ERASMUS UNIVERSITY ROTTERDAM

MASTER'S THESIS

Improved methods for solving the dynamic assignment vehicle routing problem using historical data, an insertion heuristic and a genetic algorithm

Author:
E. A. VERZIJLBERGEN

Student ID:
435397

Supervisor Erasmus University Rotterdam:
Dr. T. DOLLEVOET

Examiner Erasmus University Rotterdam:
Prof. Dr. A. P. M. WAGELMANS

Supervisor TNO:
Dr. F. PHILLIPSON

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in

Econometrics and Management Science
- Operations Research and Quantitative Logistics -
Department of Econometrics

January 25, 2021

ERASMUS UNIVERSITY ROTTERDAM

Abstract

Erasmus School of Economics
Department of Econometrics

Master of Science

Improved methods for solving the dynamic assignment vehicle routing problem using historical data, an insertion heuristic and a genetic algorithm

by E. A. VERZIJLBERGEN

Parcel delivery services distribute thousands of parcels a day, from various depots. Some of these depots have limited storage capacity, therefore parcels need to be assigned directly to distribution vehicles upon arrival. In literature this problem is known as a dynamic assignment vehicle routing problem. This thesis investigates two tracks to a three step approach to solve this problem in the request of TNO, an independent research organisation, in collaboration with DPD, an international parcel delivery service. In the first step dummy parcels are generated based on historical data, in the second step the dummy parcels are assigned to different distribution vehicles and in the third step dummy parcels are one by one replaced by arriving parcels, referred to as dynamic parcels. In the first track the assignment of dummy parcels to vehicles is done by applying a vehicle routing problem heuristic based on a sequential tour-building insertion heuristic. In the second track a decomposition method is applied, in which the problem is divided in a main clustering problem and a smaller set of travelling salesman sub-problems. A genetic algorithm is developed to solve the clustering problem and a heuristic algorithm to solve the sub-problems. Both tracks are applied to instances created using data provided by DPD. Results indicate that the vehicle routing problem heuristic track performed best, considering the total travelling distance and the number of vehicles used. Due to infeasible computation time this method was not able to generate a solution for all instances considered. The decomposition method track generated routes performing better than those of the benchmark solutions. The implication of these results is that the three step approach introduced in this thesis generates routes solving the dynamic assignment vehicle routing problem, where the two tracks introduce methods applicable in each step.

Acknowledgements

Throughout the writing of this thesis and in the past years completing my Bachelor Applied Mathematics at Delft University of Technology and my Master Econometrics and Management Science I have received a great deal of love, encouragement, support and assistance.

First of all, I would like to thank my supervisor, Dr. T. (Twan) Dollevoet, who helped formulating this thesis. Your help and feedback brought my work to a higher level. Prof. Dr. A. P. M. Wagelmans, thank you for being the co-reader of my thesis.

I would like to acknowledge my colleagues from the TNO Cyber Security and Robustness department. I am grateful to Dr. F. (Frank) Phillipson for providing me with a thesis internship during a time in which working from home is the standard and maintaining your work is challenging, never mind providing guidance to an intern. I would like to thank I. (Irina) Chiscop and my fellow interns with who I met on a regular basis for a virtual cup of tea to discuss our day to day lives and all the obstacles we encountered regarding our thesis and working from home.

My sincere thanks to my family, Conny, Theo and Rolf and my boyfriend Owen for the love, support and encouragement. The past years would not have been the same without all of my friends, thank you, Emma, Daiva, Lot, Ruth, Yanna, Thomas, Ilona, Joyce, Amila, Esther, Anne-Marte, Eva, Mathilde, Puck, Ires, Annemar, Lizan, Anna Lisa, Atty, Britt, Laurence, Gitte, Myrthe and Sophie.

Contents

List of abbreviations

CVRP	The **c**apacitated **v**ehicle **r**outing **p**roblem
DAVRP	The **d**ynamic **a**ssignment **v**ehicle **r**outing **p**roblem
DAVRPTW	The **d**ynamic **a**ssignment **v**ehicle **r**outing **p**roblem with **t**ime **w**indows
GAP	The **g**eneralised **a**ssignment **p**roblem
MST	The **m**inimum **s**panning **t**ree
OSRM	The project **O**pen **S**ource **R**outing **M**achine
TSP	The **t**ravelling **s**alesman **p**roblem
TSPTW	The **t**ravelling **s**alesman **p**roblem with **t**ime **w**indows
VRP	The **v**ehicle **r**outing **p**roblem
VRPTW	The **v**ehicle **r**outing **p**roblem with **t**ime **w**indows

Abbreviations used in the tables of Chapter 4:

#DP	The number of **d**ummy **p**arcels
#P	The number of (dynamic) **p**arcels
#PC	The number of **p**arent **c**hromosomes
#Pool	The size of the **pool** after five generations
#T	The number of **t**ours and therefore the number of required vehicles
AD	The **a**verage **d**uration of a tour
AP	The average duration of the **a**ssignment of a dynamic **p**arcel to a vehicle
DM	The computation time of the **d**ecomposition **m**ethod
IG	The number of chromosomes in the **i**ntial **g**eneration
LS	The computation time of the **l**ocal **s**earch after the correspond method
LSDA	The computation time of the **l**ocal **s**earch during the **d**ynamic **a**ssignment
TD	The total **t**ravelling **d**istance
VRP	The computation time of the **v**ehicle **r**outing **p**roblem heuristic

Chapter 1

Introduction

With the rise of e-commerce, parcel delivery service is a growing sector. In a progressive country such as the Netherlands, delivery companies keep trying to surpass one another, each providing innovative features, such as time slot delivery, track-and-trace codes and pick-up points. To stay innovative, budget is needed, which is gained easiest by reducing delivery expenses. By applying efficient planning and scheduling methods, it is possible to reduce the number of trucks, total travelling distance, number of required drivers and total working hours, while delivering the same number of parcels.

This thesis solves a dynamic assignment vehicle routing problem (DAVRP), in which the total travelling distance will be minimised. In a DAVRP, parcels arriving at the depot must immediately be assigned to a distribution vehicle, all this without the knowledge of the number of incoming parcels and their destinations. The DAVRP solved in this thesis aims to assign all incoming parcels to vehicles in such a way that the total travelling distance is minimised, whilst keeping in mind the maximum working hours of the drivers.

By dividing the problem in three steps, it is possible to apply different approaches in each step. Among these approaches are original ideas developed in this thesis, as well as known methods stemming from literature. Two tracks, both of the tree step approach, are investigated to solve the DAVRP and a comparison is based on performance with regards to solution quality and computation time. Both tracks generate a dummy data set in the first step using weighted K-means clustering. In the second step, the dummy parcels are routed by using a sequential tour-building insertion heuristic or a decomposition method, dividing the problem in a main cluster problem and many smaller travelling salesman sub-problems. In the third step the dummy parcels are one by one replaced by incoming parcels. The results of this thesis show that all routes generated using the vehicle routing problem heuristic track or the decomposition method track reduced the total travelling distance, while keeping the route duration less than nine hours and minimising the number of vehicles used compared to the benchmark solutions.

An evaluation of earlier research regarding the subjects discussed in this thesis is given in Section 1.1. Section 1.2 gives a thorough description of the problem statement.

1.1 Literature review

The vehicle routing problem (VRP) is a combinatorial optimisation problem that has been studied widely and thoroughly. To solve a VRP many exact and heuristic algorithms have been proposed. The VRP is classified as an NP-hard problem (non-deterministic polynomial-time hardness) as is shown in Garey and Johnson (1979). Hence, the use of exact optimisation methods to solve these problems may be problematic in acceptable computation times, when the problem involves real-world data sets that are extensive. The VRP is considered more computationally difficult than the travelling salesman problem (TSP), which is the simplest case of a VRP. In a TSP, a single vehicle is responsible for delivering parcels to multiple locations,

referred to as customer nodes. A route, beginning and ending at a given location, called the depot, is generated to attain the maximum possible reward. The reward is often defined as the negative of the total travelling distance or duration of the vehicle.

Bellmore and Nemhauser (1966) proposed a nearest neighbour heuristic. This heuristic starts at an arbitrary customer, then continuously adds the nearest unvisited customer until all have been visited. Another well known heuristic to solve a TSP, is the double tree heuristic. In this heuristic a spanning tree is constructed in polynomial time using the algorithm provided by Kruskal (1956) or Prim (1957), and a Eulerian cycle is constructed in polynomial time using the algorithm by Fleury (1883). Combined, these steps result in a polynomial time heuristic algorithm to solve a TSP. In Christofides (1976), a heuristic to solve a TSP combines polynomial time algorithms to generate a spanning tree, a perfect matching and a Eulerian cycle, such that it is able to construct a tour.

In case of multiple vehicles, in which each has the same maximum capacity, the objective is to optimise a set of routes, each returning to the depot, in order to attain the maximum possible reward without violating the maximum capacity of a vehicle. This problem is referred to as a capacitated vehicle routing problem (CVRP). The capacity constraint is considered in the papers below.

Many vehicle routing problems involve scheduling visits to customers who are only available during specific time windows. These problems are known as vehicle routing problems with time windows (VRPTW). For doing so, a set of vehicles (with limited capacity) must be routed along the customers which have predefined time windows and known demands. The number of used vehicles and the total travelling distance of the vehicle is minimised, while meeting the capacity and time window constraints. Solomon (1987) states that finding a feasible solution to the VRPTW when the number of vehicles is fixed is an NP-complete problem. Solomon (1987) extended multiple VRP heuristic algorithms to solve the VRPTW: a saving heuristic, based on the consideration of whether it is profitable to combine two locations in one tour or visit each separately; a time-oriented nearest neighbour heuristic, based on a nearest neighbour principle where closeness is defined as a weighted combination of their direct spatial distance, time difference and urgency of a delivery; an insertion heuristic, customer and place of insertion get chosen based on three different weighted combinations of distance expansion, time expansion and urgency of serving a customer; and a time-oriented sweep heuristic, consisting of a clustering and tour building phase.

In Cheng and Wang (2009) a VRPTW is solved by decomposing the problem into a clustering problem (main problem) and a set of travelling salesman problems (sub-problems). To solve the clustering problem, a genetic algorithm is developed and the set of sub-problems is solved by applying a heuristic algorithm to solve a travelling salesman problem. The solution of the VRPTW is obtained by iteratively solving the main problem and the sub-problems.

Vehicle routing problems can be dynamicised in multiple ways. In dynamic vehicle routing problems (DVRP), the customers' demands appear with time, and the unserved customers' points must be updated and rearranged while executing the routes. In Pillac et al. (2013) four categories of routing problems are identified.

In *static and deterministic* problems, all input needed to generate routes is known beforehand, therefore routes can be entirely constructed before the vehicle leaves the depot. In *static and stochastic* problems, not all input needed to generate routes is known beforehand. Part of the input is defined as random variables, which are revealed throughout the execution of the routes. Routes are constructed before the vehicle leaves the depot, and small changes may occur while executing the route, so is it possible to plan an extra return to the depot or skip customers. Decisions regarding the changes of the route can be made by the driver himself, no remote decision makers are needed to reschedule parts of the route, since only small changes are allowed. In *dynamic and deterministic* problems, not all input needed to generate routes

is known beforehand. Part of the input is revealed throughout the execution of the routes. Decisions regarding the construction of the route cannot be made by the driver himself, communication with a remote decision maker is maintained. Similarly, in *dynamic and stochastic* problems, not all input needed to generate routes is known beforehand. Part of the input is revealed throughout the execution of the routes, in contrast with the previous category, some input is defined as random variables. Again, the decisions regarding the construction of the route cannot be made by the driver himself, communication with a remote decision maker is maintained.

In Phillipson and de Koff (2020) the dynamic assignment vehicle routing problem (DAVRP) is introduced. This is part of the *dynamic and deterministic* VRPs, however, the assignment of the parcels is performed at the same time the destination is revealed. This means that the planning is performed dynamically, but in contrast to the common dynamic case, it is done upfront, when the route is not yet in execution, giving the possibility to change the order per route, but not to interchange parcels between the routes.

This problem is extended in Phillipson, de Koff, et al. (2020), with a generalisation of the definition of capacity, with an unknown workload, unknown number of parcels per day, and a generalisation of the objective function. Here, in the last post-processing step, they combine or integrate smaller tours and assign these larger tours to vehicles.

In Los et al. (2020), the DAVRP is further extended, by introducing time-windows to the problem, making it the Dynamic Assignment Vehicle Routing Problem with Time Windows (DAVRPTW). Introducing time windows makes it even more complicated. The method presented in Phillipson, de Koff, et al. (2020) for the general DAVRP case is not suitable to be extended for handling time windows, due to the optimisation step in the end. The approach presented in Los et al. (2020), divides the problem in two sub-problems, namely the generalised assignment problem (GAP) and multiple travelling salesman problems with time windows (TSPTW). They propose an interactive approach between the two. Solutions are generated using a three step approach:

1. Create an initial solution based on a dummy sample,
2. Create an improved solution based on the dummy sample,
3. Create a dynamic solution, substituting each dummy parcel with a real dynamic incoming parcel.

The dummy sample in Step 1 is created using randomly generated data or historical data. In Step 2 the initial solution is improved by using interroute and intraroute local search methods. In Step 3, the main idea is that dummy parcels are one by one replaced by incoming parcels.

From the papers discussed above, the approach in Los et al. (2020) struck as a way to divide the problem cleverly, offering many options to make adjustments. This thesis aims to use the historical data in a more elaborate way to generate the dummy parcels. To create a solution based on the dummy parcels, Los et al. (2020) used an insertion heuristic, which this thesis compares to another insertion heuristic based on the insertion heuristic proposed in Solomon (1987) and a decomposition method based on the decomposition method proposed in Cheng and Wang (2009). The combination of the insertion heuristic, which tackles the problem as a whole, and the decomposition method, which decomposes the problem in a main clustering problem and many smaller travelling salesman sub-problems provides an adequate balance between the two different approaches. To generate solutions to the dynamic part of the problem, the method proposed to substitute each dummy parcel with an incoming parcel, Step 3 from Los et al. (2020), will be used.

1.2 Problem description

This thesis proposes methods to construct a set of routes so that each parcel arriving at the depot will be delivered to the customer. It computes these routes based on data provided by DPD for the months September, October, November and December of the year 2018. Part of the data will be used as historical data, classified as known, another part will be used as validation data and therefore classified as unknown.

Starting from a regional distribution centre or depot, different routes have to be determined to deliver all parcels with minimum cost, by assigning each parcel to a specific vehicle. A depot has no space available to store all parcels before assigning them to the different vehicles. Therefore, a parcel is scanned upon arrival and immediately assigned to a vehicle. This defines the dynamic element of our vehicle routing problem and this distinguishes it from a regular vehicle routing problem. Reallocation of parcels is not allowed, therefore the assignment to a specific vehicle is of the utmost importance. The route, which will be executed by a specific vehicle, can be regenerated after each assignment, since the vehicle leaves after all parcels have been assigned.

First and foremost, the total travelling distance must be minimised, meaning the sum of the distances of routes driven by all vehicles. When constructing the routes, some constraints are taken into account. The total travelling time of each route needs to be considered, as drivers may not work for a period longer than nine hours. This can be interpreted as a time window assigned to each parcel with a duration of nine hours. No individual time windows will be considered, but the nine hour constraints will require a time element in the algorithms used. DPD is expected to have an unlimited number of vehicles at their disposal, therefore one of the aims is to optimise the number of vehicles used.

1.3 Structure

This thesis is structured as follows. Chapter 2 provides the methodology, which is divided in a three step approach. In the first step, dummy data is created using historical data, this is discussed in Section 2.1. Section 2.2 explains how to solve the VRP based on the dummy data using two different methods, a VRP heuristic, discussed in Section 2.2.1, and a decomposition method, discussed in Section 2.2.2. Both methods are followed by a local search, discussed in Section 2.2.3. The last step is to substitute each dummy parcel with a real dynamic incoming parcel, which is discussed in Section 2.3. Chapter 3 elaborates on the data provided by DPD. Computational results regarding the three steps of both tracks are thereafter stated and discussed in Chapter 4. Chapter 5 states the findings and discusses the influence of the methodological decisions made. Finally, conclusions and recommendations for further research are given in Chapter 6.

Chapter 2

Methodology

This section describes two tracks to a three step approach to solve the dynamic assignment vehicle routing problem. As discussed before, in Los et al. (2020) a three step approach is introduced, which first creates an initial solution based on dummy samples, then optimises this solution and finally the dummy parcels are substituted with real parcels to create the dynamic solution. Derived from this, an altered and improved three step approach is introduced:

1. Create dummy data based on historical data;
2. Create an initial solution;
3. Create a dynamic solution.

In the first step, historical data is combined in a way that allows to predict where parcels for the upcoming day need to be delivered, these predicted parcels are called dummy parcels. The second step solves the vehicle routing problem based on these dummy parcels. The first method introduced does so by applying a VRP heuristic. The second method decomposes the original problem in a clustering problem (main problem) and a set of travelling salesman problems (sub-problems). The solution is obtained through iterative interactions between the main problem and the set of sub-problems. In the third step, the dummy parcels are one by one replaced by incoming parcels, resulting in a solution to the dynamic assignment vehicle routing problem.

In Figure 2.1, the three step approach to solve the dynamic assignment vehicle routing problem is illustrated. In the figure, the definitions of terminology used throughout this thesis is provided; the parts within the dotted boxed are referred to as the steps of the approach; the part within the solid boxes are referred to as the methods; and following the paths indicated by arrows on the left and right are referred to as the tracks of the approach.

