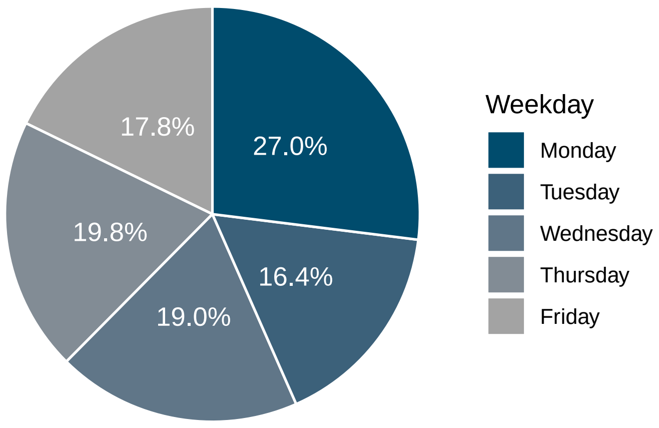

Figure 6.3: *Distribution of the proposed heuristic's emptyings made on weekdays in the network of 500 customers.*

Figure 6.3 shows that our proposed heuristic in the On-Demand setup makes 27.0% of all emptyings on a Monday in the instance of 500 customers. This is due to our proposal's allowance of emptying non-requested containers. It is, nevertheless, contrary to NSR's experiment in Helsingborg where more than 40% of all emptyings occur on a Monday. The reason for the need of emptying early on the week is simply due to emptyings not being able to take place on weekends where waste is also pilling. On the other hand, the static schedule always has all emptyings being equally distributed among the workdays.

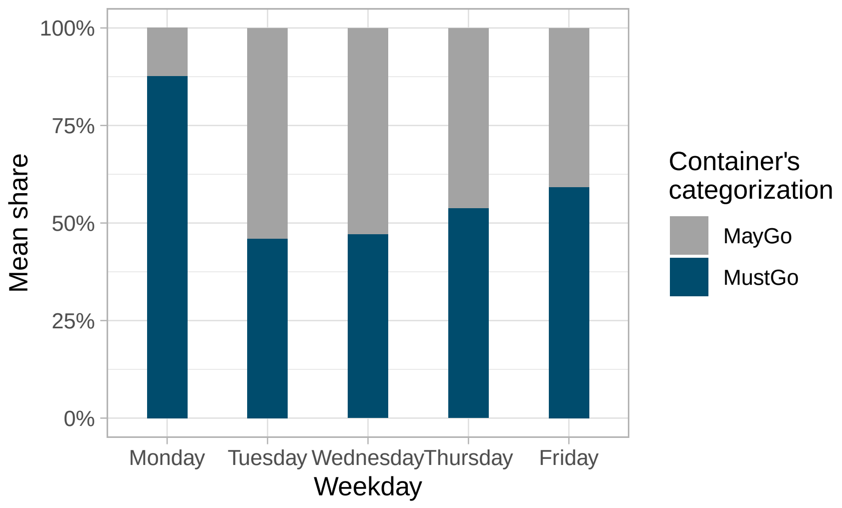

Figure 6.4: *Distribution of emptied containers' categorization on weekdays by the proposed heuristic in the network of 500 customers.*

Since many requests for emptyings have been made throughout the weekend, our heuristic averagely collects a large proportion of MustGo's on Mondays, as seen in figure 6.4. This inevitably ensures that there is a lack of capacity to then empty nearly located MayGo's on Mondays. However, throughout the week, the heuristic now seeks to utilize capacity and the MayGo's are then a more frequent appearance when emptying. The cost of utilizing capacity is thus present throughout the week as well.

Our heuristic is thus able to spread emptyings across the week similar to the static schedule. This undeniably reduces pressure on the collectors as it balances the workload. Yet, the On-Demand waste collection's increased cost is due to star-driving. Our proposal is therefore an competitive alternative.

6.3 The Alternative 3-day On-Demand Schedule

We consider our proposed heuristic in the instance of On-Demand waste collection in comparison to the convenient alternative that operates on a 3-day schedule. This is the alternative 3-day On-Demand schedule. This type of dynamic planning has been taking place during the pilot test in Glostrup. The reasoning for using such an alternative is the longer time-to-request, an equivalent less driving should be needed. For Glostrup, this corresponds to only collect waste on 3 out of 5 days throughout the week.

The 3-day On-Demand schedule takes advantage of the policy of 2 maximum days waiting before having the request of emptying fulfilled thus it only operates Mondays, Wednesdays, and Fridays. This way, the requests that occur Monday and Tuesday will be fulfilled Wednesday, and the requests that occur Wednesday and Thursday will be fulfilled Friday, and requests that occur Friday and over the weekend will be fulfilled Monday.

Such a 3-day schedule indeed puts a lot of pressure on collection on certain days. However, the schedule releases 2 full days during the weekdays which may be useful. Further note that this 3-day schedule does not consider emptying any MayGo's. This means that the cost of complete capacity utilization is not prominent. Yet, star-driving is present as both schedules consider On-Demand waste collection.

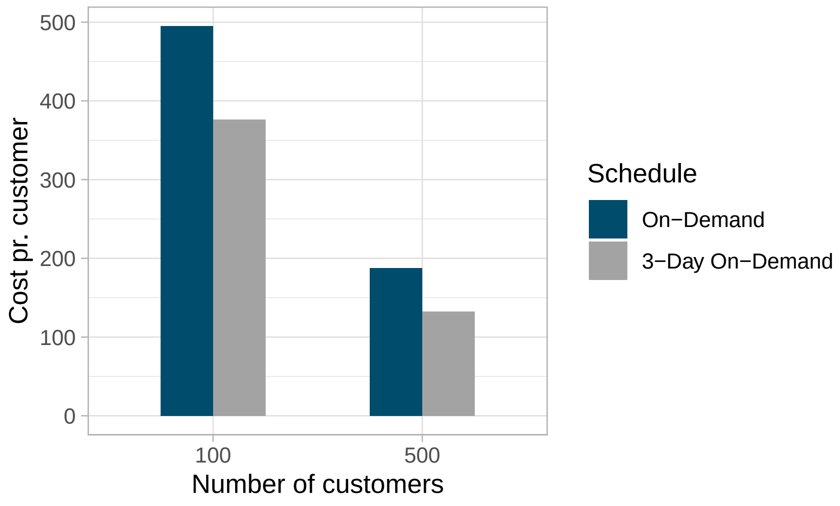

Figure 6.5: *Cost of the proposed heuristic and the 3-day schedule.*

The 3-day On-Demand schedule's cost is low compared to our proposed heuristic as seen in figure 6.5. This means that the 3-day schedule is thus a competitive alternative to our proposed heuristic, as the routes planned are respectively 24.0% and 29.4% less costly than our proposed heuristic for On-Demand scheduling in the instances of 100 and 500 customers. Both the 3-day On-Demand schedule and our proposed heuristic encounter no containers being left out for emptying more than a single workday. However, the pressure on the 3 days is present. Our proposal daily empties 90 containers on average, while the 3-day schedule daily empties 115 containers on average. This is an average workload increase of 27.8%.

The cost of complete capacity utilization and star-driving are the reasons why our proposed heuristic is producing more costly routes when considering the 3-day alternative, despite fewer emptyings. This is simply because our proposal does collect waste every weekday. However, our heuristic does not produce a high workload on certain days. Therefore, our heuristic is still a competitive alternative.

6.4 Inclusion of Neighborhoods

We consider our proposed heuristic in clustered networks. That is, we introduce clustered networks where the customers are placed in clusters as an alternative to customers being placed randomly. This is to simulate the sense of a municipality's neighborhoods.

In Glostrup, 5 neighborhoods are defined since the static planning schedule ensures waste collecting in a single neighborhood each day on a weekly cycle. Thus a network of 5 clusters, i.e. neighborhoods, would be appropriate to mirror Glostrup. Note that these new clustered networks each have the number of customers being evenly distributed to each neighborhood. That is, a network of 100 customers with 5 clusters will each cluster contain 20 customers. In appendix A.2, we visualize such clustered network.

Now, to also mirror the methodology of a static planning schedule's collection over a weekly cycle, we introduce networks consisting of 2, 5, and 10 clusters, respectively. We use a simple k-means algorithm to divide and gather the clustered network in 5 groups such that a vehicle will be collecting in one of the 5 groups each day.

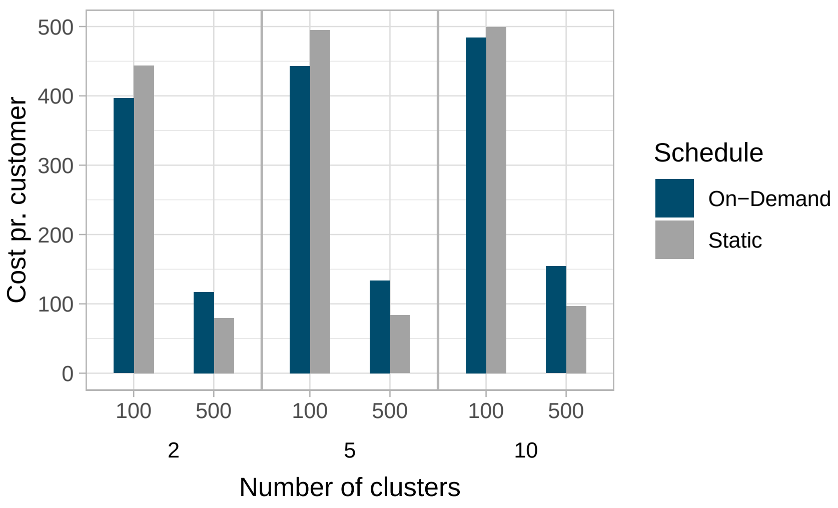

Figure 6.6: *Cost of the proposed heuristic in clustered networks by number of clusters and number of customers.*

However, no municipality is the same and the routing cost is highly dependent on the specific layout of the simulated network. This is because the cost of traveling between the different neighborhoods may be low for some networks and large for others. The customers' location in networks does play an important role in terms of efficient routing, yet it is an external factor that is to a large extent insensitive. For figure 6.6, we have generated 10 types of different clustered network instances with 100 and 500 customers each and compare our results by the arithmetic mean to gain robust estimates of the cost.

As a result, the larger number of clusters makes the cost higher as well for both schedules. Our proposal makes the vehicles travel distances 10.5% less and 60.0% more in the instances with 100 and 500 customers for 5 clusters, respectively. Thus, star-driving and complete capacity utilization is present in clustered networks too. Our proposed heuristic is still yet robust and competitive even in clustered networks.

6.5 Sensitivity of Vehicles' Capacities

To challenge the capacity of vehicles for our proposed heuristic, we introduce an alternative to the diesel-powered garbage trucks, namely an electric-powered garbage truck which may carry

20% less. An electric-powered vehicle is attractive to the municipality due to lower emissions, yet unhandy for the renovator since less capacity. However, no electric-powered vehicle is in use for the municipality of Glostrup today, but it does serve its relevance to study to comply with the future's regulations and demands.

A diesel-powered vehicle that may otherwise carry 600 full containers of household waste, may only carry 480 full containers of household waste if it is electric-powered instead. Therefore, the two electric-powered available vehicles for waste collecting each has a capacity corresponding to 1/7.6 of the total number of containers available in Glostrup.