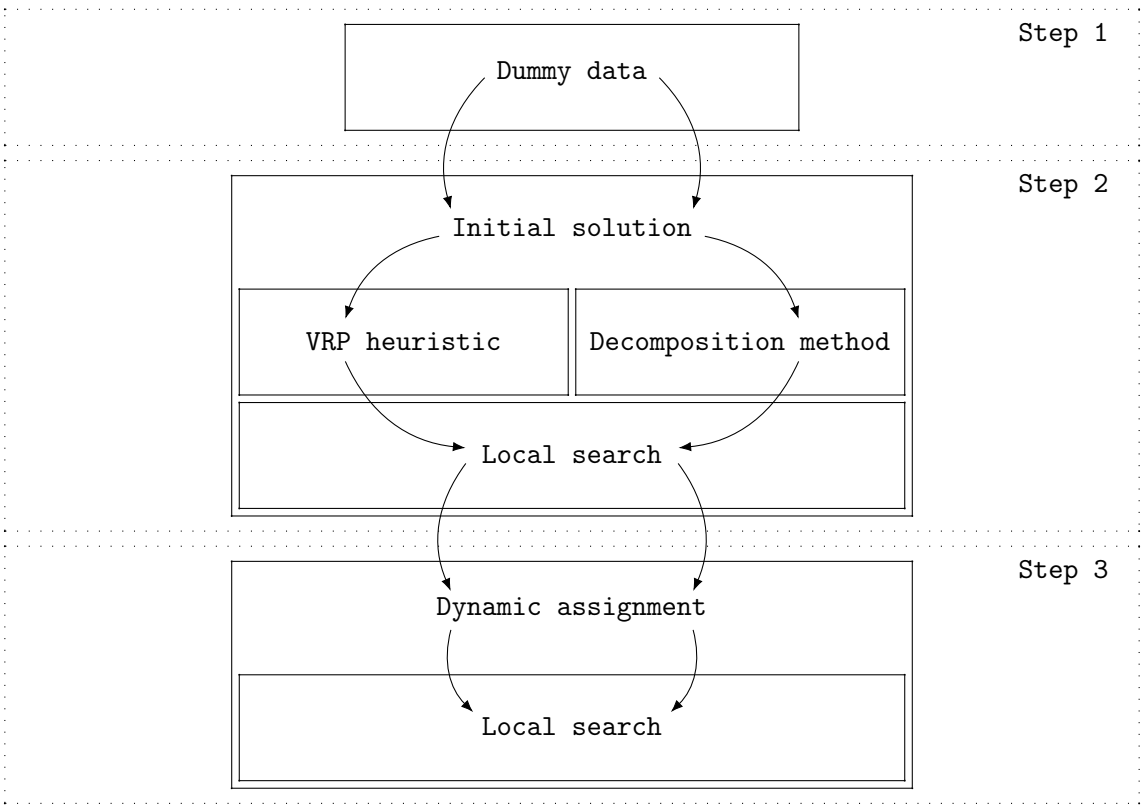

FIGURE 2.1: An illustration of terminology used throughout this thesis. The parts within the dotted boxes are referred to as the steps of the approach; the part within the solid boxes are referred to as the methods and following the paths indicated by arrows on the left and right are referred to as the tracks of the approach.

2.1 Dummy data

In this step historical data will be used to generate a set of dummy parcels, enabling the construction of an initial solution. In Los et al. (2020) the historical data at their disposal was not used extensively. The set of dummy parcels was a randomly generated set or a direct copy of parcels delivered earlier. This step is elaborated upon in this altered approach. The dummy data is supposed to be a representation of the incoming parcels, which are unknown at the moment of executing this step. This subsection explains how to generate the set of dummy parcels.

First, relevant historical data is gathered. For example, Saturday 15/09/2018, Saturday 22/09/2018 and Saturday 29/09/2018 are used to generate the dummy data for Saturday 06/10/2018. All the parcels delivered on the 15th, 22nd and 29th are combined in one set, but some parcels get assigned a different importance rate than others. This importance rate is referred to as weight. A weight is assigned to a parcel, depending upon the number of parcels delivered on the corresponding day. The more parcels delivered on the corresponding day, the lower the assigned weight. A parcel delivered on a day with only a few parcels is expected to be of greater importance to the dummy data set, than one delivered on a day with many deliveries. This can be explained using the following example: on the days preceding *Sinterklaas* a Dutch holiday, more parcels get delivered than on a regular day. Many people, who normally do not receive parcels often, might order presents needed for Sinterklaas at webshops. These customers are of less importance to the predictions than parcels delivered to customers who order on a regular basis. Therefore a parcel delivered on a busy day preceding Sinterklaas is

assigned a lower weight than a parcel delivered on a regular day.

By considering multiple days as historical data, outliers with regular deliveries become of greater importance. If these outliers show up in multiple days of the historical data set, they will be better represented by a dummy parcel in the dummy data set.

The actual dummy parcels are generated using a clustering method, the centres of the clusters are used as dummy parcels. The number of generated clusters, resulting in the number of dummy parcels, is based on the average number of parcels delivered on the days used as historical data. For example, if a customer receives a parcel on the 15th, 22nd and 29th, it is very likely that these three parcels will be combined in one cluster, the centre, in this case, is placed exactly at the customer location, and this centre is assigned a dummy parcel.

As a clustering method the weighted K-means clustering is used. This method minimises the weighted within-cluster sum of squares. Formally, this algorithm aims at minimising the objective function:

$$\sum_{N_i} \sum_{n=1}^{\#N_i} \sum_{k=1}^{K} \mathbb{1}_{\{c_n^{(i)}=k\}} ||x_n^{(i)} - \mu_k||^2 w_n^{(i)}, \tag{2.1}$$

where $N = N_1 \cup N_2 \cup .. \cup N_I$, with N_i being the observations of an individual day and $x_n^{(i)} \in N_i$. K is the number of cluster, with centroids $\{\mu_1, ..., \mu_K\}$, which will later-on correspond to the dummy parcels. $c_n^{(i)} \in \{1, ..., K\}$ indicates which of the K clusters the observation $x_n^{(i)}$ belongs to, and $w_n^{(i)}$ is the weight used to take the weighted distance. $w_n^{(i)}$ is defined as:

$$w_n^{(i)} = \frac{1}{\#N_i}. \tag{2.2}$$

This algorithm first assigns all customers to a random cluster, these serve as the first clustering. The following steps are performed iteratively until the clusters stop changing. For each of the clusters, the cluster centroid is computed. After that, the customers are assigned to the cluster whose centroid is closest. Once the algorithm has converged, the centroids are returned as dummy parcels. Now, using a clustering method, a dummy data set has been generated based on historical data.

2.2 Initial solution

This step will solve the vehicle routing problem based on the data set previously generated in the first step. Generating a solution can be done using a wide variety of methods. The official mathematical formulation of the vehicle routing problem will be given, but first some variables need to be defined. Let N be the set of all customers, it should be noted that this set is based on the cluster centroids of the previous step. K is the set of all vehicles. The decision variable x_{ijk} is 1 if vehicle k travels from customer i to customer j, and 0 otherwise. The decision variable y_{ik} is 1 if vehicle k serves customer i. The variable d_{ij} corresponds to the travelling distance from customer i to j. $N' = N \cup \{0\}$, where 0 denotes the depot. The parameter s_i is the service time of customer i, note that $s_0 = 0$. The parameter t_{ij} corresponds to the time needed to travel from customer i to j. Variable t_i is the arrival time at customer i. r_k is the maximum travelling time of vehicle k. M is a large number. The mathematical formulation is given below and based on

the formulation given in Cheng and Wang (2009):

$$\text{minimise} \sum_{i\in N'}\sum_{j\in N'}\sum_{k\in K} d_{ij}x_{ijk}, \tag{2.3}$$

$$\text{such that} \quad \sum_{k\in K} y_{ik} = 1, \qquad \forall i \in N \tag{2.4}$$

$$\sum_{i\in N'} x_{ijk} = y_{jk}, \qquad \forall j \in N', \quad \forall k \in K, \tag{2.5}$$

$$\sum_{j\in N'} x_{ijk} = y_{ik}, \qquad \forall i \in N', \quad \forall k \in K, \tag{2.6}$$

$$t_i + s_i + t_{ij} - M(1 - x_{ijk}) \leq t_j, \qquad \forall i, j \in N, \quad \forall k \in K, \tag{2.7}$$

$$\sum_{i\in N'}\sum_{\substack{j\in N' \\ j\neq i}} x_{ijk} \cdot (t_{ij} + s_j) \leq r_k, \qquad \forall k \in K, \tag{2.8}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i, j \in N', \quad \forall k \in K, \tag{2.9}$$

$$y_{ik} \in \{0,1\} \qquad \forall i \in N', \quad \forall k \in K. \tag{2.10}$$

The objective function (2.3) minimises the total distance travelled by all vehicles. Constraints (2.4) make sure each customer is visited by one vehicle only. Constraints (2.5) and (2.6) make up the balance constraints of each customer. The precedence relation between two successive customers is expressed by Constraints (2.7), and the duration of the route is restricted by Constraints (2.8).

This thesis regards two different methods to solve this vehicle routing problem. In Section 2.2.1, a vehicle routing problem heuristic is discussed. In Section 2.2.2, a decomposition method is discussed which iteratively solves a main problem and a set of sub-problems.

2.2.1 Vehicle routing problem heuristic

This section discusses a method to solve a VRP by using a heuristic algorithm. This method is based on the sequential tour-building insertion heuristic first proposed in Solomon (1987). First, the parameters and variables need to be defined before providing an in-depth explanation of the heuristic. Let N be the set of customers. The delivery at customer i, $i \in N$, involves the delivery of goods for s_i units of time, and can begin at t_i, within a time window with earliest time e_i and latest time l_i, that customer i will permit the start of service. Note that t_i for $i = 1, ..., n$, at which services begin are decision variables. This heuristic is originally meant for vehicle routing problems with time window constraints. In case the only time restriction is the maximum total travelling time of a vehicle, all time windows can be changed with earliest time e_i and latest time l_i for customer i, to a time window of the maximum travelling time of a vehicle independent of the customer, $[e, l]$. Now t_i corresponds to the arrival time as well as the time at which a delivery starts, since the arrival time cannot correspond to a time earlier than e (by then the vehicle has not yet left the depot).

First the required conditions for time feasibility are considered when inserting a customer u in a partially existing route. Consider an insertion of u between customer i_{p-1} and i_p, $1 \leq p \leq q$, on a partially constructed route, $(i_0, i_1, i_2, ..., i_q)$, $i_0 = i_q = 0$, where 0 represents the depot, for which the times to begin services, t_{i_r}, for $0 \leq r \leq q$, are known. Assumed is that the vehicle leaves as early as possible, at time e. Denote by $t_{i_p}^{\text{new}}$ the new time for service to begin at customer i_p, give that u is inserted. Since it is assumed that the triangle inequality holds for both travel distances and times, this insertion defines a *push forward* in the schedule at i_p:

$$\text{PF}_{i_p} = t_{i_p}^{\text{new}} - t_{i_p} \geq 0.$$

Furthermore,

$$\mathrm{PF}_{i_{r+1}} = \mathrm{PF}_{i_r}, \qquad p \leq r \leq q-1.$$

This equality holds, since there are no time windows for individual customers and therefore a vehicle never has to wait at a customer until it may start with service. Note that the push forward does not decrease for customers served after customer i_p. As is mentioned in Solomon (1987), "one can also define a *push backward* in the schedule at i_r for $r = 0, ..., p-1$. However, this concept is not appealing, since the current schedule can be pushed backward only if the vehicle leaves the depot later than e. Note that by initially assuming that each vehicle leaves the depot at e, the push forward is used". It can be concluded that:

Lemma 2.2.1 *The necessary and sufficient condition for time feasibility when inserting a customer u, between customers i_{p-1} and i_p, $1 \leq p \leq q$, on a partially constructed feasible route $(i_0, i_1, i_2, ..., i_q)$, $i_0 = i_q = 0$ is*

$$t_{i_q}^{new} = t_{i_q} + PF_{i_q} \leq l$$

Note that since $i_q = 0$, the use of Lemma 2.2.1 will ensure that any customer that does not permit the vehicle to return to the depot within the maximum travelling time, will not be added to the partial route.

Using Lemma 2.2.1, time feasibility can be checked and the actual routes can be constructed. The heuristic used is based upon the insertion heuristic proposed in Solomon (1987). This method is a sequential tour-building heuristic, meaning that it builds a single route at a time. Once no more customers can be inserted it starts building the next. The initialisation of a new route is done by constructing a route from the depot, back and forth to the furthest unrouted customer. At every iteration a new customer u will be inserted in the current route between two adjacent customers i and j, this is done using two criteria, $c_1(i,u,j)$ and $c_2(i,u,j)$. These two criteria will be defined later, the way of defining depends on the preferred relation between serving a customer and its travelling distance and time added to the route when served.

Define $(i_0, i_1, i_2, ..., i_q)$ to be the current partial route, with $i_0 = i_q = 0$, where 0 represents the depot. For each customer u, the best feasible insertion place in the emerging route is first computed:

$$c_1(i(u), u, j(u)) = \text{minimum}_{p=1,...,q}[c_1(i_{p-1}, u, i_p)]. \tag{2.11}$$

The feasibility of the insertion place is checked using Lemma 2.2.1. Secondly, the best customer u to be inserted in the route is selected as the one for which

$$c_2(i(u^*), u^*, j(u^*)) = \text{optimum } [c_2(i(u), u, j(u))], \tag{2.12}$$

where costumer u is unrouted and feasible. Customer u^* is then inserted in the route between $i(u^*)$ and $j(u^*)$. When no more customers with feasible insertions can be found, the method starts a new route, unless it has already routed all customers.

Now the two criteria $c_1(i,u,j)$ and $c_2(i,u,j)$ are defined more specifically and in detail. Two different approaches are regarded. The first is formulated as:

$$c_1(i,u,j) = \alpha_1 c_{11}(i,u,j) + \alpha_2 c_{12}(i,u,j), \qquad \alpha_1 + \alpha_2 = 1, \quad \alpha_1, \alpha_2 \geq 0, \tag{2.13}$$

$$c_2(i,u,j) = \lambda d_{0u} - c_1(i,u,j), \qquad \lambda \geq 0. \tag{2.14}$$

With,

$$c_{11}(i,u,j) = d_{iu} + d_{uj} - \mu d_{ij}, \qquad \mu \geq 0, \tag{2.15}$$

$$c_{12}(i,u,j) = t_j^{\text{new}} - t_j. \tag{2.16}$$

Where t_j^{new} is the new time for service to begin at customer j, given that u is inserted. This approach is based on the principle that it is more beneficial to serve a customer in the partial route rather than on a direct route, as expressed in Equation (2.14). The best feasible insertion place for an unrouted customer is the one that minimises the weighted combination of the added distance and time after insertion, as expressed in Equation (2.13). Different values of λ, μ, α_1 and α_2 do result in different routes being generated.

The second type of insertion heuristics aims to select customers whose insertion costs minimise a measure of total route distance and time, and is formulated as:

$$c_1(i,u,j) \text{ is defined as before in Equation (2.13)} \tag{2.17}$$

$$c_2(i,u,j) = \beta_1 R_d(u) + \beta_2 R_t(u), \qquad \beta_1 + \beta_2 = 1 \quad \beta_1 \geq 1 \quad \beta_2 > 0, \tag{2.18}$$

where $R_d(u)$ and $R_t(u)$ are the total distance and time of the current partial route, given that u is inserted. Different values of μ, β_1 and β_2 do result in different routes being generated.

2.2.2 Decomposition method

This section discusses a method to solve a VRP by decomposing the problem into a clustering problem (main problem) and a set of TSPs (sub-problems). A genetic algorithm is developed to solve the clustering problem and a heuristic algorithm to solve the set of travelling salesman sub-problems. The solution of the original problem is obtained through iterative interactions between the main problem and the set of sub-problems.

The mathematical formulation given in Section 2.2 can be decomposed into a main and sub-problem. The variable y_{ik}, $i \in N$, determines which customer should be served by vehicle $k \in K$. By way of explanation, for the total number of m used vehicles, this variable clusters all customers into m groups. After y_{ik} is determined, the variable x_{ijk}, $\forall i, j$, decides the travelling sequence of the customers in the group that is served by vehicle k. Let the variable z_k denote the cost of clustering certain customers into group k, then the clustering problem can be defined as the minimisation of the total cost of a clustering result. This problem is referred to as the main problem of the model and is formulated as:

$$\text{minimise} \qquad \sum_{k \in K} z_k, \tag{2.19}$$

$$\text{such that} \qquad \sum_{k \in K} y_{ik} = 1, \qquad \forall i \in N \tag{2.20}$$

$$y_{ik} \in \{0,1\} \qquad \forall i \in N, \quad \forall k \in K. \tag{2.21}$$

The objective function (2.19) minimises the total cost of a clustering result. Constraints (2.20) make sure each customer is visited by one vehicle only, these correspond to Constraints (2.4).

Based on the formulation given in Equations (2.3) - (2.10), the cost of clustering certain customers in group k is defined as the total travelling distance of a single vehicle serving all customers assigned to group k. The total travelling distance is obtained by solving the travelling salesman problem corresponding to group k. Let N_k be the set of all customers served by

vehicle k. $N'_k = N_k \cup \{0\}$, where 0 denotes the depot. The kth sub-problem, i.e., a travelling salesman problem, is formulated as:

$$\text{minimise} \qquad z_k = \sum_{i \in N'_k} \sum_{j \in N'_k} d_{ij} x_{ijk}, \tag{2.22}$$

$$\text{such that} \qquad \sum_{i \in N'_k} x_{ijk} = 1, \qquad \forall j \in N'_k, \tag{2.23}$$

$$\sum_{j \in N'_k} x_{ijk} = 1, \qquad \forall i \in N'_k, \tag{2.24}$$

$$t_i + s_i + t_{ij} - M(1 - x_{ijk}) \leq t_j, \qquad \forall i, j \in N_k, \tag{2.25}$$

$$\sum_{i \in N_k} \sum_{\substack{j \in N_k \\ j \neq i}} x_{ijk} \cdot (t_{ij} + s_j) \leq r_k, \tag{2.26}$$

$$x_{ijk} \in \{0, 1\} \qquad \forall i, j \in N'_k. \tag{2.27}$$

The objective function (2.22) minimises the total travelling distance of a cluster. Constraints (2.23) and (2.24) make up the balance constraints of each customer, these correspond to constraints (2.5) and (2.6). Constraints (2.25), correspond to Constraints (2.7), and express the precedence relation between two successive customers. Constraints (2.26), correspond to Constraints (2.8), and restrict the duration of the route.

To come to a solution of the VRP, the main problem and the many sub-problems need to be solved iteratively. This process is illustrated in Figure 2.2. The main problem formulates m clusters corresponding to m vehicles, and passes these to the sub-problems. For each of these clusters, the values of z_k are calculated, the total travelling distance within a cluster. A summation of all these distances is then returned to the main problem as a performance rate of this clustering. Depending on the performance, a new clustering will be formulated.

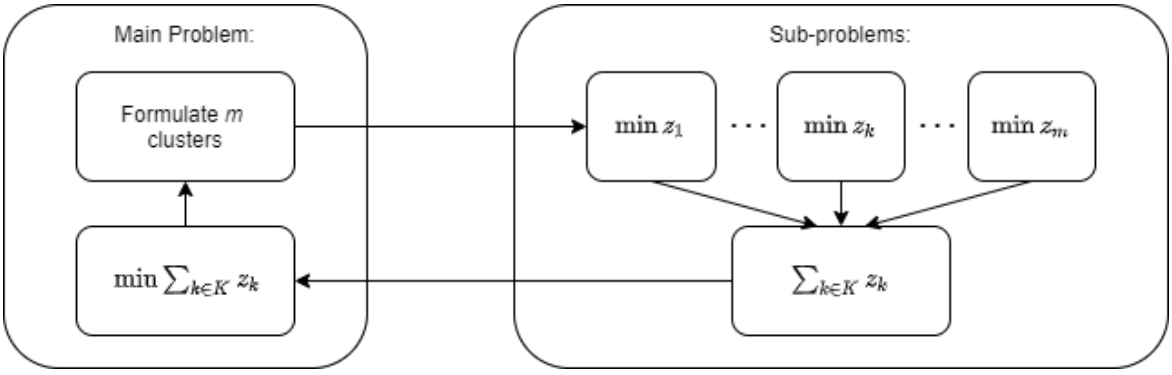

FIGURE 2.2: Interaction between the main problem and sub-problems.
Source: Cheng and Wang 2009, Figure 2.