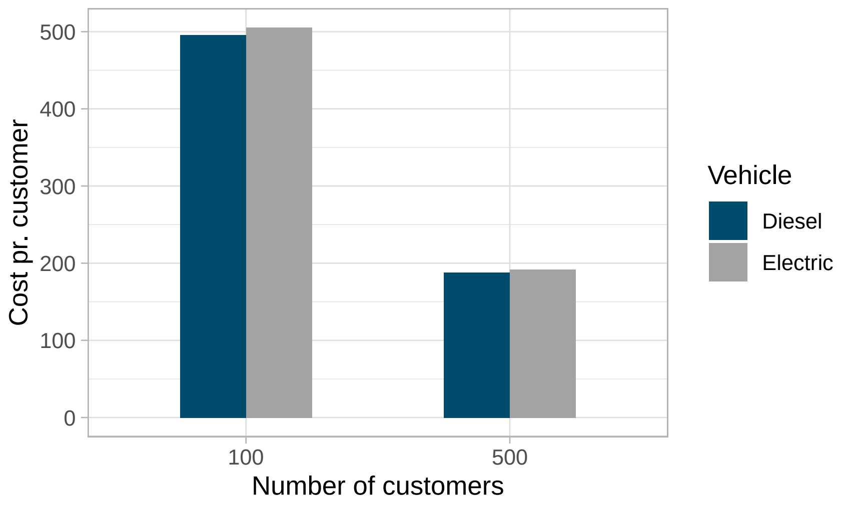

Figure 6.7: *Cost of operating with diesel-powered and electric-powered vehicles.*

The cost of deploying electric-powered vehicles is 2.0% and 1.8% more expensive as seen in figure 6.7 for the networks of 100 and 500 customers, respectively. The small increase in cost is because of the complete capacity utilization that is now diminished as the vehicles' capacities are smaller.

Additionally, we see the number of deployments of either 0, 1, or 2 vehicles differ when deploying a new type of vehicle as seen in figure 6.8. Specifically, the number of days when 2 vehicles are deployed to fulfill all requests increases when the electric-powered vehicles are possessed. Yet, within the 100 customer network, we do see a single occurrence of no routes being deployed. This is simply explained by having a small amount of customers present increases the likelihood of having a large proportion of customers on similar stochastic filling rates, in which case no MustGo's are present.

Lastly, for the instance of 100 customers, 0 and 24 containers waited 1 day for having the request fulfilled for both diesel-powered and electric-powered vehicles, respectively. For the instance of 500 customers, 30 and 60 containers waited 1 day for having the request fulfilled for both diesel-powered and electric-powered vehicles, respectively. No containers were waiting more than 1 day in both cases. Therefore, our proposed heuristic experience no issues of re-

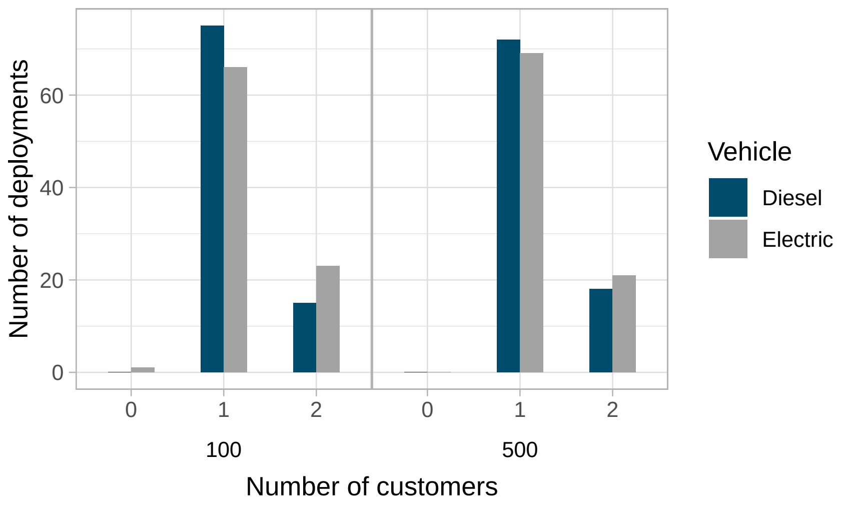

Figure 6.8: *Deployment occurrences for schedules with 2 diesel-powered and 2 electric-powered vehicles available, respectively.*

quest fulfillment when downsizing vehicles' capacities by switching to electric-powered vehicles for waste collection purposes, hence its high flexibility in capacity utilization.

6.6 Number of Available Vehicles' Dependence

We may specify our heuristic to only produce routes for a single vehicle. Due to the vehicle's large capacity, a third vehicle is never an attractive option. That is, given the setup of this waste collection in Glostrup, a third vehicle is not even introduced on any given day throughout the entire simulation.

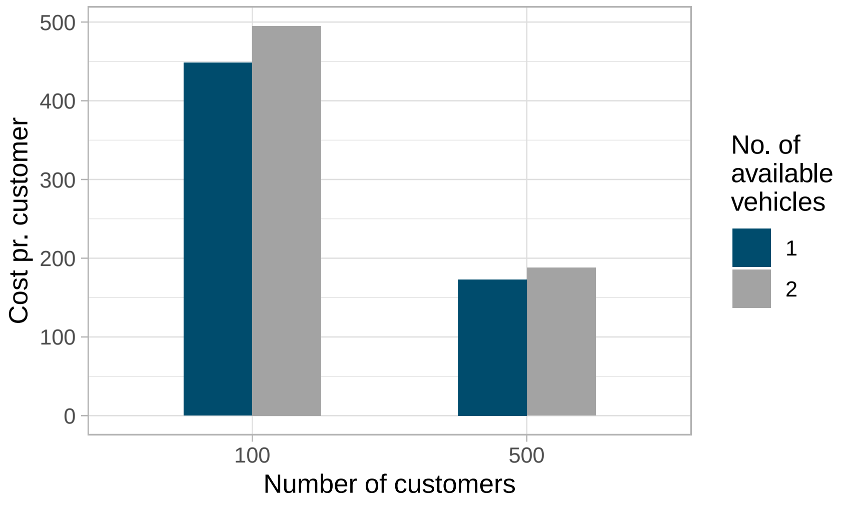

Figure 6.9: *Cost in instances when a different number of vehicles is available*

In figure 6.9, we show that specifying the number of vehicles to a single vehicle in fact reduces the cost by 10.4% and 8.4% in the networks of 100 and 500 customers, respectively. This is due to the high initial cost of deploying a second vehicle hence the depot's placement, which therefore makes short routes very costly pr. customer.

However, not all containers can be collected on the particular day of requesting when only deploying a single vehicle. Therefore, the number of containers that wait for request fulfillment increases. When deploying a single vehicle in the network of 500 containers, 191 containers waited 1 day, 28 containers waited 2 days, and 1 container waited more than 2 days before getting emptied. Contrary, only 30 containers waited 1 day and no containers waited more than 1 day before getting emptied when 2 vehicles were available for deployment. Therefore, the cost of capacity utilization is also reduced, as the single vehicle is busy emptying MustGo's instead of inserting the additional MayGo's on the route.

Our proposed heuristic may handle waste collection when considering a specified number of vehicles at trade-off at having an increased number of customers wait before request fulfillment. As the containers are allowed to be skipped for emptying, this could be an ideal trade-off as seen from the renovator's point of view if initiating more than a single vehicle may altogether be avoided. Note that this cost is only the cost of traversing during the simulation and not the cost of actually holding a vehicle as an asset. Thus, whether or not the allowance of two vehicles to collect waste within an On-Demand setup is a sunk-cost or not is debatable. This cost of traversing does not factor in the reliability of having two vehicles for collection.

6.7 Insertion Heuristics' Significance

Our proposed heuristic is capable of handling different single-route optimization methods. The Cheapest Insertion heuristic is indeed used for the insertion of MayGo containers. However, since many of the days build routes that only require a single vehicle, a pure TSP route optimization method is preferable. In the cases of days where more than a single route is build, the default setup still follows.

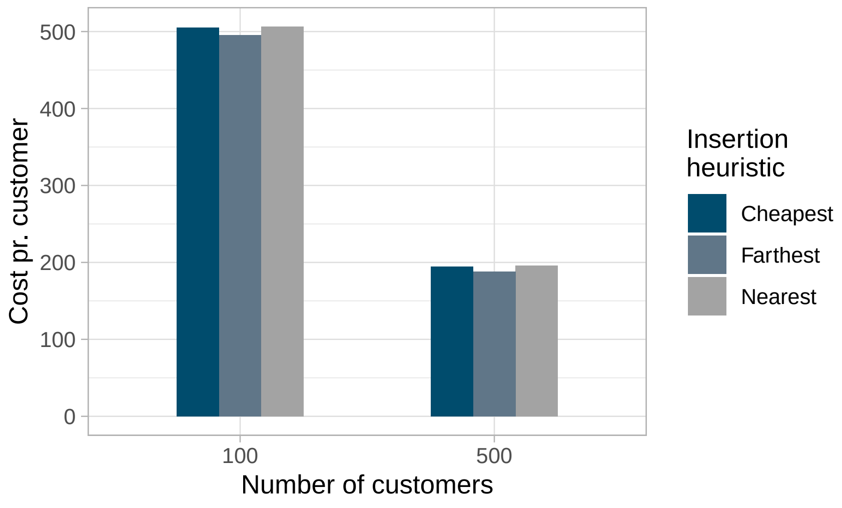

Figure 6.10: *Cost by three different insertion heuristics.*

The Farthest insertion that links nodes that are far away into the route and thereby form a peripheral outline of the route early is the better alternative as seen in figure 6.10. The Farthest Insertion includes the specific characterization of building peripheral routes early which is ideal for an On-Demand waste collection since the vehicles now traverse the whole network each day, contrary to a static schedule in which the vehicle only traverses in a single neighborhood.

Both the Cheapest and Nearest Insertion does not necessarily build peripheral routes, thus costs are then higher due to On-Demand waste collection's need for star-driving. Yet, the three insertion heuristics in practice makes only a very small difference in cost for our proposed heuristic. Our proposed heuristic is thus robust in terms of how each route is specifically planned since the cost between single-route optimization methods is similar.

6.8 Different Filling Rates in the Population

We may specify different filling rates within the network to account for individual filling rates of waste containers. The waste filling rate is an external factor that is to a large extent insensitive. However, different filling rates are present in other waste fractions, e.g., glass waste, where the filling rates' are lower. Now, we specify some of the network's customers to a new Gamma distribution arbitrarily selected as $\text{Gamma}(2.237, 0.452)$. This means some of the population moves from an average daily filling-level of 9.90 to 4.95. In figure 6.11, we refer to the low filling rate as this newly selected Gamma distribution.

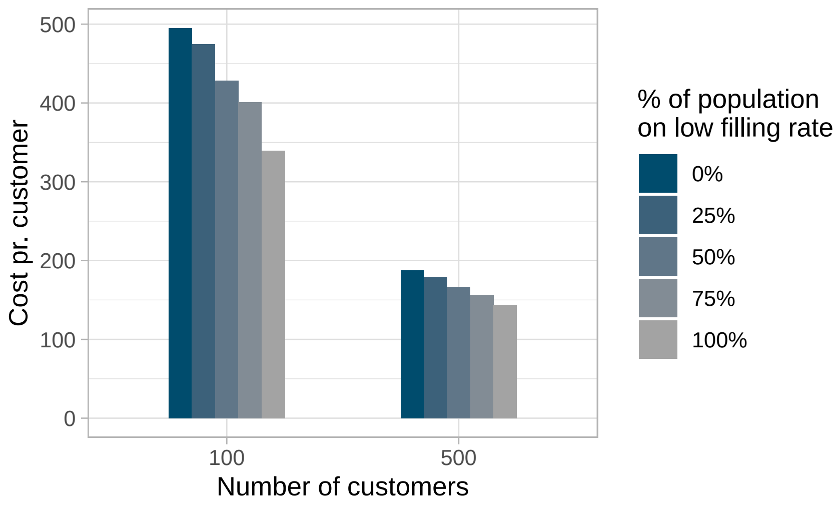

Figure 6.11: *Cost by different filling rates.*

As more of the population fills waste with a lower rate, the strictly less expensive the overall cost is, as seen in figure 6.11. This is no surprise since the less frequent the collector must visit a customer, the less costly it becomes due to the avoidance of star-driving. The complete capacity utilization cost is now more prominent. Thus, other waste fractions than household waste may be attractive for On-Demand collection, in which the proposed heuristic is efficient since the cost of emptying is reduced as more customers are on the lower filling rate.