The main problem is solved using a genetic algorithm. The decomposition of the VRP does result in a set of smaller problems, nonetheless these smaller problems are still difficult to solve. Therefore, the sub-problems, which are travelling salesman problems, will be solved by a heuristic algorithm.

As described in Vyas (2018), "a genetic algorithm is a search heuristic that is inspired by Charles Darwin's theory of natural evolution. This algorithm reflects the process of natural selection,

where the fittest individuals are selected for reproduction in order to produce offspring, generating the next generation".

The decision variable, y_{ik} of the main problem is represented as a list which is referred to as a chromosome. In the classical approach a single chromosome consists of zeros and ones where the index of the ones indicate the customers in a single cluster, see Figure 2.3 for the classical representation. The classical approach requires a great amount of storage. Therefore, an alternative way to represent the clusters is used. Here, the numbers indicate the cluster to which the index is assigned, see Figure 2.4 for the alternative representation. Looking at the example in Figure 2.3, it can be concluded that customers 1, 3 and 7 are served by vehicle 1, customers 2, 6 and 9 by vehicle 2 and customers 4, 5 and 8 by vehicle 3. All this information can now be stored in a single chromosome using the alternative approach illustrated in Figure 2.4. The values of y_{ik}, will be given as $y_{11} = y_{22} = y_{31} = y_{43} = y_{53} = y_{62} = y_{71} = y_{83} = y_{92} = 1$ and $y_{ik} = 0$ for all other combinations of i and k.

	1	2	3	4	5	6	7	8	9
chromosome 1	1	0	1	0	0	0	1	0	0
chromosome 2	0	1	0	0	0	1	0	0	1
chromosome 3	0	0	0	1	1	0	0	1	0

FIGURE 2.3: The classical representation of chromosomes. Here, customers 1, 3 and 7 are served by vehicle 1, customers 2, 6 and 9 by vehicle 2 and customers 4, 5 and 8 by vehicle 3.

	1	2	3	4	5	6	7	8	9
chromosome 1	1	2	1	3	3	2	1	3	2

FIGURE 2.4: The alternative way to represent a chromosome. Here, customers 1, 3 and 7 are served by vehicle 1, customers 2, 6 and 9 by vehicle 2 and customers 4, 5 and 8 by vehicle 3.

To determine the performance of a single chromosome, the fitness function is used. Thus, the fitness function is defined by the objective function of the clustering (main) problem. Meaning, the fitness of a chromosome is valued by the total travelling distance of all vehicles within this clustering.

Each step of the algorithm will now be discussed. A structured and detailed description will be provided later. A flow chart of the algorithm can be found in Figure 2.5.

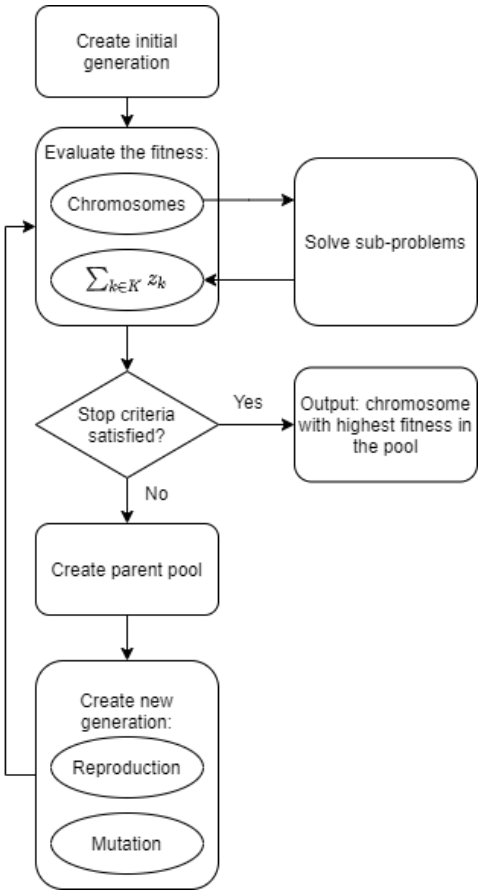

FIGURE 2.5: Flowchart of the decomposition method.

For starters, a set or generation of chromosomes is created. Meaning: in each clustering, each customer is assigned to a vehicle. This thesis discusses three ways to generate a chromosome. First of all a chromosome can be generated randomly, here each customer is assigned to a vehicle at random. The steps of the algorithm for randomly generating a chromosome are described as follows:

Algorithm 1: Random chromosome generation	
Step 0.	Initialisation: Let 0 be the depot. Let N be the set of customers.
Step 1.	Assign a customer to a vehicle: For a customer, that is not assigned to a vehicle, randomly select a vehicle that will serve the customer.
Step 2.	Stop criterion: If there are no more customers left that are not assigned to a vehicle, go to Step 3; otherwise go to Step 1.
Step 3.	Chromosome generation: Return the chromosome containing all customers and their assigned vehicles.

Secondly, a chromosome can be generated using the sweep heuristic algorithm based on the sweep algorithm of Gillett and Miller (1974). Here, all Cartesian coordinates get converted into polar coordinates, their relation is given in Figure 2.6. The customers get sorted by polar angle. This algorithm starts by adding an arbitrary customer and continually adds the next customer as long as the vehicle returns within the maximum travelling time. If the maximum

travelling time is reached, a new vehicle is introduced and the next customer is added. An example of the sweep algorithm applied to a sample set of parcels is illustrated in Figure 2.7. The customers are sorted by angle, indicated by their subscript. They are added to the route until the travelling time surpasses the maximum travelling time. The maximum travelling time is surpassed at the moment v_5 is added to the route, therefore v_1, v_2, v_3 and v_4 get assigned to vehicle 1 and a new route is initialised starting at v_5. The next customers get added to the route, until the travelling time surpasses the maximum travelling time. This process continues as long as there are unrouted customers. Note that the sweep algorithm is not completed at the end of the example, since there are unrouted customers left. The steps of the sweep algorithm for generating a chromosome are described as follows:

Algorithm 2: Sweep chromosome generation

Step 0.	Initialisation: Let 0 be the depot. Let N be the set of customers.
Step 2.	Sort customers:
	Step 2.1. Convert each customer's Cartesian coordinates into polar coordinates.
	Step 2.2. Sort the customers by polar angle.
	Step 2.3 Randomly select one customer as the first.
Step 3.	Route initialisation: Start by creating a route from the depot to the first customer, and back to the depot.
Step 4.	Assign a customer to the route:
	Step 4.1. Insert the next customer of the sorted list at the end of the route, before returning to the depot.
	Step 4.2. Calculate the returning time.
Step 5.	Stop criteria: If the returning time surpasses the maximum travelling time, remove the customer from the route, save the route, and go to Step 3; if there are no more unrouted customers, go to Step 6; otherwise remove the customer from the sorted list and go to Step 4.
Step 6.	Chromosome generation: Return the chromosome containing all customers and their assigned vehicles.

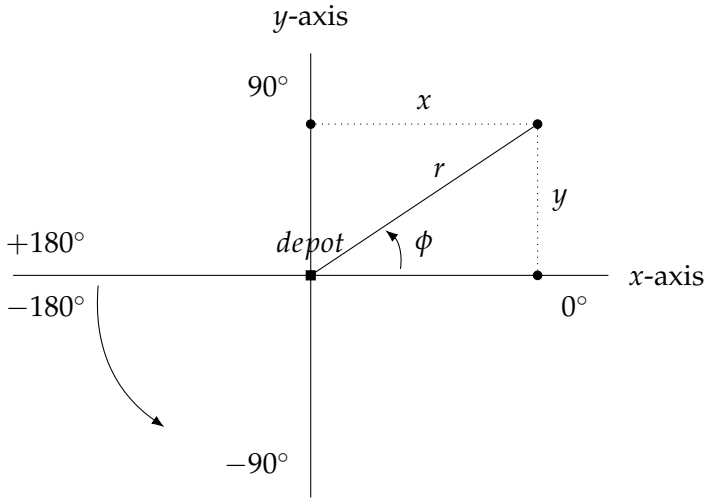

FIGURE 2.6: Visual representation of the relation between Cartesian and polar coordinates. Cartesian coordinates are presented as (x, y), the horizontal and vertical distance with respect to the depot. Polar coordinates are presented as (ϕ, r), the angle and distance with respect to the depot. Their relation is given as: $\phi = \tan^{-1}(y/x)$, $r = \sqrt{x^2 + y^2}$.

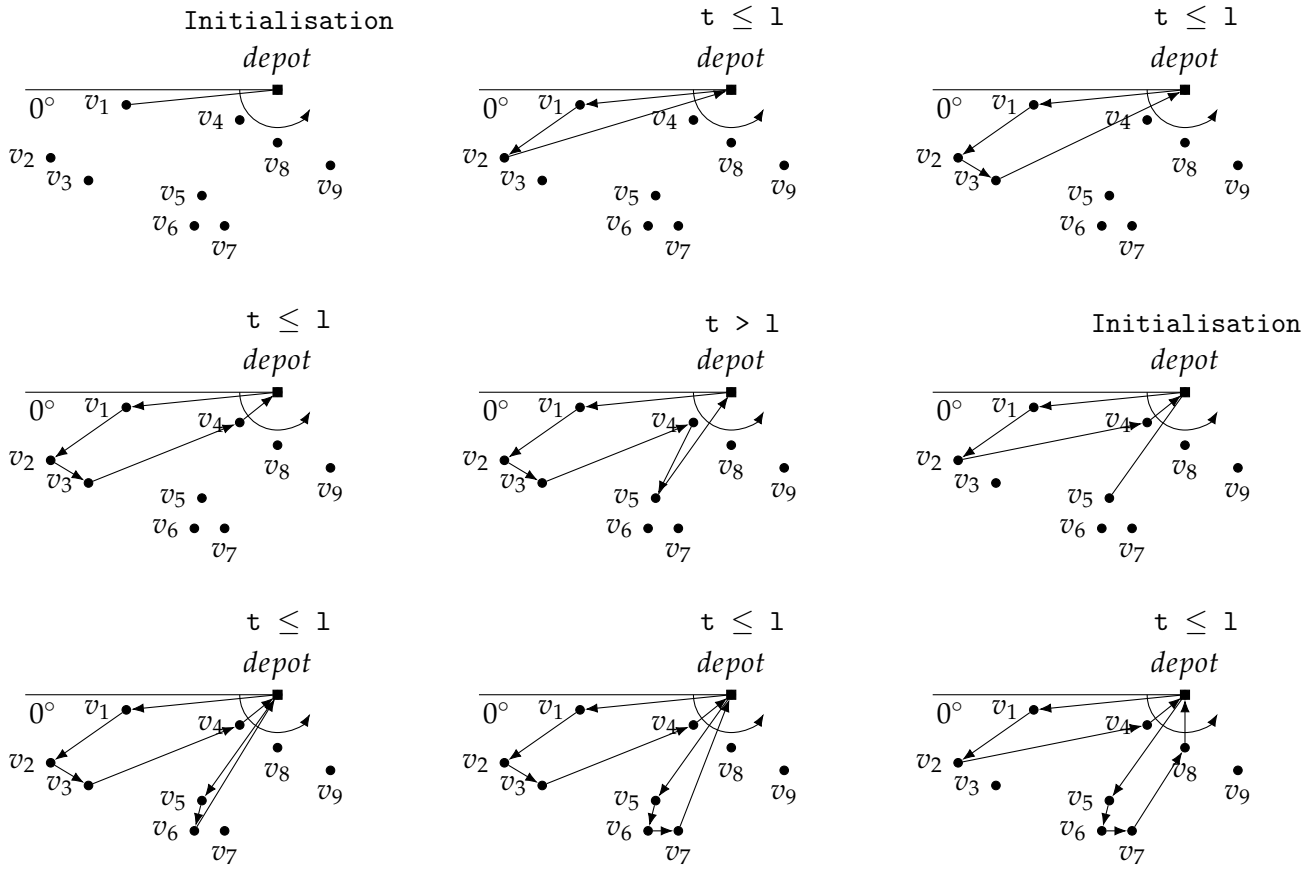

FIGURE 2.7: The sweep algorithm applied to a sample set of parcels. Whenever the travelling time surpasses the maximum travelling time, a new vehicle is introduced.

Thirdly, a chromosome can be generated using the sweep heuristic algorithm combined with split areas. Routes are generated in the same manner as using the sweep algorithm, the difference is that customers are not solely sorted using the size of the angle, ϕ, but different sorted groups are generated depending on their radius, r, as well. Routes get generated for each area separately, where the areas are defined using ranges of radius values. An example of the sweep algorithm with split areas applied to a sample set of parcels is illustrated in Figure 2.8. The distinct areas are indicated by dashed lines in the figures. The customers are sorted by angle, indicated by their subscript, within an area, indicated by their superscript. The customers are added to the route until the travelling time surpasses the maximum travelling time or until there are no unrouted customers in that specific area. Therefore v_1^I, v_2^I and v_3^I get assigned to the same vehicle. The next vehicle is initialised since there are no unrouted customers left in the inner area. The new vehicle visits only v_1^{II}, since visiting v_1^{II} and v_2^{II} surpasses the maximum travelling time. After that, v_2^{II}, v_3^{II} and v_4^{II} are assigned to a new vehicle. Now that there are no unrouted customers in the second area, a new vehicle is initialised in the third area. v_1^{III} and v_2^{III} are assigned to the same vehicle. The sweep algorithm works perfectly as long as the order in the sorted list of customers gives an approximation of nearest neighbours. If areas are densely populated, the sorted list is not an accurate representation anymore, this can be seen in Figure 2.9. The vehicle does not execute an efficient route and consequently the maximum travelling time is surpassed after visiting few customers. A differently generated route could visit more customers. To adjust the sorted list such that the customers are added to the vehicle in a convenient order, is the list updated after adding a customer to the route. The head of the sorted list is defined by a present number of customers. After adding a customer

to the route, the head of the list is reordered such that the next customer is closest to the last added customer. By applying reordering, the list is not solely based on angle anymore. This way the routes are generated in a convenient way and more customers are assigned to a single vehicle, consequently less vehicles are required. The steps of the sweep algorithm combined with split areas for generating a chromosome are described as follows:

Algorithm 3: Sweep combined with split areas chromosome generation

Step 0. Initialisation:
Let 0 be the depot.
Let N be the set of customers.

Step 2. Order customers:
- Step 2.1. Convert each customer's Cartesian coordinates into polar coordinates.
- Step 2.2. Sort the customers by polar angle, within a list specified to a distinct area.
- Step 2.3 Randomly select one as the first for each area.

Step 3. Route initialisation:
- Step 3.1. Start by selecting an area, corresponding to a radius range, containing customers.
- Step 3.2. Create a route from the depot to the first customer of the set, and back to the depot.

Step 4. Assign a customer to the route:
- Step 4.1. Reorder the head of the list such that the next customer is closest to the last added customer in the route.
- Step 4.2. Insert the next customer at the end of the route, before returning to the depot.
- Step 4.3. Calculate the returning time.

Step 5. Stop criteria:
If the returning time surpasses the maximum travelling time, remove the customer from the route, save the route, and go to Step 3.2;
if there are no more unrouted customers left, go to Step 6;
if there are no more unrouted customers left in the area, go to Step 3.1;
otherwise remove the customer from the sorted list and go to Step 4.

Step 6. Chromosome generation:
Return the chromosome containing all customers and their assigned vehicles.

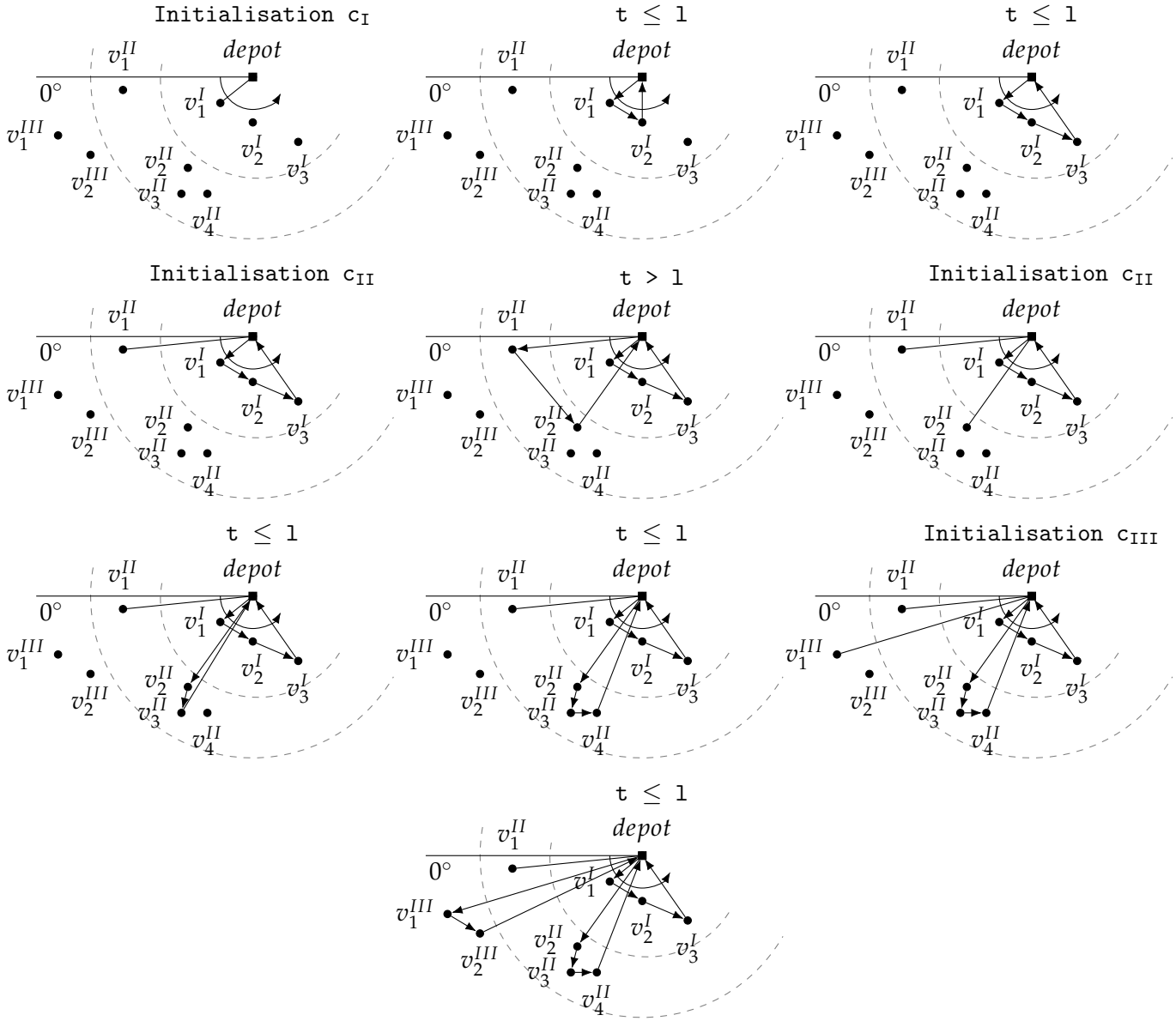

FIGURE 2.8: The sweep algorithm combined with split areas applied to a sample set of parcels. The borders of the areas are specified by the dashed lines. Whenever the travelling time surpasses the maximum travelling time, or whenever there are no more customers in an area, a new vehicle is introduced.