Therefore, our proposed heuristic is robust to different filling rates within the population, as it handles waste collection when different filling rates are present, which is the ground and premise for initiating On-Demand waste collection.

6.9 The Penalty Scaling Factor

The travel distances and the penalty cost are highly correlated since the proposed heuristic must consider which MustGo's are to be neglected emptying if capacity is otherwise reached. The heuristic explicitly balances the cost of collecting on a better route and the penalty cost of not considering the earliest requested containers.

Thus, we may assign a sufficiently large penalty cost in our heuristic, to ensure minimal neglecting of constraints and policy violations. In this case, after dropping a customer to make a feasible solution for the day, the heuristic does not drop the same customer across multiple days as the penalty for doing so may exceed any further reduction in travel distance. If doing so, it will increase the objective function dramatically since the penalty function is polynomial. On the other hand, one may also assign smaller penalties, in which case the heuristic will drop the same customers multiple times since they may be located remotely and thereby will make the route's traveling cost large by default.

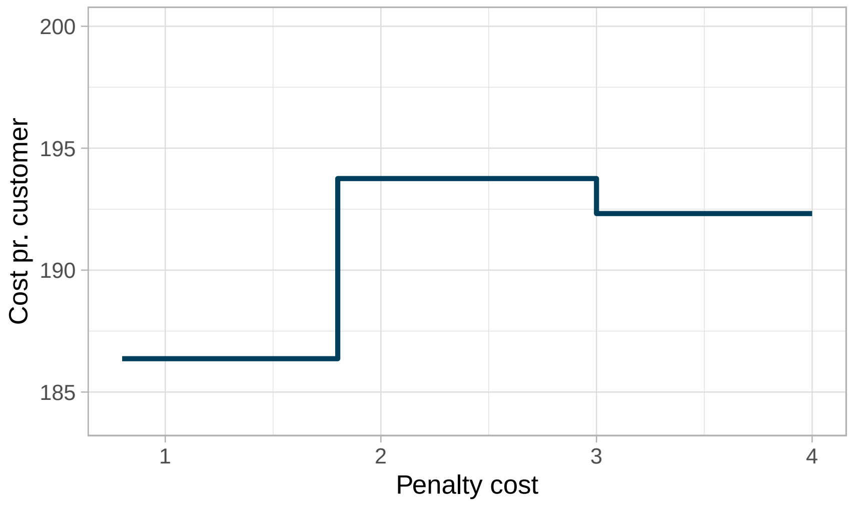

Figure 6.12: *Cost by varying penalty cost in a network of 100 customers. The 2 available vehicles for collection have been limited to* 1/14 *capacity to increase the risks of neglecting planning of all MustGo's.*

The cost increases whenever the penalty cost becomes too leading as seen in figure 6.12. When penalties are too leading, the heuristic does dispatch costly routes which lead to no customers experience waiting. Likewise, when penalties are inconsequential, the heuristic does not dispatch costly routes, leading to customers experiencing waiting.

Throughout a long period of simulation, short-term decisions such as selecting a certain container for collection on a specific day may thus affect other routes and thus the long-term cost.

This is because the heuristic only plans routes for the current day, simply because the premise of On-Demand waste management is the stochasticity of which customers being present. This too explains why the cost may drop right after an increase since the penalty was balanced in consideration of the routing cost, as seen in figure 6.12.

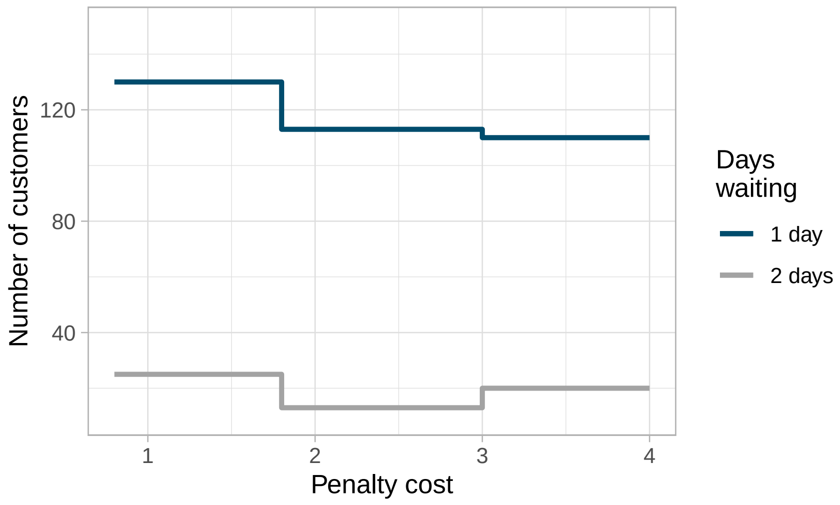

Figure 6.13: *Number of customers waiting by varying penalty cost in the network of 500 customers. The 2 available vehicles for collection have been limited to $n/14$ capacity to increase the risks of neglecting planning of all MustGo's.*

Moreover, as the penalty cost becomes too leading, the number of customers waiting 1 and 2 days decreases as seen in figure 6.13. Note that the total numbers of customers do not necessarily drop to 0 when the penalty cost is very leading. This is because the vehicles may not be able to fulfill all requests despite how large the penalty cost becomes, which can not be avoided. Although, our proposed heuristic considers the trade-off between cost and the number of customers that must wait before their request is met when applying a penalty scaling factor.

6.10 The Threshold Level's Significance

We consider the threshold level that determines when a container is said to be convenient to collect and thereby labeled as a MayGo. That is, the higher the threshold level, the closer to overflowing the container is before it is considered for collection without it having been requested. At a threshold level of 100%, the heuristic does not consider any containers for potential collection. Such setup is equivalent to an ordinary dynamic schedule.

In figure 6.14, we see how the increasing threshold level shows an approximately linear decrease in the cost of the proposed heuristic. The lower cost at high threshold levels is the consequence of reducing our heuristic's ability to utilize capacity. The higher cost at low threshold levels is thus the opposite consequence of increasing our heuristic's ability to utilize capacity. Note that a higher threshold does not necessarily increase star-driving as requests will always be present throughout the whole network. Yet, the cost of capacity utilization is reduced enough to offset the cost of star-driving at higher threshold levels such that the traveling cost is reduced.

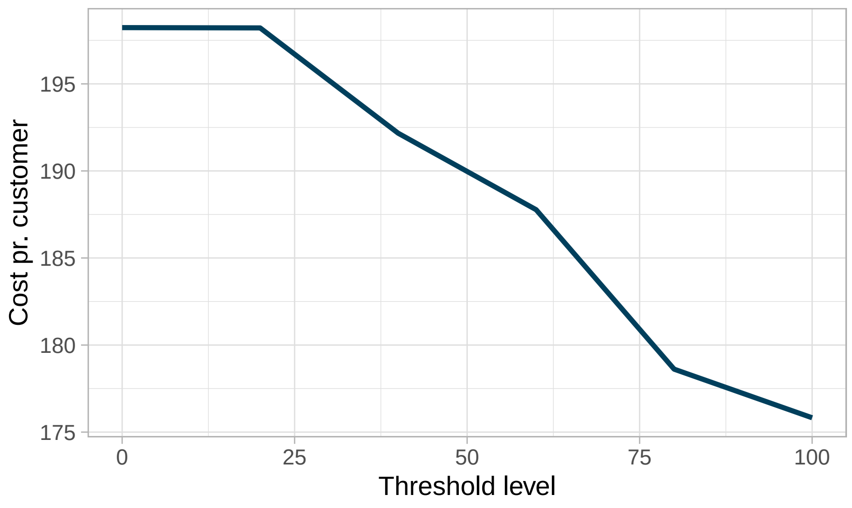

Figure 6.14: *Cost by varying threshold levels in the network of 500 customers.*

Now, the proposed heuristic considers the trade-off between an equal distribution of collection and low cost. For this, consider table 6.1. Here we see the distribution of the emptyings taking place throughout the week for varying thresholds. Note that the lower the threshold, the more equal distribution of emptyings.

	Distribution of emptyings				
Threshold level	Monday	Tuesday	Wednesday	Thursday	Friday
0	19.0%	20.3%	19.2%	20.9%	20.6%
20	17.0%	22.7%	18.8%	19.4%	22.1%
40	21.4%	23.3%	20.1%	18.2%	17.1%
60	27.0%	16.4%	19.0%	19.8%	17.8%
80	36.2%	12.2%	16.0%	18.4%	17.2%
100	36.3%	12.5%	15.1%	18.5%	17.6%

Table 6.1: *Distribution of emptyings for varying threshold levels.*

Nevertheless, at a high threshold level of 60%, 80%, and 100%, the number of containers that waited more than 1 day when requested was 30, 55, and 92 during the simulation, respectively. At the lower threshold levels, no containers waited any days at all. Yet for all instances of varying threshold levels, no containers waited more than 1 day. Thus, there is a much greater risk of eluding the policy of maximum days too by increasing the threshold level.

The trade-off between intense collection on certain days and low cost, and distributed emptyings throughout the week and high cost is present in On-Demand waste management. However, our proposed heuristic is capable of handling this by controlling the threshold level. Therefore, our heuristic is robust in different degrees of dynamics in On-Demand scheduling as to which and how many containers to consider for collection.

6.11 Final Comments on Verification of the Proposal

Our proposed heuristic is shown to be a robust solution method for On-Demand waste management applications as it considers dynamic routing in a network with estimated waste volumes with the addition of an efficient collection of nearby non-requested containers. However, the heuristic plan routes that are more costly despite fewer emptyings. This can be counter-intuitive because it seems so obvious that fewer emptyings should result in less driving.

The higher cost of On-Demand waste collection is a consequence explained by star-driving. That is, having the vehicles traverse in the whole network each day, contrary to a static schedule in which the vehicles are limited to traverse in a neighborhood. The preconditions for On-Demand emptying may claim waste collection in the same neighborhood up to 23 days within the same month, while the static schedule in the same neighborhood always claims waste collection 4 days within the same month. It is thus the fact that On-Demand waste collection spreads the collection over many days without reducing the collection area which then causes the driving distance to increase. This is also where we find the explanation for why the planning methodologies that spread a lot, such as both the 3-day schedule and our proposed heuristic are more expensive than those that spread a little, such as the static schedule.

We also have the cost increased by the introduction of complete capacity utilization by inserting non-requested containers on the routes. This feature may be unwanted, yet one will have to assess the trade-off between increased cost or increased risk of neglecting the collection of actually requested containers and an unbalanced workload throughout the week. Further, the density of the network highly affects the economies of scale when waste collecting.