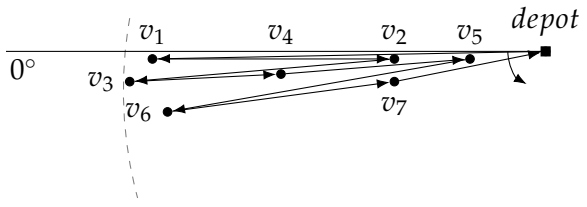

FIGURE 2.9: The sweep algorithm combined with split areas applied to a sample set of parcels. The border of the inner area is specified by the dashed line. Shown is that whenever there are many customers with the same, or almost the same angle, ordering according to angle is not representative as a list of nearest neighbours.

Note that chromosomes generated using the sweep heuristic algorithm combined with split areas have no clusters with parcels from different radius ranges. All clusters are generated within a distinct area, therefore the radius ranges can be seen as borders.

For each of these chromosomes, their performance is determined by using the fitness function, this is done by solving the travelling salesman sub-problems. After this, a new generation of chromosomes will be generated through evolution. This evolution contains processes such as reproduction and mutation, just like evolution in nature. The chromosomes used for reproduction are selected in a similar way to 'survival of the fittest', the chromosomes with the highest fitness will be used to reproduce. This process is continued for a preset number of generations.

Whenever the classical representation of chromosomes is used, a well known reproduction technique is two-point crossover. Two-point crossover on classically represented chromosomes is illustrated in Figure 2.10. In the figure, parent 1 represents a single vehicle serving customers 1, 3, 4 and 8. Parent 2 represents a vehicle serving customers 2, 5 and 9. Customers 6 and 7 are served by another vehicle, which is not shown. Two-point crossover swaps a randomly selected middle part of the parents to create two children. The idea is that good combinations of customers get passed down to the next generation, which does indeed make sense since parts of clusters stay together when applying this reproduction technique.

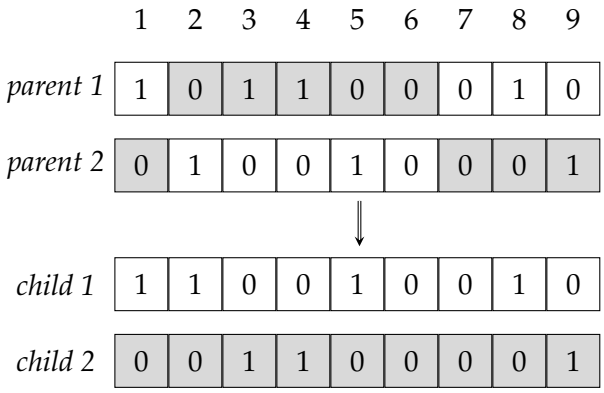

FIGURE 2.10: Two-point crossover on classically represented chromosomes. Parent vehicle 1 serves customers 1, 3 and 4. Parent vehicle 2 serves customers 2, 5 and 9. After applying two-point crossover two new vehicles are created. Child vehicle 1 serves customers 1, 2, 5 and 8. Child vehicle 2 serves customers 3, 4 and 9.

Applying two-point crossover on alternatively represented chromosomes does not make sense. This is illustrated in Figure 2.11, where it can be seen that both parents have customers 1, 3 and 6, and customers 4 and 7 grouped together, but after reproduction, the children do not have customers 1, 3 and 6 grouped together. Some stay together, like customers 4 and 7, but this depends on luck, on whether the vehicles had the same index by coincidence. By applying two-point crossover on alternatively represented chromosomes, good combinations do only get passed down to the next generation by accident. Thus, two-point crossover in combination with alternatively represented chromosomes is inefficient.

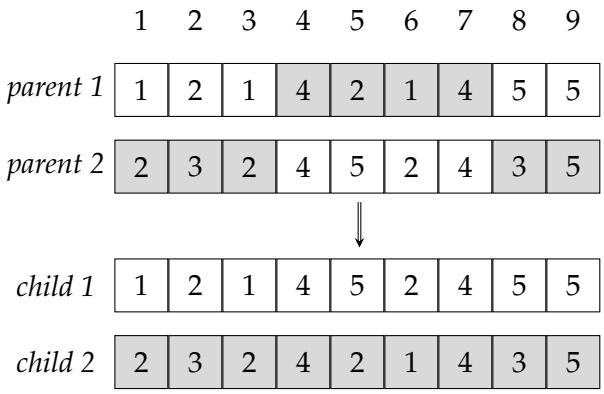

FIGURE 2.11: Two-point crossover on alternatively represented chromosomes. Both parents have customers 1, 3 and 6, and customers 4 and 7 grouped together, but after reproduction, the children do not have customers 1, 3 and 6 grouped together. Some stay together like customers 4 and 7 but this depends on luck, on weather or not the vehicles had the same index by coincidence.

To efficiently perform two-point crossover on alternatively presented chromosomes, the indices of vehicles serving grouped customers should correspond. To match vehicles, the parents get ordered. To order vehicles, the following approach is applied: the centre of each vehicle is generated, it can be seen as the mean x and y coordinates of the served customers. The Cartesian coordinates of the centres are converted to polar coordinates, their relation is given in Figure 2.6. The polar coordinates get ordered by angle and the routes get assigned new vehicle indices, this is illustrated in Figure 2.12. Henceforth, the indices get aligned, the first vehicle is checked to be closest to the first or second vehicle of the other parent. In case the second vehicle is closest, the vehicle indices are increased by one.

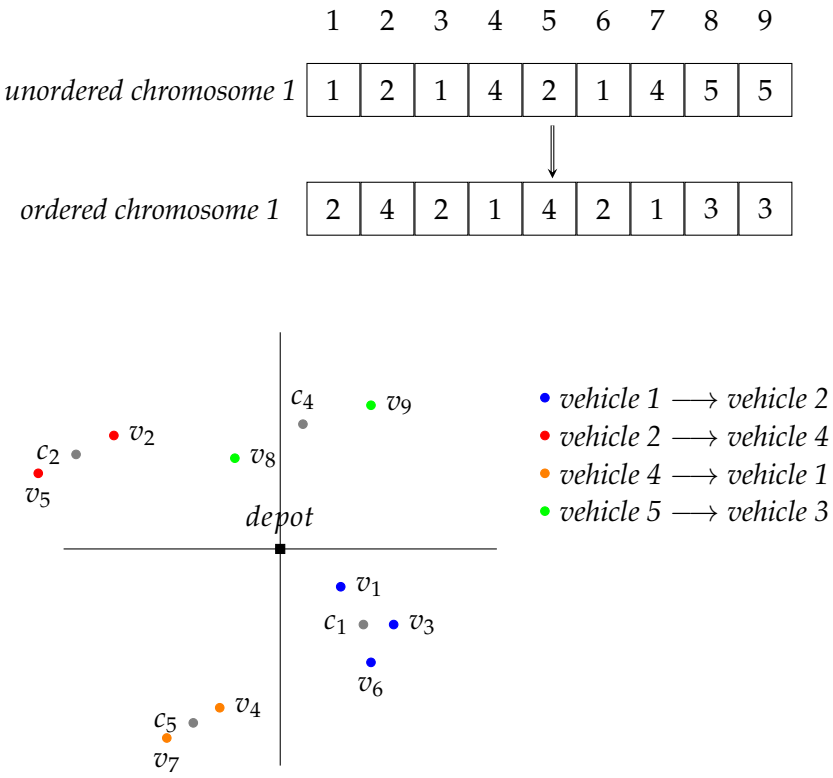

FIGURE 2.12: The ordering of the vehicles in a chromosome according to the angle of the polar coordinates of the centres. Centres $c_1, \ldots, c_4$ are assigned by averaging the x and y coordinates of customers $v_1, \ldots, v_9$ served by a single vehicle.

Once all chromosomes with the highest fitness have reproduced, some of the children of the new generation are copied and mutations occur. In a mutation, a customer is moved to either the next or previous vehicle. Note that due to the fact that these chromosomes are reproductions of ordered chromosomes, the next and previous indexed vehicles lie close to the mutated customer. This way one set of parents might create multiple children, two through reproduction, and a maximum of one mutated copy of each child. After the properties of parents with a high fitness have been transmitted to the children, a new generation is created. Past generations and current generation are combined in a large pool. A new iteration of the algorithm starts with the new generation. Below you find a detailed description of each step of the algorithm to solve the main clustering problem:

Algorithm 4: Main clustering

Step 0. Initialisation:
Let H be the total number of chromosomes in the initial generation.
Let G be the maximum number of evolutionary generations.
Let P be the number of parents.
Let p_m be the mutation rate.
Set a generation counter $g \leftarrow 1$.

Step 1. Initial generation:
- Step 1.1. Generate a chromosome and add it to the generation.
- Step 1.2. If the total number of chromosomes in the initial generation equals H then go to Step 2;
  otherwise go back to Step 1.1.

Step 2. Fitness evaluation:
- Step 2.1. For each chromosome in the generation, solve its corresponding set of sub-problems, and obtain its fitness value $\sum_{k \in K} z_k$.
- Step 2.2. Add all chromosomes in the generation to the pool.
- Step 2.3. Find the maximum fitness value, F^*, of the pool.

Step 3. Stop criterion:
If $g = G$ then stop;
otherwise go to Step 4.

Step 4. New chromosome generation $g \leftarrow g + 1$:
- Step 4.1. Select, from the pool, P parents by tournament selection.
- Step 4.2. Reproduction:
  Pick two parent chromosomes.
  Generate two child chromosomes and add to the new generation.
- Step 4.3. Mutation:
  Randomly duplicate some children of the new generation.
  Apply mutation to these duplicates.

Step 5. Go to Step 2.

In Step 2 of Algorithm 4, the corresponding sub-problems are solved to obtain the fitness value of each chromosome in the generation. Each sub-problem corresponds to an NP-complete problem as is shown in Garey and Johnson (1979), therefore a heuristic is used to solve the sub-problems. First two TSP heuristics will be discussed, one more greedy than the other, therefore finding solutions of lower quality, whilst having a lower computation time. Since neither one of the TSP heuristics proposed meets both of the set criteria regarding solution quality and computation time, a minimum spanning tree (MST) heuristic is introduced, generating solutions of quality similar to the higher quality TSP heuristic and computation time equal to the faster TSP heuristic.

The first algorithm starts by adding the customer closest to the depot to the route. The algorithm continuously adds the customer closest to the customer that was added last. Once all customers have been assigned to the route it returns to the depot. This TSP heuristic is called a nearest neighbour heuristic and runs in quadratic time. The steps of the TSP nearest neighbour heuristic algorithm for solving the kth sub-problem are described as follows:

Algorithm 5: TSP heuristic: nearest neighbour

Step 0.	Initialisation: Let 0 be the depot. Let N_k be the set of customers to be served by vehicle k.
Step 1.	Initial route: Start by creating a route from the depot to the closest customer in N_k.
Step 3.	Assign a customer to the route: Add the customer closest to the customer that was added last at the end of the route.
Step 4.	Stop criterion: If there are no more customers left which are not assigned to the route, go to Step 5; otherwise go to Step 3.
Step 5.	Fitness value: Complete the route by returning to the depot. If the total travelling time is lower than the maximum travelling time, return the total travelling distance as the fitness of the kth sub-problem; otherwise return the total travellin distance doubled as the fitness of the kth sub-problem.

The second algorithm starts by adding the customer furthest from the depot to the route. It then continuously inserts a customer at a place in the route with the lowest total distance increase. Meaning that it checks each insertion place for each customer at every iteration. This TSP heuristic is called an insertion heuristic and runs in cubic time. The steps of the TSP insertion heuristic algorithm for solving the kth sub-problem are described as follows:

Algorithm 6: TSP heuristic: insertion

Step 0.	Initialisation: Let 0 be the depot. Let N_k be the set of customers to be served by vehicle k.
Step 1.	Initial route: Start by creating a route from the depot to the furthest customer in N_k, and back to the depot.
Step 3.	Assign a customer to the route: Step 3.1. Find for each customer, which is not assigned to the route, the insertion place with the lowest increase in the total travelling distance. Step 3.2. Insert the customer with the lowest increase in the route.
Step 4.	Stop criterion: If there are no more customers left which are not assigned to the route, go to Step 5; otherwise go to Step 3.
Step 5.	Fitness value: If the total travelled time is lower than the maximum travelling time, return the total travelling distance as the fitness of the kth sub-problem; otherwise return the total travelling distance doubled as the fitness of the kth sub-problem.

The time complexity of Algorithm 5 is lower than that of Algorithm 6, but Algorithm 6 will result in more efficient routes. The travelled distances from and to the depot are not considered when calculating the cost of a single vehicle, because the cost should reflect how well clustered the customers assigned to a single vehicle are. The cost is thus solely based on the part of the route connecting the customers. Note that the highest fitness would be achieved if every vehicle served a single customer. However, the number of vehicles can never equal the number of customers, since the number of vehicles is upper bounded by the highest number of vehicles assigned to a chromosome in the initial generation.

To calculate the fitness of a vehicle, clustering behavior should be rewarded. Since the total distance of a minimum spanning tree, without considering the depot, does exactly that, it is used to base the fitness value. Generating a minimum spanning tree using the algorithm of Prim can be done in quadratic time, therefore it is as complex as Algorithm 5, whereas the results will be as representative as those of Algorithm 6. The steps of the MST heuristic algorithm

of Prim for solving the kth sub-problem are described as follows:

Algorithm 7: MST heuristic: Prim

Step 0.	Initialisation: Let N_k be the set of customers to be served by vehicle k.
Step 1.	Initial route: Start by creating a set of visited customers, V and randomly add a customer, v, from N_k. Keep track of the not visited customers, U.
Step 3.	Visit the customer, $u \in U$ closest to the visited customers, V.
Step 4.	Stop criterion: If there are no more customers left which are not visited, go to Step 5; otherwise go to Step 3.
Step 5.	Fitness value: Return the total distance as the fitness of the kth sub-problem.

Note that in Step 3 of Algorithm 7, u is visited; therefore, u is removed from U and added to V, furthermore, the fitness value returned in Step 5 of Algorithm 7, is solely based on the total distance of the MST. No actual route is generated, therefore no penalty can be applied for surpassing the maximum travelling duration. Algorithm 7 runs in quadratic time in case the distance of each not yet visited customer to the visited customers, is recorded and updated after a new customer is visited. This way not all distances to the visited customers have to be checked in each iteration.

After the decomposition method has run for the preset number of generations, the chromosome with the highest fitness is returned. The chromosome provided is a clustering, therefore the actual routes of the vehicles still have to be determined. This is done using Algorithm 6, continuously inserting a customer at a place in the route which has the lowest total distance increase. Algorithm 6 is used since it is the construction of the final routes and therefore performance outweighs computation time.

2.2.3 Local search

At this moment the dummy parcels are routed using either the VRP heuristic or the decomposition method. Local search methods can be applied to improve the routes, where intraroute and interroute methods are distinguished. Since at this point the customers considered are dummy customers, customers can be re-positioned within a tour, as well as interchanged between two different tours. Before applying local search methods the service times belonging to dummy customers are removed, for two reasons; these should not prevent a vehicle from returning within the maximum travelling time after local search methods improved the total travelling distance, and leeway is provided for the insertion of parcels in the dynamic assignment. Note that due to the first reason there is a small preference of removing the service time before compared to after applying local search methods and that a constant time of one minute per customer is removed from the route.

The intraroute method used is called 2-opt neighbourhood search. This method is a 2-exchange method for travelling salesman problems, it destroys two randomly selected arcs and replaces them by connecting different customers. The steps of the local search 2-exchange method are described as follows:

Algorithm 8: Local search: 2-exchange method

Step 0. Initialisation:
Let $l \to \ldots \to i \to j \to \ldots \to k \to l$ be a tour driven by a single vehicle.

Step 1. Select two arcs in the tour: $i \to j$ and $k \to l$.

Step 2. Remove these arcs from the tour, resulting in two paths:

$$P_1 = l \to \ldots \to i \quad \text{and} \quad P_2 = j \to \ldots \to k.$$

Step 3. Connect the paths to form a new tour:

$$l \to \ldots \to i \to k \to \ldots \to j \to l.$$

Figure 2.13 illustrates a deletion of two randomly selected arcs in the first tour. Separating the two paths $P_1 = v_6 \to v_7 \to v_8 \to v_9 \to v_{10} \to depot \to v_1 \to v_2$ and $P_2 = v_3 \to v_4 \to v_5$. The two paths are reconnected by arcs $v_2 \to v_5$ and $v_3 \to v_6$, resulting in the new tour. This new tour obviously has a lower total distance, thus an improvement is found. Note that the direction of P_2 has been reversed. Multiple combinations of arcs are checked, and every time an improvement is found, meaning that the total travelling distance is lowered, the 2-exchange is performed and the tour is updated.

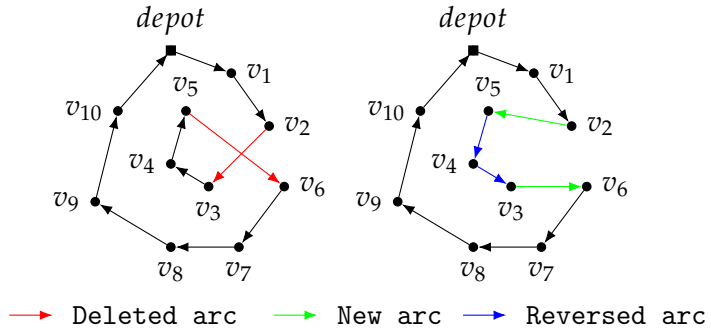

FIGURE 2.13: An example of a tour on which a 2-exchange is performed and results in an improvement.

No interroute methods will be used, even though at this stage this would still be possible since the scheduled parcels considered are dummy parcels. The methods used to generate the routes value the clustering ability. Simple interroute local search methods might improve the routes, but are solely based on decreasing the total distance. Some interroute local search method exchange parts of routes. These methods do not consider the locations of the exchanged customers with respect to the customers in the other tour. This way the carefully constructed clusters of customers are spread out.