In addition, the variation in the individuals' waste filling rate also affects On-Demand waste collection. This is because the estimated distribution largely underestimates some containers with very long intervals between emptyings, i.e. the fat right tails of the parametric estimate. By underestimating the long time-to-requests, we can not ensure efficient individual container collection, since the containers may not be full on the day of collection. Yet, suggestions such as IoT-sensor-equipment may be able to solve such a problem, since one may then accurately view the containers' filling-levels in real-time. A planning methodology based on sensors is indeed similar to our proposed heuristic which incorporated the base stock model's threshold to determine when it is convenient to collect a particular container based on parametric filling. However, hence the large variation of waste filling within household containers that the individual citizen delivers from day-to-day, sensors in private household containers are often thought of as inconsiderable (Freiesleben, 2021). On the other hand, using large shared containers instead, the amount of waste causes the variations to balance each other. Parametric estimates even without sensors will be more accurate when using large shared containers.

Chapter 7

Conclusion

In this thesis, we have proposed a heuristic as a planning methodology for the waste collection problem, in which emptying-as-needed is introduced. The methodology presented applies to the general class of Inventory Routing Problems with many customers, and the findings of this thesis are thus useful for waste collection in municipalities.

The proposed heuristic combined classic VRP literature that can handle large-scale instances for waste collection purposes such as the Savings heuristic, insertion heuristics, and a Two-Opt heuristic, in which case time complexity of quadratic time is ensured. We extended the heuristic by incorporating survival analysis to parameterize the waste filling rate of containers. With the introduction of an inventory modeling approach into waste management, we provided a model in a time-discretized version of reverse base stock modeling to cope with the stochastic filling in a waste collection problem, together with a distinguishing of waste containers in a routing setting. We thereby achieved a formal definition of the dynamic selection of both containers considered and containers convenient for emptying in On-Demand waste management applications. This proposed heuristic extended the dynamic waste collection schedule to now also select non-requested containers for waste collection, to ensure complete capacity utilization.

In addition, we provided insight into the applicability of our proposed heuristic as our proposal was implemented and launched in a case study of Glostrup municipality where On-Demand waste collection has been an initiated project. Verification in simulation shows that this heuristic in On-Demand waste management applications provides competitive results in terms of cost and a balanced workload throughout the week, compared to ordinary dynamic waste collection without the extension.

We gave results that demonstrated that the proposed heuristic suggested in this thesis did produce more costly waste collection compared to both a static and a dynamic schedule when considering household waste. Despite fewer emptyings when On-Demand waste collection is deployed, it is not sufficient to make up for the significantly more driving, i.e. star-driving, that is needed when this On-Demand waste collection is present. However, the proposed heuristic may be used to avoid pressure on certain days for waste collection as it may generate routes that make a more equally distributed amount of emptyings throughout the week. This is because the proposal may allow the emptying of non-requested containers based on the introduction of a distinguishing between containers. As the proposed heuristic is parameterized, the specific parameters may be tuned to consider this trade-off between cost and an equally distributed workload. This includes how aggressive the emptying of non-requested containers is. If it is assumed that all containers are convenient to empty, then traveling costs are highly increased due to increased capacity utilization. On the other hand, if all containers are to only be emptied when they are requested then traveling costs will be reduced due to the ignoring of complete capacity utilization.

The weakness of all waste collection planning is the scale and the density of customers in the collection area. This highly interferes with results for all waste collection schedules. However, this external factor is highly insensitive but highly visible for On-Demand waste collection. Further, the introduction of On-Demand waste collection reinforces the second weakness in terms of the lack of accurate modeling of waste filling rates. This thesis thus concluded that given the right technological accessories such as sensors reading the waste filling-level inside a container, the proposed heuristic may turn this shortcoming into a strength.

Bibliography

K. Altinel and T. Öncan. A New Enhancement of the Clarke and Wright Savings Heuristic for the Capacitated Vehicle Routing Problem. *The Journal of the Operational Research Society, vol. 56 (8)*, pages 954–961, 2005.

C. Archetti, L. Bertazzi, G. Laporte, and M. G. Speranza. A Branch-and-Cut Algorithm for a Vendor Managed Inventory Routing Problem. *Transportation Science, vol. 41*, pages 382–391, 2007.

S. Arora and B. Barak. *Computational Complexity.* Cambridge University Press, 2009.

J. F. Bard, L. Huang, P. Jaillet, and M. Dror. A Decomposition Approach to the Inventory Routing Problem with Satellite Facilities. *Transportation Science, vol. 32 (2)*, pages 189–203, 1998.

C. Caplice. Transportation Management: Vehicle Routing, 2006.

N. Christofides and S. Eilon. An algorithm for the vehicle dispatching problem. *Journal of the Operational Research Society, vol. 20*, page 309–318, 1969.

G. Clark and J. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research, vol. 12*, pages 568–581, 1964.

S. E. Comert, H. R. Yazgan, S. Kir, and F. Yener. A cluster first-route second approach for a capacitated vehicle routing problem: a case study. *International Journal of Procurement Management, vol. 11 (4)*, pages 399–419, 2018.

Danmarks Statistik. Boliger efter område, beboertype, anvendelse, udlejningsforhold, ejerforhold og opførelsesår, 2021. URL `www.statistikbanken.dk/BOL1`.

G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science, vol. 6*, pages 80–91, 1959.

P. Erlitz. Det virker: Smart tømning af affaldsbeholdere. *Vestegnen Sydkysten*, 2021.

M. L. Fisher. Optimal Solution of Vehicle Routing Problems Using Minimum K-trees. 1994.

M. L. Fisher and R. Jaikumar. Generalized Assignment Heuristic for Vehicle Routing. *Networks, vol. 11*, pages 109–124, 1981.

S. Freiesleben. Professor om fleksibel tømning: ”Ruteplanlægning er ikke en tryllestav”. *ING/WasteTech*, 2021.

T. J. Gaskell. Bases for Vehicle Fleet Scheduling. *Operational Research Society, vol. 18 (5)*, pages 281–295, 1967.

M. Gendreau, A. Hertz, and G. Laporte. A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science, vol. 40*, pages 1276–1290, 1994.

B. E. Gillett and L. R. Miller. A Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research, vol. 22 (2)*, pages 340–349, 1974.

Glostrup Kommune. Regulativ for Husholdningsaffald, 2014.