2.3 Dynamic assignment

In the previous step the initial solution has been generated. Meaning that a vehicle routing problem is solved using the dummy data set generated in the first step. In the final step each dummy parcel is substituted by a dynamic parcel. These dynamic parcels are the real parcels which one by one arrive at the depot. Up until the dynamic assignment, parcels could change vehicle. From this moment on, this is not possible anymore since the parcels considered are

actual parcels. The method used for the dynamic assignment is described in Los (2019) as well as in Los et al. (2020).

For each incoming parcel u, different tours get evaluated such that the insertion place and substitution with the largest cost decrease or smallest cost increase are found. The cost, C, or benefit, B, is expressed in terms of travelling distance d_{ij} between parcels i and j. The evaluation is done by considering a single vehicle and iterating over all parcels in that route, either dummy, or already placed dynamic, parcels. When iterating over parcel k, the following is checked:

Type 1 insertion: if parcel k, with neighbours i and j, is a dummy parcel:

- Does replacing parcel k result in a feasible route, given the time restriction?
- Calculate the costs for this insertion, given by

$$C(i,u,j) = (d_{iu} + d_{uj}) - (d_{ik} + d_{kj}) \tag{2.28}$$

Type 2 insertion: if parcel k, with neighbours i and j, is an already placed dynamic parcel:

- Does inserting parcel u between parcels i and k result in a feasible route, given the time restriction? Calculate the costs for this insertion, given by,

$$C(i,u,j) = (d_{iu} + d_{uk}) - d_{ik} \tag{2.29}$$

- Does inserting parcel u between parcels k and j result in a feasible route, given the time restriction? Calculate the costs for this insertion, given by,

$$C(i,u,j) = (d_{ku} + d_{uj}) - d_{kj} \tag{2.30}$$

- Determine which dummy parcel m, with (different) neighbours i and j, can be removed with the highest benefit, given by

$$B(i,m,j) = (d_{im} + d_{mj}) - d_{ij} \tag{2.31}$$

The insertion place with the largest cost decrease, or smallest cost increase is chosen. In case of a type 2 insertion, the decision regarding the insertion place is solely based on the change in cost, benefit is not considered yet.

The two different types of insertions will be explained in more detail using examples. Figure 2.14 illustrates a tour consisting of only dummy parcels, v_1 till v_5, in which a dynamic parcel, d_1, is inserted. Each dummy parcel is substituted by the dynamic parcel, and the most profitable substitution is performed. Tour 2 to 6 in Figure 2.14 illustrate the substitutions. The cost of each substitution is calculated using Equation (2.28). The substitution with the largest cost decrease, or smallest cost increase is chosen, which in this case is substituting dummy parcel v_4.

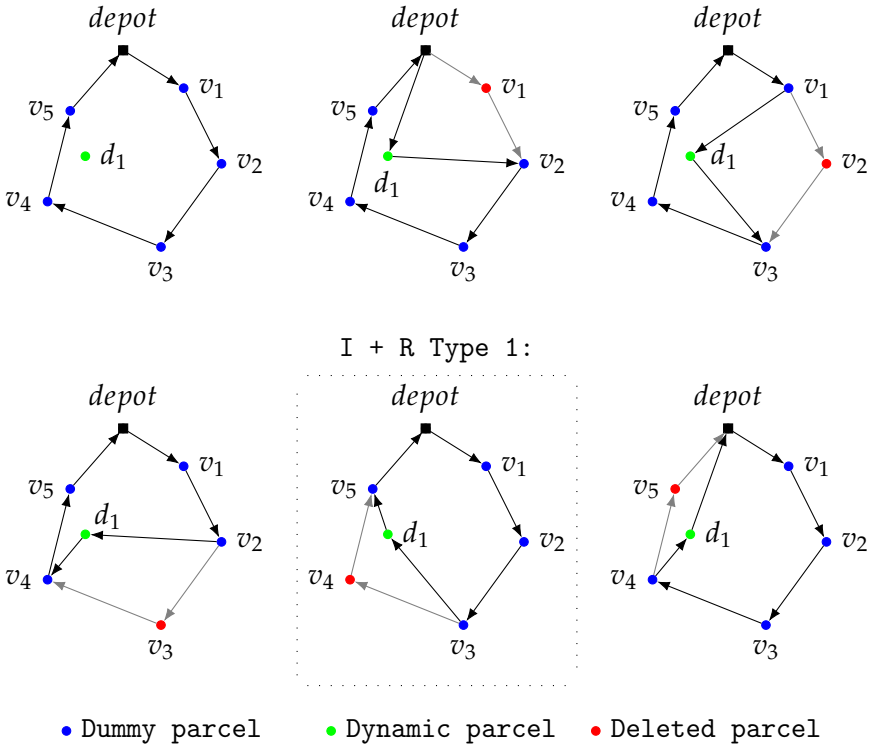

FIGURE 2.14: An example of a tour showing a type 1 insertion and removal. Tour 1 consists of dummy parcels $v_1, \ldots, v_5$, in which dynamic parcel d_1 will be inserted. Tour $2, \ldots, 6$ are the results of a type 1 insertion, each tour showing the substitution of a dummy parcel. Tour 5 shows the insertion with the largest cost decrease and is therefore applied. Dummy parcel v_4 is substituted by dynamic parcel d_1, and thus removed from the route.

The insertion of a second dynamic parcel is considered. The second tour of Figure 2.15 shows the substitution of the first dynamic parcel d_1, and this is our starting point for determining the most profitable insertion for the second dynamic parcel, d_2. The insertion of d_2 is again a type 1 insertion, where dummy parcel v_3 is substituted. The substitution of v_3 by d_2 is illustrated in the third tour in Figure 2.15 and this is our starting point for determining the most profitable insertion of a third dynamic parcel, d_3. The process of inserting dynamic parcel d_3 is illustrated in Figure 2.16.

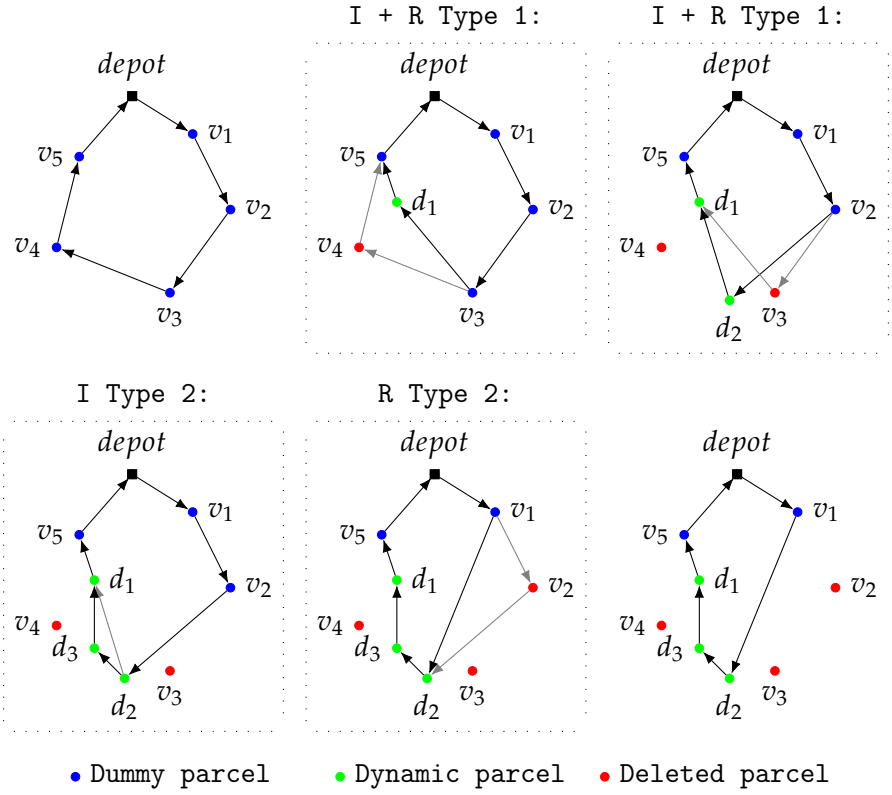

FIGURE 2.15: An example of a tour showing two type 1 insertions and removals and a type 2 insertion and removal.

Figure 2.16 illustrates a tour consisting of three dummy parcels, v_1, v_2 and v_5, and two dynamic parcels d_1 and d_2, in which the dynamic parcel d_3 is inserted. Each possible substitution of a parcel in the tour is considered. The first two parcels considered are v_1 and v_2, both dummy parcels, a type 1 insertion is applied and costs are calculated using Equation (2.28). The third parcel, d_2, cannot be substituted since it is an already placed dynamic parcel. Two different type 2 insertions are considered, before and after d_2, illustrated in the fifth and sixth tour of Figure 2.16. The cost of inserting d_3 before d_2 is calculated using Equation (2.29), and the cost of inserting d_3 after d_2 is calculated using Equation (2.30). Equivalent insertions are performed when iterating over parcel d_1. Last, dummy parcel v_5 is considered. After all costs are calculated using either Equation (2.28), (2.29) or (2.30), the most profitable insertion is performed, which in this case is a type 2 insertion after dynamic parcel d_2. Note that the sixth and eighth tour illustrated in Figure 2.16 are identical. This is a result of dynamic parcels d_1 following d_2, and that for both a possible type 2 insertion before and after the dynamic parcel is considered.

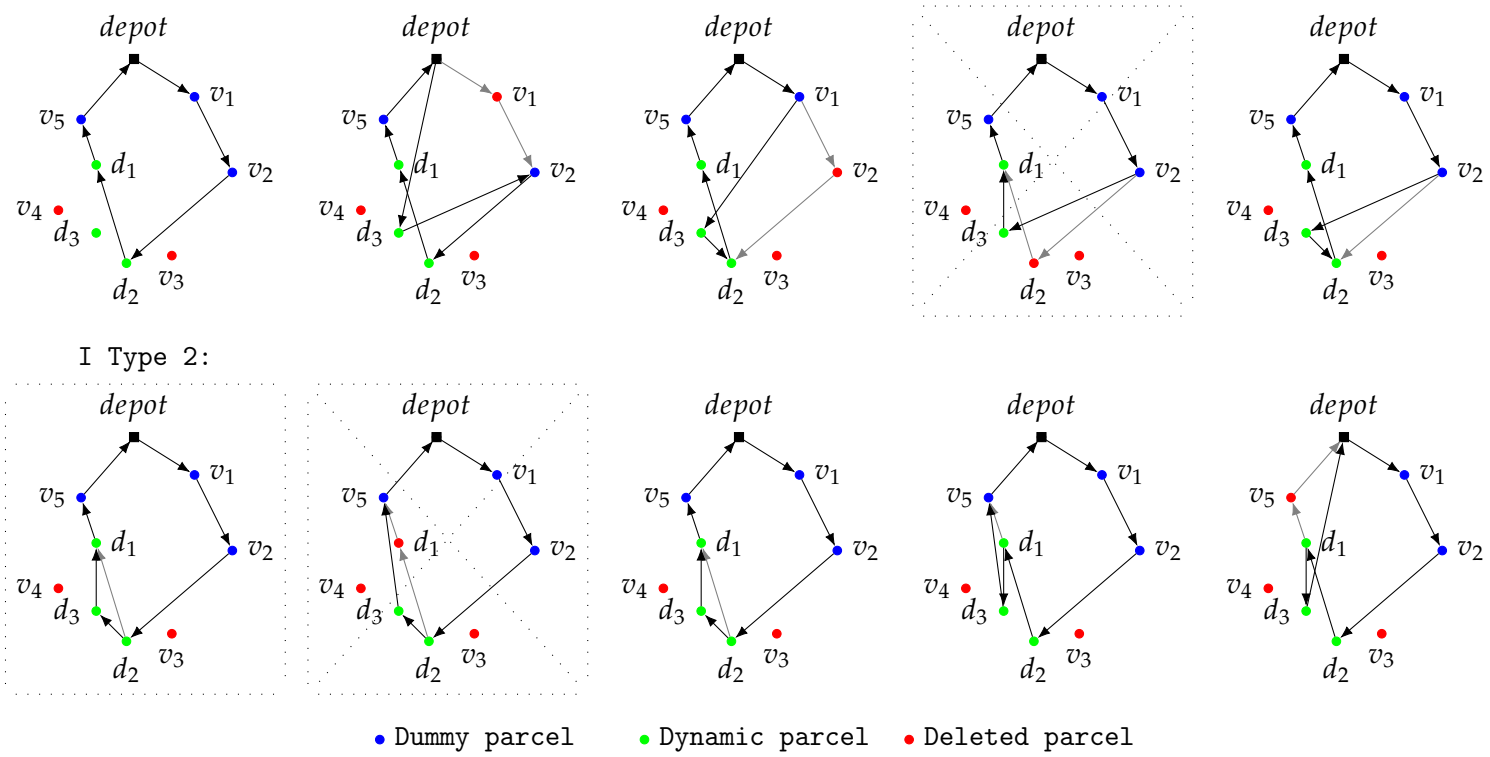

FIGURE 2.16: An example of a tour showing a type 2 insertion.

The type 2 insertion discussed in Figure 2.16 is illustrated in the fourth tour of Figure 2.15. After a type 2 insertion is performed, a dummy parcel still has to be removed from the tour. All dummy parcels are considered and their benefits are calculated using Equation (2.31). The dummy parcel with the highest benefit is removed, resulting in the removal of dummy parcel v_2, illustrated in the fifth tour of Figure 2.15.

During the execution of the algorithm it is computationally profitable to set boundaries to the number of tours investigated for feasible insertion places. The proposed method by Los (2019) and Los et al. (2020) is to only consider the X 'closest' vehicles. Using the distance to the (dynamic and dummy) parcels in the surrounding area a set of the X closest vehicles can be generated. After all dynamic parcels are inserted, the remaining dummy parcels, if any remained, are removed from the routes. This way the vehicles deliver all dynamic parcels without the influence of remaining dummy parcels.

2.3.1 Local search

Once the insertion of dynamic parcels starts, the assignment to a vehicle is final. Even though the assignment to a vehicle is final, the route can still be optimised. Within route optimisation, or intraroute optimisation, is performed such that the total travelling distance is minimised, by doing so, often the total travelling time reduces. The local search methods can be applied throughout the insertion of dynamic parcels, for example after a certain number of arrived dynamic parcels, and at the end once all dynamic parcels are inserted. The intraroute method used after the formulation of the initial solution, Algorithm 8, can be used during as well as after the dynamic insertion.

Chapter 3

Data

Data used to generate and compare results throughout this thesis is provided by DPD through TNO. DPD uses ten depots scattered throughout the Netherlands to cover the 4,069 neighbourhoods (4-digit postal codes) in which the Netherlands is divided. The depots are located in Amsterdam, Eindhoven, Etten-Leur, Joure, Maastricht, Meppel, Rijssen, Rotterdam, Tynaarlo and Veenendaal. Figure 3.1 illustrates the locations of these depots. During this research, the main focus is on the depot of Veenendaal.

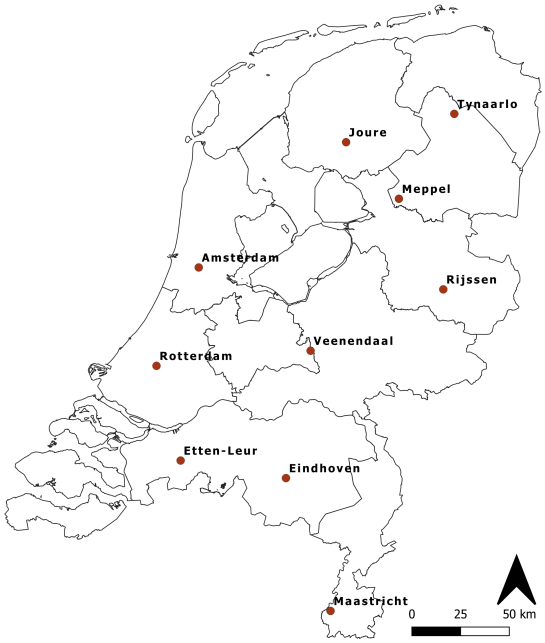

FIGURE 3.1: The locations of the ten depots of DPD displayed on the map.

A list containing all postal codes covered by the depot of Veenendaal is provided. Each postal code is presented as an alphanumerical code consisting of four numbers and two letters of the alphabet referring to a specific area covered by the depot. Geographical coordinates are provided for each of the postal codes. All coordinates are defined as a longitude and latitude combination. As defined by contributors (2021), "longitude is a geographic coordinate that specifies the east-west position of a point on the Earth's surface. It is an angular measurement, usually expressed in decimal degrees. Meridians (lines running from pole to pole) connect points with the same longitude. The prime meridian, which passes near the Royal Observatory, Greenwich, England, is defined as 0° longitude by convention. Positive longitudes are east of the prime meridian, and negative ones are west. Latitude is a geographic coordinate that specifies the north-south position of a point on the Earth's surface. Latitude is an angle which ranges from 0° at the equator to 90° (north or south) at the poles. Lines of constant latitude,

or parallels, run east-west as circles parallel to the equator. Latitude is used together with longitude to specify the precise location of features on the surface of the Earth". Note that locations situated in the Netherlands have positive longitude and latitude. The coordinates of the depot in Veenendaal are gathered using Google Maps, and defined as [52.0458462 N, 5.5633169 E]. To give an approximation of the outlines of the area covered by depot Veenendaal, all postal codes provided are illustrated in Figure 3.2.

FIGURE 3.2: All postal codes covered by the depot of Veenendaal.

The data covers the period of 10/09/2018 untill 22/12/2018, where deliveries are made Monday through Saturday, no parcels are delivered on Sundays. For each day, the data provides an ordered list of parcels (referred to by postal codes) arriving at the depot. Remember that the fact that the list is ordered is an important feature. The last week of December is removed from the data, since it is unclear which data belongs to which day due to changed delivery days surrounding Christmas. Throughout this thesis references are made to week numbers, where week number 37 starts on Monday 10/09/2018, and week 51 ends on Sunday 23/12/2018, (with the last delivery of that week on Saturday 22/12/2018). In Los (2019), who pre-processed this data, it is explained that all data regarding pickups and data showing irregularities is removed. The data is reduced from 18,026,183 observations to 11,559,615 observations after filtering.

It is investigated if there are trends or patterns in the data which make the usage of historical data valuable. First of all, the number of deliveries on different days is examined. The exact numbers of delivered parcels on each day can be found in Table A.1. Visually comparing the number of deliveries in a table is hard, therefore the number of delivered parcels is depicted in Figure 3.3.