F. E. Harrell. *Regression Modeling Strategies.* 2001.

R. A. Holmes and R. G. Parker. A Vehicle Scheduling Procedure Based Upon Savings and a Solution Perturbation Scheme. *Operational Research Quarterly, vol. 27 (1)*, pages 83–92, 1976.

M. Johansson. Utvärdering av innovationstjänsten OnDemand (Behovsanpassadtömning) och uppkopplade avfallskärl - Intervjuanalys av testpersoner och personal i samband med bestäm-själv-tömning. 2019.

G. Laporte, M. Gendreau, J.-Y. Potvin, and F. Semet. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research, vol. 7 (4-5)*, pages 285–300, 2000.

E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys. *The Traveling Salesman Problem.* Wiley, 1985.

G. F. Lawler. *Introduction to Stochastic Processes.* Chapman and Hall/CRC, second edition, 2006.

J. Lysgaard. Clarke & Wright's Savings Algorithm. 1997.

M. Mes. Using Simulation to Assess the Opportunities of Dynamic Waste Collection. *Use Cases of Discrete Event Simulation*, 2012.

M. Mes, M. Schutten, and A. P. Rivera. Inventory routing for dynamic waste collection. *Waste Management, vol. 34 (9)*, 2013.

J. M.Gutierreza, M. Jensen, M. Henius, and T. Riaz. Smart Waste Collection System Based on Location Intelligence. *Procedia Computer Science, vol. 61*, 2015.

R. H. Mole and S. R. Jameson. Bases for Vehicle Fleet Scheduling. *Operational Research Quarterly, vol. 27 (2)*, pages 503–511, 1976.

NSR. On Demand - Resultat och Statistik. 2019.

A. Omara, D. Gulen, B. Kantarci, and S. F. Oktug. Trajectory-Assisted Municipal Agent Mobility: A Sensor-Driven Smart Waste Management System. *Journal of Sensor and Actuator Networks, vol. 7*, 2018.

I. Or. *Traveling Salesman-Type Combinatorial Problems and Their Relation to the Logistics of Regional Blood Banking.* Northwestern University, 1976.

I. H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research, vol. 41*, page 421–451, 1993.

H. Paessens. The savings algorithm for the vehicle routing problem. *European Journal of Operational Research, vol. 34 (3)*, pages 336–344, 1988.

G. K. Rand. The life and times of the Savings Method for Vehicle Routing Problems. *ORiON, vol. 25 (2)*, page 125–145, 2009.

D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis. An Analysis of Several Heuristics for the Traveling Salesman Problem. *Society for Industrial and Applied Mathematics Journal on Computing, vol. 6*, 1977.

D. M. Ryan, C. Hjorring, and F. Glover. Extensions of the Petal Method for Vehicle Routing. *Journal of the Operational Research Society, vol. 44*, pages 289–296, 1993.

L. V. Snyder and Z.-J. M. Shen. *Fundamentals of Supply Chain Theory.* Wiley, second edition, 2019.

P. Toth and D. Vigo. *Vehicle Routing Problems, Methods, and Applications.* Society for Industrial and Applied Mathematics, second edition, 2014.

Appendix A

Generated Networks

A.1 Random Generated Network

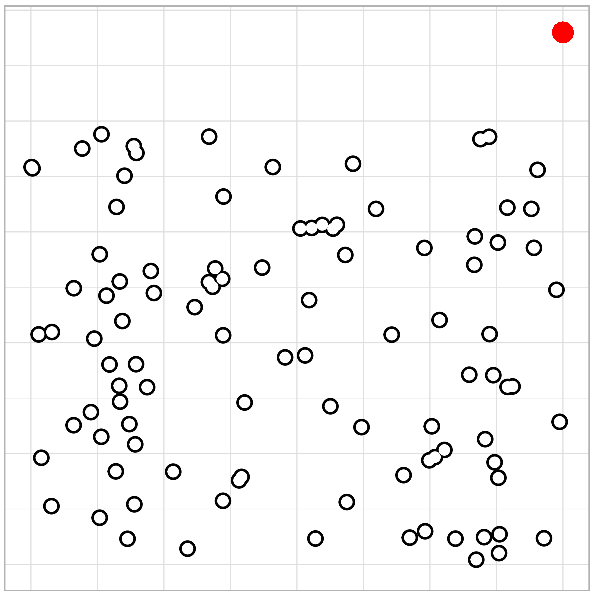

Figure A.1: *Randomly generated network of 100 customers. The depot is the red dot in the top right corner.*

A.2 Random Generated Clustered Network

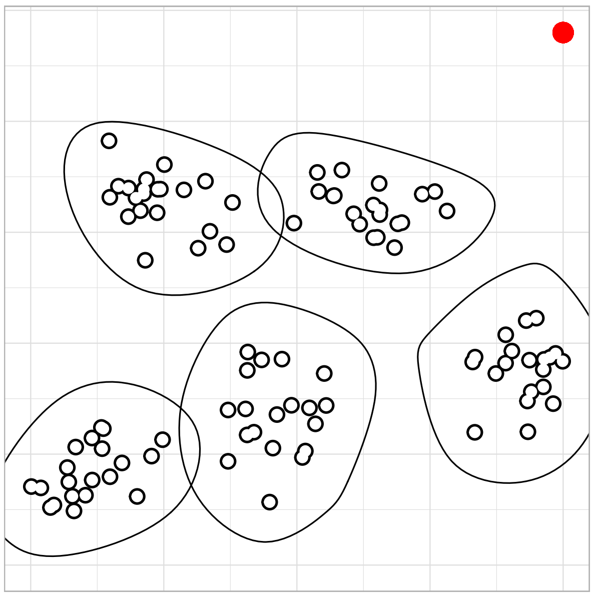

Figure A.2: *Randomly 5-cluster generated network of 100 customers. Each cluster is encircled for visualization. The depot is the red dot in the top right corner.*