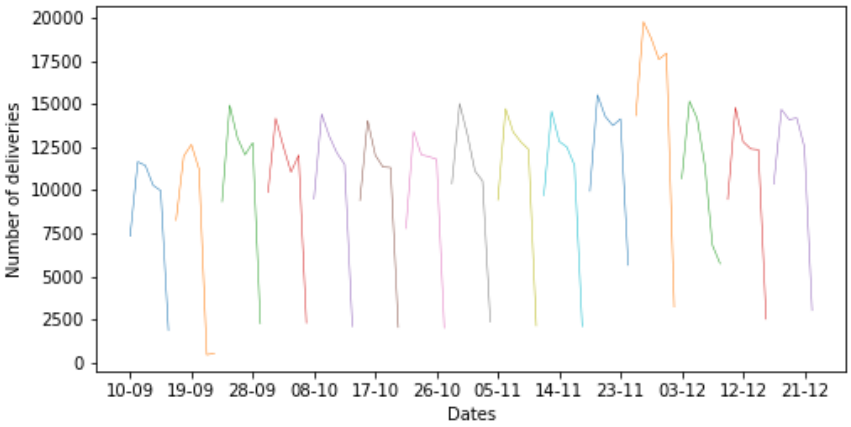

FIGURE 3.3: The number of deliveries per day over for the period of 10/09/2018 till 22/12/2018.

A clear reoccurring pattern is seen in Figure 3.3. The number of deliveries has a weekly pattern, therefore the data of each week is depicted separately in Figure 3.4.

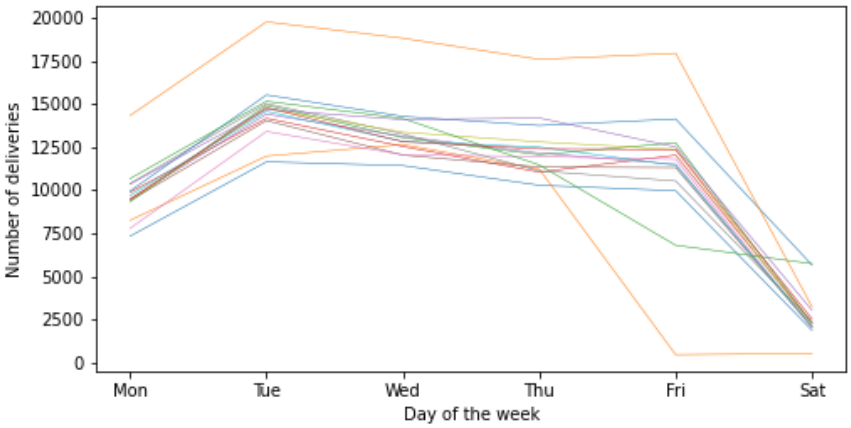

FIGURE 3.4: The number of deliveries per day over for the period of 10/09/2018 till 23/12/2018, where each week is depicted separately.

It is concluded that the number of parcels delivered on a certain day, say Monday, is comparable to the number of parcels delivered on past Mondays. Notice that week 48 has significantly more deliveries compared to other weeks, probably since it is the week after Black Friday, a day on which many stores offer highly promoted sales and the week before Sinterklaas, the Dutch holiday. The Friday and Saturday of week 38 had little deliveries, which has no seemingly clear cause.

It is now known how to derive an estimate of the number of parcels on a certain day. Next, predictions are made regarding the expected placements of the parcels to certain postal codes. To do so, three dates (16/10/2018, 23/10/2018 and 30/10/2018) are used to check repeated deliveries to customers. These dates correspond to the three Tuesdays preceding the first Tuesday in November, later used as data to generate results. A spatial representation is shown in Figure 3.5.

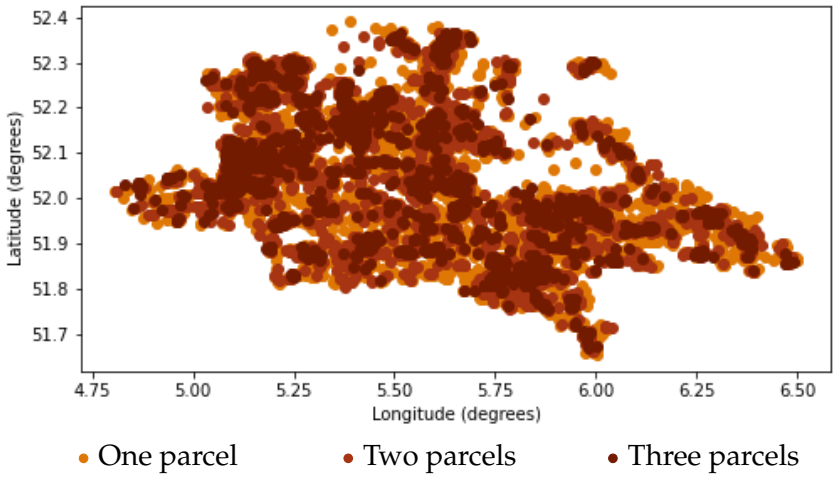

FIGURE 3.5: Frequency of parcel deliveries at postal codes on 16/10/2018, 23/10/2018 and 30/10/2018.

Regarding the colors of Figure 3.5, it is clear that there is a repetitiveness in the delivery of parcels in the historical data, but by looking at this spatial representation the exact pattern is not visible. To provide a more extensive overview Tables A.2, A.3, A.4 and A.5 have been generated. Each table states the percentage of customers on each day, who had a parcel delivered a week earlier, in the past three weeks, in the past five weeks, and in the past ten weeks on the same day of the week. The averages have been summarised in Table 3.1.

TABLE 3.1: Average percentage of customers who had a parcel delivered in the past weeks on the same day of the week.

Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Historical data size	Past week	28.94	31.21	31.66	31.13	29.40	10.20
	Past three weeks	49.12	55.73	54.42	53.01	51.39	21.89
	Past five weeks	59.85	67.87	65.75	64.20	62.31	29.26
	Past ten weeks	74.27	82.75	80.27	78.81	77.35	41.95

The average percentage for each day increases as more weeks are considered. When considering more weeks, the number of parcels considered increases, therefore it is not unexpected that the percentage of customers who had a parcel delivered in the previous weeks gets higher. Note that parcels do not get taken into consideration multiple times. If a single customer received more than one parcel in the past, this does not influence the response, it is only considered once. The same holds for customers, if a customer (a postal code) receives multiple parcels on a single day, it is only considered once. Therefore a higher percentage truely does reflect the repetitiveness of the customers on a certain day, and not the repetitiveness of a driver visiting the same postal code multiple times on a single day. A trade-off has to be made for the size of the historical data set and the corresponding percentage of parcels delivered in the past. Larger sets will significantly increase the computation time of generating dummy data.

Previously discussed methods combined with the knowledge of structures in the data, can now be used to generate results. To be able to elaborate on the results found, they are compared to those presented in Los (2019), results of DPD as well as among themselves.

Chapter 4

Computational results

This section discusses the results generated. As mentioned in Section 1.2, the main goal is to minimise the total travelling distance, whilst satisfying the maximum travelling time constraint. The subgoal is to reduce the number of vehicles used. When stating the results, additional information will be presented. Particular features of the routes, such as the number of vehicles, the total travelling distance and the average travelling time, will be given. The computational running times of the algorithms will be discussed, since there are different time requirements per step of the approach. All results presented in this thesis correspond to the first ten days of November on which deliveries occurred (01/11 to 03/11, 05/11 to 10/11 and 12/11).

Implementations are done in Python 3.8 and experiments are executed using a Intel(R) Core(TM) i5-7300U 2.60GHz processors with 2 cores and 8.0 GB RAM.

Section 4.1 discusses how historical data is used to generate the dummy data. Section 4.2 introduces the common parameter settings for the second step of the approach. Case specific parameter settings and results for the vehicle routing problem heuristic and the decomposition heuristic are discussed in Sections 4.2.1 and 4.2.2, respectively. In Section 4.2.3, both methods are compared to each other and to the results presented in Los (2019). In Section 4.3, the parameter settings and results of the third step of the approach, the dynamic assignment, are discussed. The results of both approaches are compared to the benchmark results in Section 4.4.

4.1 Dummy data

Historical data is used to generate a set of dummy parcels. From the summarised results in Table 3.1, a trade-off has to be made for the size of the considered historical data set and the corresponding percentage of parcels delivered in the past. It is chosen to consider the data of the past three weeks corresponding to the same day of the week. In the weighted K-means clustering, used to generate the dummy parcels, each historical parcel is assigned a weight. As given in Equation (2.2), weight is assigned according to the number of parcels delivered on that specific day. The number of clusters, and therefore dummy parcels, generated corresponds to the average number of parcels delivered in the past three weeks on the same day of the week.

Results regarding the generation of dummy parcels can be found in Table 4.1. The table presents the number of dummy parcels generated, the actual number of parcels (which will be considered in the third step of the approach) and the computation time needed to generate the dummy parcels.

TABLE 4.1: Historical data considered for, and results of the generation of dummy parcels.

Day		01/11	02/11	03/11	05/11	06/11	07/11	08/11	09/11	10/11	12/11
Historical data (No. of weeks ago)	3	12,161	11,508	2,080	9,409	14,046	12,041	11,348	11,316	2,033	7,783
	2	11,348	11,316	2,033	7,783	13,415	12,062	11,951	11,802	2,002	10,386
	1	11,951	11,802	2,002	10,386	15,038	13,269	11,124	10,539	2,348	9,436
No. of dummy parcels generated		11,820	11,542	2,039	9,193	14,167	12,458	11,475	11,219	2,128	9,202
Actual No. of parcels		11,124	10,539	2,348	9,436	14,737	13,375	12,808	12,373	2,124	9,693
Computation time (m:s)		22:19	20:40	00:24	11:41	37:06	25:10	20:02	18:39	00:25	11:00

The number of dummy parcels is supposed to approximate the number of actual parcels, for all instances considered it does so properly. Time needed to generate the dummy parcels is 20 to 30 minutes for most days considered, although a day with considerably more parcels can take up to 40 minutes. The computation time of this step is not of the utmost importance. All information required to compute the set of dummy parcels is available 6 days in advance.

4.2 Initial solution

The VRP heuristic and the decomposition method require the knowledge of each pairwise travelling distance and duration of dummy parcels to generate a vehicle routing problem solution. Generating a distance and duration matrix once and using this for the VRP heuristic and decomposition method is faster than continuous calculations. Generating the distance and duration matrix is done using the open source routing machine, OSRM, of Luxen and Vetter (2011). When given a maximum of a 1000 locations the complete distance and duration matrix are returned within approximately 60 seconds. To generate the distance and duration matrix for larger sets, the locations are divided in groups of 500 where each combination of two groups is generated once. Using an innovative way to fill a matrix, the complete distance and duration matrix can be generated in acceptable computation time, taking the size of the problem into account. This way it is possible to base the solutions generated in this thesis on actual travelling distance and duration (instead of an approximation using the Manhattan distance as is done in Los (2019)).

Results regarding the generation of the distance and duration matrix can be found in Table 4.2. The table presents the number of dummy parcels, the number of OSRM calls and the computation time.

TABLE 4.2: The number of OSRM calls and computation time needed to generate the distance and duration matrix.

Day	01/11	02/11	03/11	05/11	06/11	07/11	08/11	09/11	10/11	12/11
No. of dummy parcels	11,820	11,542	2,039	9,193	14,167	12,458	11,475	11,219	2,128	9,202
No. of OSRM calls	276	276	10	171	406	300	253	253	10	171
Computation time (h:m)	04:06	04:05	00:06	02:31	05:52	04:20	03:32	03:26	00:07	02:17

The computation time needed to calculate the distance and duration matrix is acceptable. Note that the number of OSRM calls grows quadratically with the input size. Calculating distance and duration matrices for days with many expected deliveries (more than 15000 for

example) results in computation times of six hours or more. All information required to compute the set of dummy parcels is available 6 days in advance.

The only parameter defined in the same manner for both solution methods is the service time. The service time is defined as one minute per parcel; in my opinion this seems too little, but this makes the results comparable to the benchmark solutions. Since the solutions of this step are generated using either the VRP heuristic or the decomposition method, both will be discussed separately in Section 4.2.1 and 4.2.2, respectively.

4.2.1 Vehicle routing problem heuristic

Section 2.2.1 discusses two approaches regarding the definition of insertion criteria. The first approach is based on the principle of whether it is more beneficial to serve a customer in the partial route rather than in a direct route. The second approach aims to select customers whose insertion costs minimise a measure of total route distance and time. The way of defining the insertion criteria depends on the preferred relation between serving a customer and its travelling distance and time added to the route when served. The implemented VRP heuristic uses the first approach, since this approach compares the preference of actually assigning a customer by a value based on the assignment to the route as well as a direct trip, whereas the second approach bases its value solely on the assignment to the route. The first approach defines the parameters α_1, α_2, λ and μ. The parcels are preferred to lie geographically close together, therefore distance is weighed more heavily. c_2, defined in Equation (2.14), is based on the principle of whether it is more beneficial to serve a customer in the partial route rather than on a direct route. λ is defined as 1, this way a direct trip is weighed against the added costs of insertion. c_1 contains variables α_1 and α_2 which define the weight distribution of the added distance and time after insertion. The value of c_1 should be able to outweigh a distance variable. To do so, the value of c_1 should be mostly influenced by the added distance and not the duration and therefore $\alpha_1 = 0.8$, so $\alpha_2 = 1 - \alpha_1 = 0.2$. μ is defined as 1, this way the formulation of c_{11}, defined in Equation (2.15), returns the absolute added distance, the same way c_{12}, defined in Equation (2.16), returns the absolute added duration. This results in the following: the smaller the increase in distance and time, mostly distance, the larger the priority for insertion.

After routes have been generated using the VRP heuristic, the local search method, explained in Section 2.2.3, runs a preset number of iterations to optimise the routes. The local search optimises each route separately for a number of iterations equal to the squared number of parcels in that specific route. Results regarding the routes generated by the vehicle routing problem heuristic before and after applying local search can be found in Table 4.3. The table presents, among others, the number of dummy parcels, the number of required vehicles and the total travelling distance and average duration of the routes, before and after local search is applied.

TABLE 4.3: Results regarding the routes generated by the vehicle routing problem heuristic before and after applying local search.

			VRP heuristic			Local search		Computation time	
Day	Date	#DP	#T	TD (km)	AD (h:m)	TD (km)	AD (h:m)	VRP (h:m)	LS (m:s)
Thu	01/11	11,820	63	12,877	8:52	12,624	8:46	21:46	3:19
Fri	02/11	11,542	62	12,900	8:54	12,589	8:46	14:55	3:11
Sat	03/11	2,039	20	5,201	8:33	5,038	8:22	0:03	0:05
Mon	05/11	9,193	52	11,088	8:56	10,804	8:48	2:13	2:06
Tue	06/11	14,167	-	-	-	-	-	-	-
Wed	07/11	12,458	-	-	-	-	-	-	-
Thu	08/11	11,475	61	12,640	8:57	12,360	8:50	15:52	3:23
Fri	09/11	11,219	61	12,620	8:51	12,348	8:44	10:27	5:25
Sat	10/11	2,128	20	5,170	8:38	4,983	8:25	0:03	0:05
Mon	12/11	9,202	53	11,291	8:51	11,064	8:44	1:55	1:15

The average duration of routes generated using the VRP heuristic is nearly nine hours, the maximum travelling time of a vehicle. It is concluded that the travelling time of a vehicle is almost optimally utilised. The quadratically growing computation time of the VRP heuristic is acceptable for instance sizes up until 11,000 parcels, for larger instances the computation time becomes infeasibly long. As mentioned before, the computation time of this step is not of the utmost importance. All information required to route the set of dummy parcels is available 6 days in advance, nonetheless results should be generated within acceptable time, surpassing 24 hours is undesirable. No routes are generated corresponding to the dummy parcels of 06/11 and 07/11 using the VRP heuristic due to the expected computation time. The local search method runs quickly, reducing the total travelling distance and the average duration. The computation time of the local search did not surpass six minutes and the total travelling distance was reduced by 2.5% on average.

4.2.2 Decomposition method

All parameters used in the decomposition method are either chosen to be scaled according to the number of dummy parcels or fixed. Three ways were introduced to generate the chromosomes belonging to the initial generation. Firstly, at random, as explained in Algorithm 1. Secondly, using the sweep algorithm, as explained in Algorithm 2. Thirdly, using the sweep algorithm combined with split areas, as explained in Algorithm 3. In the initial generation no chromosomes are generated using Algorithm 1 or 2. All chromosomes in the initial generation are generated using Algorithm 3. The number of chromosomes generated using Algorithm 3 equals 20% of the number of dummy parcels. For example, when considering a set of 10,000 dummy parcels, 2000 chromosomes are generated using the sweep algorithm combined with split areas. The initial generation is chosen to be generated in this manner, since randomly generated chromosomes as well as chromosomes generated using the sweep algorithm never occurred as parents. Their fitness was never comparable to that of the chromosomes generated using the sweep algorithm combined with split areas. When randomly generating chromosomes, the number of vehicles used equalled the maximum number of vehicles used in a chromosome generated using the sweep algorithm.

The areas defined in the sweep algorithm combined with split areas are constructed geographically such that an inner, middle and outer area are defined. The radiuses are set at 0.2 and 0.4 degrees longitude and latitude, meaning; if a parcel's radius is less than 0.2 when its coordinates are converted to polar coordinates, a parcel belongs to the inner circle; parcels with a radius between 0.2 and 0.4, belong to the middle circle; and parcels with a radius larger than

0.4 belong to the outer circle. Figure 4.1 illustrates the boundaries of the areas used. Note that the circles became ellipses on the map. This is a result of the fact that distance corresponding to one degrees longitude is not static, but rather a function of the latitude. At the equator one degree latitude covers the same distance as one degree longitude. In the Netherlands, at a higher latitude, the longitudinal and latitudinal distance are not equal, longitudinal degrees correspond to a shorter distance than latitudinal degrees.

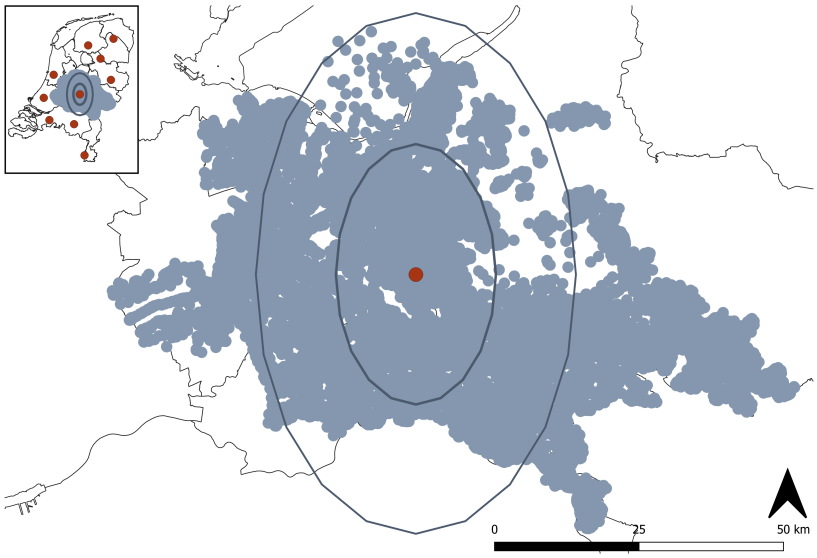

FIGURE 4.1: All postal codes covered by the depot of Veenendaal and the boundaries used to divide different areas using radius ranges.

Next, the parameters defining the number of generations and the number of parents that reproduce each generation are set, which are highly correlated. Whenever there are more generations a smaller number of parents suffices, and vice versa. It is important to notice that too few parents can result in a local minimum and too few generations mean too few iterations to result in the optimal solution. Too many generations and parents result in high computation times. The necessary number of parents and generations do not necessarily guarantee the optimal solution, since the discussed approach is a heuristic algorithm. The number of parents is scaled according to the number of dummy parcels, and set at 2% of the number of dummy parcels. The number of generations was set at 0.5% of the number of dummy parcels, but later fixed at five. This will later be explained in more detail.

Due to the fact that no chromosomes in the initial generation are constructed randomly, the number of needed generations is reduced significantly. Normally, it would take many generations until there are chromosomes with a fitness equal to those constructed using the sweep algorithm combined with split areas.

To design the reproduction process the criteria to appoint the actual parents are set. It is chosen to select the best chromosomes, which are the chromosomes with the highest fitness. The parents are paired randomly, this widens the search space. Randomly selecting parents or adding parents with a lower fitness would widen the search space even more. It is chosen not to do so, because selecting parents with the highest fitness reduces the running time and generates solutions of acceptable quality quickly. Note that fitness is calculated using the distance of the minimum spanning tree and therefore the lowest fitness is actually used for implementation purposes, this could of course be adjusted easily by taking the negative. For ease of

explanation, the highest fitness is referred to as the best chromosomes.

After reproduction has taken place, some children of the new generation are subjected to mutations. Each set of parents reproduces at least two children. Each child is with a 75% chance subjected to mutation, meaning a clone is created. In this clone, every parcel has a 2% probability to be moved to another vehicle, it has equal chance to move to the vehicle with the next, or the previous index. Note that at this point the clone is created from a child made through reproduction of ordered chromosomes in which the vehicles are ordered according to the angle of the centre of their visited customers, so the next and previous indexed vehicles lie in the surrounding area.

As said earlier, the number of generations was not fixed at five. At first, the number of generations was set at 0.5% of the number of dummy parcels. Using these settings, the dummy parcels corresponding to 02/11 were assigned to vehicles and routed. The parameter setting of the decomposition method are given in Table 4.4. The table presents the number of dummy parcels, the number of chromosomes in the initial generation, the number of parents in each generation and the final size of the pool. The computation time spent in the different parts of the decomposition method are given in Table 4.5. The table presents the time spent generating the initial generation, calculating the fitness values, selecting parents, applying crossover, applying mutation and the total computation time.

TABLE 4.4: Parameters of the decomposition method, considering 02/11.

Day	02/11
No. of dummy parcels	11,542
Initial generation	2,309
Generations	58
Parent	232
Pool	24,177

TABLE 4.5: Time spent in the different parts of the decomposition method, considering 02/11.

Day	02/11	Computation time
No. of dummy parcels	11,542	(h:m:s)
Initial generation		00:27:46
Fitness calculation		06:06:01
Parent selection		00:00:01
Crossover		00:00:13
Mutation		00:04:11
Total time		06:42:57

In the initial generation 2,309 chromosomes were generated using the sweep algorithm combined with split areas. After 58 generations in which 232 parents were selected, 24,177 chromosomes were generated. After 55 of the 58 generations still 217 of the 232 parents, were chromosomes generated in the initial generation. In addition, the chromosome with the best fitness value still originated from the initial generation. From the above explanation can be concluded that the number of parents as well as the number of generations should be enlarged, even though this is undesirable due to the increase in computation time.

Consider the dummy parcels corresponding to 03/11, a Saturday with only 2,093 dummy parcels. The decomposition method was applied using even larger percentages of parents and generations. When significantly increasing the percentages, children with fitness values higher

than the best fitness value from the original generation were found. Nonetheless the improvements were of insignificant difference, the increase in computation time did not outweigh the profit gained in fitness value. To be able to get the algorithm to find improvements for larger values, luck and extreme computation times would be necessary. Luck is needed due to the random element in the matching of parents and mutation. As a result, the number of generations is fixed at five. This significantly reduces the computation time but performs the genetic algorithm for the sake of demonstration.

After routes are generated using the decomposition method, the local search method, explained in Section 2.2.3, runs a preset number of iterations to optimise the routes. The local search optimises each route separately for a number of iterations equal to the squared number of parcels in that specific route. Results regarding the routes generated by the decomposition method before and after applying local search can be found in Table 4.6. The table presents, among others, the number of dummy parcels, the number of required vehicles and the total travelling distance and average duration of the routes, before and after local search is applied.

TABLE 4.6: Information and results regarding the generation of routes by applying the decomposition method.

			Parameters			Decomposition method			Local search		Computation time	
Day	Date	#DP	IG	#PC	#Pool	#T	TD (km)	AD (h:m)	TD (km)	AD (h:m)	DM (h:m)	LS (s)
Thu	01/11	11,820	2,364	238	4,016	83	15,271	7:17	14,938	7:10	1:45	63
Fri	02/11	11,542	2,309	232	3,943	86	15,437	6:58	15,129	6:52	1:40	54
Sat	03/11	2,093	408	42	709	23	5,613	7:43	5,442	7:33	0:01	4
Mon	05/11	9,193	1,839	184	3,106	68	12,739	7:18	12,454	7:12	0:50	46
Tue	06/11	14,167	2,834	284	4,817	98	17,131	7:04	16,771	6:57	10:51	103
Wed	07/11	12,458	2,492	250	4,232	87	15,636	7:11	15,326	7:05	2:05	65
Thu	08/11	11,475	2,295	230	3,907	84	15,016	7:01	14,700	6:56	1:38	60
Fri	09/11	11,219	2,244	226	3,845	81	14,762	7:08	14,430	7:02	1:36	55
Sat	10/11	2,128	246	44	735	23	5,627	7:52	5,434	7:42	0:01	3
Mon	12/11	9,202	1,841	186	3,152	68	12,974	7:21	12,714	7:14	0:52	47

The routes generated using the decomposition method have an average duration of 7:17 hours, each vehicle can drive, on average, almost an extra two hours. Due to the lack in efficient use of total travelling time, unnecessarily many tours are generated. The computation time of the decomposition method is acceptable for most instances. A computation time of a little over two hours when considering almost 12,500 dummy parcel suffices. Considering the instance of 06/11, where more than 14,000 parcels are routed, the computation time surpasses 10 hours. When increasing the number of dummy parcels, the size of the initial generation and number of parents increase accordingly, resulting in more clusterings (chromosomes) and more clusters (vehicles), which in turn results in higher computation times. As mentioned before the computation time of this step is not of the utmost importance. All information required to route the set of dummy parcels is available six days in advance. The local search method runs quickly, reducing the total travelling distances and the average duration. The computation time did not surpass two minutes and the total distance was reduced by 2.3% on average.

4.2.3 Comparison of results

The routes generated using the VRP heuristic and the decomposition heuristic (after applying local search) are compared to Method 2 discussed in Los (2019). The computation times of Method 2 can be found in Table A.6. Note that no comparison is made to the DPD benchmark solutions since the routes considered in this step are generated using dummy parcels. The

main goal of all three methods is to minimise the total travelling distance. Results regarding the routes generated by Method 2, the VRP heuristic and the decomposition method can be found in Table 4.7. The table presents, among others, the number of required vehicles and the total travelling distance and average duration of the routes.

TABLE 4.7: Information and results regarding the routes generated as initial solution.

		Method 2[1]			VRP heuristic			Decomposition method		
Day	Date	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)
Thu	01/11	134	14,852	6:54	63	12,624	8:46	83	14,938	7:10
Fri	02/11	133	14,744	6:53	62	12,589	8:46	86	15,129	6:52
Sat	03/11	28	4,415	6:45	20	5,038	8:33	23	5,442	7:33
Mon	05/11	118	13,453	6:53	52	10,804	8:48	68	12,454	7:12
Tue	06/11	160	17,442	6:55	-	-	-	98	17,131	7:04
Wed	07/11	149	16,275	6:54	-	-	-	87	15,326	7:05
Thu	08/11	114	13,359	6:55	61	12,360	8:50	84	14,700	6:56
Fri	09/11	115	13,566	6:55	61	12,348	8:44	81	14,430	7:02
Sat	10/11	32	4,877	6:43	20	4,983	8:25	23	5,434	7:42
Mon	12/11	111	12,753	6:55	53	11,064	8:44	68	12,714	7:14

[1] Presented in Los (2019).

In most cases, the total travelling distance is lowest for routes generated using the VRP heuristic. Saturday is an exception, there Method 2 generated routes with the lowest total travelling distance. The total travelling distance of the routes generated using Method 2 and the decomposition method are comparable. Neither one of them outperforms the other comparing all days considered. Comparing the number of tours generated, both methods discussed in this thesis outperform Method 2. The VRP heuristic, in many cases, used less than half of the vehicles Method 2 used. It can be concluded that the VRP heuristic generated routes with the lowest total travelling distance along with using the least vehicles. The average duration of the routes, shows almost optimal utilisation of the total travelling time of nine hours. When considering the average duration of the routes generated using Method 2 or the decomposition method, the vehicles are not optimally utilised, since each vehicle preferably drives nine hours. When comparing the number of tours generated using Method 2 and the decomposition method, the total distance of the decomposition method is expected to be lower since the number of vehicles decreased significantly. In Method 2 more computation time was spent applying interroute and intraroute local search methods. When considering the route of a single vehicle it is probably near optimal, whereas the routes generated using the decomposition method may be improved. Note that the results generated using Method 2 consider Manhattan distances, whereas the VRP heuristic and decomposition method are applied to real world travelling distances and durations. When disregarding the time spent generating the distance matrix, which is somewhat unfair since calculating Manhattan distances is included in the computation time of Method 2, the computation times of Method 2 and the decomposition method are comparable considering instances up to 12,000, whereas the VRP heuristic takes significantly longer.

4.3 Dynamic assignment

In the last step, the assignment of the actual dynamic parcels to the vehicles is performed. For the assignment of each parcel, five routes in the surrounding area are considered. In Los (2019), the same number of routes is considered. These routes are selected due to the fact that their centres lie near the dynamic parcel. Other routes are expected to lie far from the parcel and are therefore not considered. In case a parcel cannot be inserted in one of these five routes a new

route is generated, at first only containing this single parcel. Note that the centres of the routes are generated without considering the depot, it is chosen to do so since the parcels should lie in groups, disregarding the depot.

To improve the routes during the process of assigning dynamic parcels, a local search is performed. It is chosen to perform a 2-exchange local search method after every 100 inserted parcels. Local search is not performed as extensively as was done after the initial solution was generated. The number of local search iterations are proportional to the number of parcels assigned to a route. The number of iterations equals the squared number of 10% of the number of assigned parcels. Therefore, on a route containing 80 parcels (dummy and dynamic), $8^2 = 64$ iterations of local search are performed. After all dynamic parcels are assigned and the remaining dummy parcels are removed from the routes, another, more extensive local search is performed. The number of iterations applying the local search to a final route equals the squared number of assigned parcels.

Results regarding the dynamic assignment based on routes generated by the VRP heuristic and decomposition method will be discussed separately first. In Section 4.4, the routes generated in each of the two tracks are compared to each other along with the benchmark solutions of DPD and Method 2 from Los (2019).

Information regarding the dynamic assignment of parcels in routes generated using the VRP heuristic can be found in Table 4.8. The table presents, among others, the number of required vehicles and the total travelling distance and average duration of the routes.

TABLE 4.8: Information and results regarding the dynamic assignment in routes generated by the vehicle routing problem heuristic.

				VRP heuristic			Dynamic assignment			Local search		Computation time			
Day	Date	#DP	#P	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)	TD (km)	AD (h:m)	DA (h:m)	LSDA (h:m)	AP (s)	LS (s)
Thu	01/11	11,820	11,124	63	12,624	8:46	64	11,035	7:34	10,825	7:33	13:17	1:16	4.3	104
Fri	02/11	11,542	10,539	62	12,589	8:46	62	10,589	7:28	10,391	7:24	12:07	1:09	4.1	94
Sat	03/11	2,093	2,348	20	5,038	8:33	23	5,285	7:49	5,110	7:39	1:23	0:03	2.1	14
Mon	05/11	9,193	9,436	52	10,804	8:48	55	10,007	7:53	9,789	7:47	10:31	0:51	4.0	95
Tue	06/11	14,167	14,737	-	-	-	-	-	-	-	-	-	-	-	-
Wed	07/11	12,458	13,375	-	-	-	-	-	-	-	-	-	-	-	-
Thu	08/11	11,475	12,808	61	12,360	8:50	73	12,890	7:41	12,682	7:37	15:22	1:26	4.3	134
Fri	09/11	11,219	12,373	61	12,348	8:44	71	12,293	7:36	12,093	7:32	14:42	1:22	4.3	129
Sat	10/11	2,128	2,124	20	4,983	8:25	21	4,718	7:37	4,566	7:30	1:12	0:02	2.0	12
Mon	12/11	9,202	9,693	53	11,064	8:44	59	10,765	7:46	10,508	7:38	10:41	0:52	4.0	93

The total travelling distance of almost all routes after dynamic assignment is less than before, meaning that the one by one assignment of parcels did not negatively influence the total distance. Some more vehicles were used than generated in the VRP heuristic. This is especially the case if the number of dummy parcels underestimated the number of dynamic parcels, as can be seen for 08/11 and 09/11. Added vehicles negatively influence the total distance since they deliver parcels which in the first place could not be assigned to vehicles in the surrounding area. The average duration of a route is lower than the maximum travelling time, meaning that the vehicles are not optimally utilised. The same, or lower, total travelling distance should be achievable with less vehicles. Part of the computation time is devoted to the local search heuristic improving the routes, this is most probably the reason that the average duration is lower after the dynamic assignment compared to before. The computation time of the dynamic assignment is important since the assignment of parcel to vehicles is in a real-time setting, not all instances considered allow overnight assignment.

Information regarding the dynamic assignment of parcels in routes generated using the decomposition method can be found in Table 4.9. The table presents, among others, the number of required vehicles and the total travelling distance and average duration of the routes.

TABLE 4.9: Information and results regarding the dynamic assignment in routes generated by the decomposition method.

				Decomposition method			Dynamic assignment			Local search		Computation time			
Day	Date	#DP	#P	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)	TD (km)	AD (h:m)	DA (h:m)	LSDA (h:m)	AP (s)	LS (s)
Thu	01/11	11,820	11,124	83	14,938	7:10	83	12,623	6:15	12,447	6:09	9:55	1:08	3.2	93
Fri	02/11	11,542	10,539	86	15,129	6:52	86	12,633	5:47	12,410	5:42	8:39	1:00	3.0	76
Sat	03/11	2,093	2,348	23	5,442	7:33	25	5,303	7:05	5,121	6:58	1:16	0:03	1.9	14
Mon	05/11	9,193	9,436	68	12,454	7:12	68	10,889	6:31	10,718	6:26	7:36	0:44	2.9	79
Tue	06/11	14,167	14,737	98	17,131	7:04	99	14,885	6:25	14,710	6:17	14:21	1:48	3.5	129
Wed	07/11	12,458	13,375	87	15,326	7:05	87	13,260	6:40	13,071	6:32	12:12	1:25	3.3	120
Thu	08/11	11,475	12,808	84	14,700	6:56	88	13,971	6:28	13,817	6:24	11:31	1:17	3.2	114
Fri	09/11	11,219	12,373	81	14,430	7:02	82	12,821	6:38	12,660	6:34	10:58	1:12	3.2	109
Sat	10/11	2,128	2,124	23	5,434	7:42	23	4,795	7:02	4,613	6:54	1:00	0:02	1.7	13
Mon	12/11	9,209	9,693	68	12,714	7:14	67	11,028	6:43	10,815	6:36	8:38	0:49	3.2	81

The total travelling distance of the routes after dynamic assignment is less than before, meaning that the one by one assignment of parcels did not negatively influence the total travelling distance. Reflecting on the average duration of a vehicle, the same, or lower, total travelling distance should be achievable with less vehicles. Not many extra vehicles were used in comparison to generated in the decomposition method, even if the number of dynamic parcels was underestimated.

Applying to both, the dynamic assignment in the VRP heuristic track and the decomposition track, the average duration of the vehicles is lower compared to before the assignment of dynamic parcels. Presumably there was room for improvement on the routes generated in the second step. The computation time of the dynamic assignment in the decomposition track is significantly lower than that of the VRP heuristic track, this is probably due to the availability of more routes with lower average duration, resulting in less time infeasibility and therefore less complicated insertions.

4.4 Comparison of results

Routes generated after dynamic assignment in the VRP heuristic track and the decomposition method track are compared to the routes of DPD and Method 2 discussed in Los (2019). The computation times of Method 2 can be found in Table A.6. The main goal was to reduce the total travelling distance compared to the current method of DPD for routing customers. Efficiently routing the customers will not only reduce the total travelling distance but also the number of vehicles used. Results regarding the routes generated by DPD, Method 2, the VRP heuristic track and the decomposition method track can be found in Table 4.10. The table presents, among others, the number of required vehicles and the total travelling distance and average duration of the routes.

TABLE 4.10: Information and results regarding the routes generated as dynamic solution.

		DPD[1]			Method 2[1]			VRP heuristic			Decomposition method		
Day	Date	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)	#T	TD (km)	AD (h:m)
Thu	01/11	141	16,612	6:08	134	14,004	6:30	64	10,825	7:33	83	12,447	6:09
Fri	02/11	140	16,998	6:08	133	13,842	6:52	62	10,391	7:24	86	12,410	5:42
Sat	03/11	43	6,048	5:29	30	4,517	7:28	23	5,110	7:39	25	5,121	6:58
Mon	05/11	125	15,260	6:28	118	12,675	6:53	55	9,789	7:47	68	10,718	6:26
Tue	06/11	155	22,803	7:58	160	16,489	7:24	-	-	-	99	14,710	6:17
Wed	07/11	147	17,851	7:04	149	15,418	7:13	-	-	-	87	13,071	6:32
Thu	08/11	147	18,152	7:00	121	14,170	7:59	73	12,682	7:37	88	13,817	6:24
Fri	09/11	147	17,665	6:43	149	16,054	6:43	71	12,093	7:32	82	12,660	6:34
Sat	10/11	40	5,697	5:32	32	4,484	7:02	21	4,566	7:30	23	4,613	6:54
Mon	12/11	130	15,946	6:20	111	12,097	7:25	59	10,508	7:38	67	10,815	6:36

[1] Presented in Los (2019).

Considering the results of DPD and Method 2, using the same quality of the routes, an improvement can be made since the average duration of the routes is less than nine hours, therefore the vehicles used are not utilised optimally. All three methods outperform DPD considering the total travelling distance. The VRP heuristic and the decomposition method outperform Method 2 for almost all instances considered. In addition, the number of used vehicles is comparable for DPD and Method 2, whereas the decomposition method uses a little more than half of the vehicles DPD uses. The VRP heuristic track used less than half the vehicles used by DPD, but was unsuccessful in generating solution for all instances considered. The computation time of Method 2 is significantly lower compared to the VRP heuristic and the decomposition method. Considering the average duration of the routes generated by the VRP heuristic and the decomposition method, it is possible to reduce the number of vehicles even more, since the vehicles are not utilised optimally.

Chapter 5

Discussion

This thesis proposes two tracks to construct a set of routes so that each parcel arriving at the depot will be delivered to the customer. The total travelling distance must be minimised while satisfying the maximum travelling time constraint and optimising the number of vehicles used. The three step approach applied in this thesis is an altered and improved version of the three step approach proposed in Los et al. (2020). Historical data is used in a more elaborate way to generate dummy parcels. The VRP heuristic is based on the insertion heuristic of Solomon (1987) and the decomposition method is based on the method proposed in Cheng and Wang (2009), where many aspects of the genetic algorithm are altered. This way two new methods to route the dummy parcels are introduced. The third step is based on the dynamic assignment proposed in Los et al. (2020). By combining different methods, each based on earlier research, as well as original ideas, two tracks are constructed to generate solutions for the dynamic assignment vehicle routing problem. The proposed tracks are applied to instances provided by DPD. The methods proposed in this thesis outperform the method used by DPD.

In the second step of the approach, dummy data generated in the first step is routed. Two different methods are introduced, the VRP heuristic and the decomposition method. The combination of the insertion heuristic, which tackles the problem as a whole, and the decomposition method, which decomposes the problem in a main cluster problem and many smaller travelling salesman sub-problems provides an adequate balance between the two different approaches.

The VRP heuristic and the decomposition method are both heuristic algorithms, therefore results generated are not guaranteed to be optimal. Heuristic algorithms are applied since checking all possible solutions and selecting the best requires infeasible computation times. Additionally, generating near optimal solutions is unnecessary since the dummy parcels are merely an approximation of the dynamic parcels. Overfitting could be a concern if routes are nearly optimal and the dummy parcels do not perfectly represent the dynamic parcels.

The VRP heuristic and the decomposition method require the knowledge of each pairwise travelling distance and duration of dummy parcels to generate a vehicle routing problem solution. Generating the distance and duration matrix is done using the open source routing machine of Luxen and Vetter (2011), this way the solutions generated in this thesis are based on actual travelling distances and durations. Generating a distance and duration matrix when given a 1000 locations takes approximately 60 seconds. For large instances it took up to 400 OSRM calls to generate the entire matrices, taking almost six hours. If one wishes to avoid long computation times, approximations could be used. Travelling distances and durations could be generated using an approximation, for example the Manhattan distance. The data considered is based solely on the postal code, not considering the house number, and the routes considered are generated using heuristic algorithms. Therefore, distances and durations based on approximations might not have a notable negative influence on the results generated and might reduce the computation time since the distance and duration matrix are not calculated using the OSRM.

The VRP heuristic has many adjustable variables, these are defined in a way that heavily weighs distance, in order to ensure that customers served by a single vehicle lie geographically clustered. The local search method at the end of the second step was able to reduce the total travelling distance. However, this caused the average duration of the routes to reduce, resulting in an average duration less than the maximum duration of nine hours, therefore further optimisation could be applied in order to make optimal use of the vehicles. If the routes are optimised while applying the VRP heuristic, more parcels would be assigned to a single vehicle and less vehicles would be needed. Note that the VRP heuristic is a sequential tour-building heuristic, whenever a new vehicle is introduced, no more parcels are added to earlier routed vehicles.

The VRP heuristic has infeasible computation time for the largest instances. Each iteration considers all unrouted customers inserted in all possible places in the route under construction, this process is very time consuming and can be parallelised. As defined in Prabhakaran (n.d.), "parallel processing is a mode of operation where the task is executed simultaneously in multiple processors in the same computer". It is meant to reduce the computation time.

The decomposition method was designed based on the idea that assigning vehicles to clustered customers made dynamic insertion in the third stage easier, since a dynamic parcel would be inserted in the vehicle of the cluster covering the surrounding area. Unfortunately, despite reducing the computation time in multiple ways, like changing the fitness function to a minimum spanning tree instead of a travelling salesman problem, the number of required generations could not be performed in feasible computation time. The sizes of the instances provided by DPD are too large to generate solutions using a genetic algorithm with the design choices proposed in this thesis.

In the event that only 5 generations are performed, the importance of the initial generation is clearly illustrated. The total travelling distance of the resulting routes is comparable to the total travelling distance of Method 2. From this it can be learned that generating acceptable clusterings can be done in little time if the genetic algorithm process is left out. The sweep algorithm combined with split areas was designed to construct chromosomes that form the initial generation and not the final clustering. Many aspects, like splitting the area in three parts, could be adjusted to obtain better results. That way, the entire genetic algorithm could be replaced by a heuristic designing the chromosomes of the initial generation. Routes could be generated for all instances considered, and even larger ones, and the results will outperform the decomposition method and can be generated in less computation time.

In the construction of the initial generation of the decomposition method, longitude and latitude degrees of dummy parcels are used as Cartesian coordinates to be converted into polar coordinates. The radius and angle of the polar coordinates are used in the sweep algorithm combined with split areas. Ordering parcels using the polar coordinates, generated using longitude and latitude as approximations of Cartesian coordinates, suffices. To perform this process accurately, longitude and latitude should be converted into three dimensional Cartesian coordinates, (x, y, z), using the centre of the earth as origin. The three dimensional Cartesian coordinates can be converted into three dimensional polar coordinates (r, θ, ϕ). Proper proportions of the boundaries of the split areas could also be achieved by generating an actual ellipse, setting different radius values in longitudinal and latitudinal direction.

In the last step the assignment of the actual dynamic parcels to the vehicles is performed. For the assignment of each parcel, five routes in the surrounding area are considered. The computation time of step three in both tracks does not allow overnight assignment for all instances considered. By reducing the number of routes considered the computation time would decrease, which makes it suitable for real-time application. This might increase the total travelling distance since less routes are considered and therefore lower quality insertions are possibly performed.

Chapter 6

Conclusions and recommendations

In a dynamic assignment vehicle routing problem the depot considered has no space available to store all parcels before assigning them to the different vehicles. Therefore, a parcel is scanned upon arrival and immediately assigned to a vehicle. The assignment to vehicles is done in such a way that routes generated for each vehicle minimise the total travelling distance while considering the nine hour maximum travelling duration of a vehicle and optimising the number of vehicles used.

This thesis describes two tracks to a three step approach to solve the dynamic assignment vehicle routing problem. In the first step historical data is combined in a way that allows to predict where parcels for the upcoming day need to be delivered, these predicted parcels are called dummy parcels. The second step solves the vehicle routing problem based on these dummy parcels. The first method introduced does so by applying an insertion heuristic. The second method decomposes the original problem in a clustering problem (main problem) and a set of travelling salesman problems (sub-problems). The solution is obtained through iterative interactions between the main problem and the set of sub-problems. In the third step, the dummy parcels are one by one replaced by incoming parcels, resulting in a solution to the dynamic assignment vehicle routing problem. The routes constructed using the methods described in this thesis reduce the total travelling distance and the number of vehicles used compared to DPD for all instances considered.

In the first step historical data is combined to generate dummy parcels. It is chosen to combine data of the past three weeks corresponding to the same day of the week. The method to combine the historical data is called weighted K-means clustering. The number of clusters, equalling the number of dummy parcels, is defined as the average number of parcels delivered on the three corresponding historical days. The weight is based on the number of deliveries on that specific day and the locations of the dummy parcels are based on the centres of the clusters. The method performs effectively as the number of generated dummy parcels and the placements represent the actual arriving parcels.

In the second step two methods are introduced: the VRP heuristic and the decomposition method. The VRP heuristic is a sequential tour-building insertion heuristic. The customer inserted has the highest benefit to be served on the partial route rather than on a direct route. The best insertion place is one that minimises the weighted combination of the added distance and time after insertion. The decomposition method is a genetic algorithm, which formulates clusters to assign to vehicles. The performance of a cluster is based on the distance of the minimum spanning tree.

In the third step each dummy parcel is substituted by an incoming dynamic parcel. The computation time of the dynamic assignment is important since the assignment of parcels to vehicles is done in a real-time setting. The computation time of step three in both tracks does not allow overnight assignment for all instances considered. The total travelling distance of the routes after the dynamic insertion is even lower than before, the local search method optimising the routes throughout the dynamic insertion may have contributed to this decrease.

Regarding the average duration of the routes can be concluded that they are not optimally utilised. Delivering the parcels should be possible using less vehicles. The reduction of the number of vehicles should be done in the second step. Applying local search methods on the routes generated throughout this step optimises distance and duration of the routes and therefore allows for the assignment of more parcels to a single route, resulting in less used vehicles.

Recommended is to adapt the second step of the three step approach as here most improvement can potentially be made. A better routing of the dummy parcels will result in a lower total travelling distance and less vehicles used. Only when vehicles are routed nearly optimal, overfitting could be a concern due to the leeway needed for the dynamic insertion of parcels later.

The VRP heuristic generally provides the shortest routes using the least vehicles, but was unable to generate solutions to all instances considered. Reducing the computation time, for example by parallelising the algorithm, would make this method suitable for application.

The decomposition method performed well due to the construction of the initial generation. The computation time of the decomposition method increased significantly for larger instances. For larger instances many chromosomes are generated in the initial generation, each of these chromosomes containing a large number of parcels. Calculating the fitness of large chromosomes takes longer. For larger instances, the five new generations contain many children since many parents were selected. Separating the process of formulating an initial generation from the genetic algorithm would result in routes of similar quality in less computation time.

It can be concluded that the routes generated by the VRP heuristic track performed best, compared to DPD the total travelling distance significantly decreased and less than half the number of vehicles were used. Due to infeasible computation time this method was not able to generate solutions for all instances considered.

The decomposition method did generate routes for all instances considered. The total travelling distance significantly decreased and the number of vehicles were reduced compared to DPD. All routes generated using the VRP heuristic track and the decomposition method track reduced the total travelling distance while keeping the route duration less than nine hours and minimising the number of vehicles used.

Bibliography

[1] M. Bellmore and G. L. Nemhauser. "The Traveling Salesman Problem: A Survey". In: *Operations Research* 16.3 (1966), pp. 538–558. DOI: 10.1287/opre.16.3.538.

[2] C. Cheng and K. Wang. "Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm". In: *Expert Syst. Appl.* 36 (2009), pp. 7758–7763. DOI: 10.1016/j.eswa.2008.09.001.

[3] N. Christofides. *Worst-case analysis of a new heuristic for the travelling salesman problem*. Tech. rep. Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[4] Wikipedia contributors. *Longitude*. (visited on 20/01/2021). 2021. URL: https://en.wikipedia.org/wiki/Longitude.

[5] M. Fleury. "Deux problemes de geometrie de situation". In: *Journal de mathematiques elementaires* 2.2 (1883), pp. 257–261.

[6] M. R. Garey and D. S. Johnson. *Computers and intractability*. Vol. 174. freeman San Francisco, 1979.

[7] B. E. Gillett and L. R. Miller. "A Heuristic Algorithm for the Vehicle-Dispatch Problem". In: *Operations Research* 22.2 (1974), pp. 340–349. ISSN: 0030364X, 15265463. URL: http://www.jstor.org/stable/169591.

[8] J. B. Kruskal. "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem". In: *Proceedings of the American Mathematical Society* 7.1 (1956), pp. 48–50. ISSN: 00029939, 10886826. URL: http://www.jstor.org/stable/2033241.

[9] K. J. Los. "Improving delivery efficiency using a dynamic assignment method: a DPD case study". MA thesis. the Netherlands: Erasmus University Rotterdam, 2019.

[10] K. J. Los, F. Phillipson, E. A. van Kempen, H. J. Quak, and U. Stelwagen. "Dynamic Assignment Vehicle Routing Problem with Time Windows". In: *Proceedings of 11th International Conference on Computational Logistics*. Enschede, The Netherlands, 2020, pp. 135–150.

[11] D. Luxen and C. Vetter. "Real-time routing with OpenStreetMap data". In: *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. GIS '11. Chicago, Illinois: ACM, 2011, pp. 513–516. ISBN: 978-1-4503-1031-4. DOI: 10.1145/2093973.2094062. URL: http://doi.acm.org/10.1145/2093973.2094062.

[12] F. Phillipson and S. de Koff. "Immediate parcel to vehicle assignment for cross docking in city logistics". In: *Proceedings of 9th International Conference on Operations Research and Enterprise Systems (ICORES)*. Valetta, Malta, 2020, pp. 138–142.

[13] F. Phillipson, S. de Koff, C. van Ommeren, and H. J. Quak. "Dynamic assignment vehicle routing problem with generalised capacity and unknown workload." In: *Proceedings of 9th International Conference on Operations Research and Enterprise Systems (ICORES)*. Valetta, Malta, 2020, pp. 329–335.

[14] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia. "A review of dynamic vehicle routing problems". In: *European Journal of Operational Research* 225.1 (2013), pp. 1–11.

[15] S. Prabhakaran. *Parallel Processing in Python - A Practical Guide with Examples*. (visited on 20/01/2021). URL: https://www.machinelearningplus.com/python/parallel-processing-python/.

[16] R. C. Prim. "Shortest connection networks and some generalizations". In: *The Bell System Technical Journal* 36.6 (1957), pp. 1389–1401.

[17] M. M. Solomon. "Algorithms for the vehicle routing and scheduling problems with time window constraints". In: *Operations research* 35.2 (1987), pp. 254–265.

[18] A. Vyas. *Genetic Algorithm*. (visited on 20/01/2021). 2018. URL: https://medium.com/@anshul.vyas380/genetic-algorithm-5ba5eb5b7090.

Appendix A

Appendix

A.1 Number of deliveries

TABLE A.1: Number of parcels delivered on specific weekdays in specific weeks.

September							
Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	37	7,331	11,651	11,433	10,289	9,972	1,857
	38	8,252	11,984	12,653	11,224	443	516
	39	9,338	14,942	13,056	12,064	12,756	2,255
October							
Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	40	9,877	14,178	12,543	11,054	12,037	2,291
	41	9,496	14,424	13,133	12,161	11,508	2,080
	42	9,409	14,046	12,041	11,348	11,316	2,033
	43	7,783	13,415	12,062	11,951	11,802	2,002
	44	10,386	15,038	13,269			
November							
Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	44				11,124	10,539	2,348
	45	9,436	14,737	13,375	12,808	12,373	2,124
	46	9,693	14,585	12,838	12,508	11,446	2,085
	47	9,952	15,538	14,290	13,762	14,138	5,649
	48	14,339	19,784	18,834	17,605	17,956	
December							
Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	48						3,222
	49	10,677	15,170	14,171	11,468	6,802	5,733
	50	9,471	14,807	12,816	12,404	12,357	2,521
	51	10,359	14,693	14,103	14,197	12,515	3,018

A.2 Corresponding historical parcels

TABLE A.2: Percentage of customers who had a parcel delivered a week earlier on the same day of the week.

Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	38	29.00	31.89	33.24	31.74	40.92	7.57
	39	29.30	29.07	30.81	29.75	1.65	1.98
	40	28.69	32.22	32.97	31.59	31.93	10.74
	41	28.74	31.63	31.11	30.34	32.14	12.72
	42	30.26	32.44	32.92	32.16	32.12	11.31
	43	30.09	32.27	31.10	30.11	30.64	9.52
	44	23.10	29.33	30.13	30.54	30.91	9.00
	45	30.33	32.23	31.54	29.86	28.62	10.73
	46	28.99	32.60	31.89	32.32	31.73	9.26
	47	28.89	31.13	30.19	30.29	28.05	6.32
	48	23.82	28.46	28.50	28.66	29.36	16.15
	49	31.33	36.38	35.51	36.61	40.27	10.30
	50	34.18	33.24	32.68	30.76	21.30	18.70
	51	28.47	32.99	30.70	31.14	31.28	8.56
Average percentage		28.94	31.21	31.66	31.13	29.40	10.20

TABLE A.3: Percentage of customers who had a parcel delivered in the past three weeks on the same day of the week.

Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	40	48.98	53.16	54.79	52.57	44.20	17.01
	41	48.17	54.78	54.56	52.05	46.27	20.34
	42	50.47	55.97	54.92	53.11	54.45	22.16
	43	50.39	56.21	54.07	52.10	51.81	20.69
	44	44.59	53.26	52.64	51.64	50.67	18.81
	45	48.30	54.91	52.20	51.46	51.48	20.81
	46	48.24	55.72	54.16	52.58	52.46	19.92
	47	48.62	55.14	53.16	51.89	48.77	15.76
	48	42.99	51.14	49.93	49.49	49.93	23.94
	49	48.89	57.94	55.34	56.84	59.80	24.31
	50	55.71	59.66	58.05	56.37	53.89	33.25
	51	54.07	60.93	59.23	56.00	52.94	25.67
Average percentage		49.12	55.73	54.42	53.01	51.39	21.89

TABLE A.4: Percentage of customers who had a parcel delivered in the past five weeks on the same day of the week.

Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	42	60.76	67.54	66.01	64.33	60.80	26.40
	43	61.50	67.31	65.57	63.60	59.23	24.45
	44	56.41	66.27	65.34	63.06	63.00	27.14
	45	59.50	67.02	65.01	62.93	63.34	29.61
	46	60.14	68.12	65.76	63.34	63.55	27.94
	47	58.77	67.16	63.93	62.85	60.20	22.59
	48	53.60	64.08	61.44	60.98	60.48	29.97
	49	59.53	69.33	66.51	66.97	69.18	30.83
	50	64.92	70.33	68.30	66.84	64.87	39.29
	51	63.35	71.52	69.60	67.07	64.45	34.33
Average percentage		59.85	67.87	65.75	64.20	62.31	29.26

TABLE A.5: Percentage of customers who had a parcel delivered in the past ten weeks on the same day of the week.

Day of the week		Mon	Tue	Wed	Thu	Fri	Sat
Week number	47	74.57	82.34	79.36	78.32	74.49	33.41
	48	69.86	80.32	77.42	75.79	74.06	40.18
	49	73.84	83.14	80.54	79.86	81.66	42.02
	50	77.12	83.72	81.71	80.04	78.69	50.10
	51	75.98	84.24	82.27	80.02	77.84	44.05
Average percentage		74.27	82.75	80.27	78.81	77.35	41.95

A.3 Computation times Method 2

TABLE A.6: The computation times of Method 2[1].

Day	Date	Computation time		
		Initial solution (h:m:s)	Local search (h:m:s)	Dynamic solution (h:m:s)
Thu	01/11	56:22	48:40	59:30
Fri	02/11	1:08:28	56:05	46:21
Sat	03/11	3:57	4:04	4:18
Mon	05/11	55:48	1:06:45	41:25
Tue	06/11	59:33	1:08:12	50:49
Wed	07/11	1:02:01	1:02:34	1:01:23

[1] Presented in Los (2019).