Bachelor Thesis

Industrial Engineering and Management

Improving a Vehicle Routing Problem algorithm at Districon

Tom Huizingh

s2178567

23-08-2021

Faculty: Behavioural, Management and Social Sciences (BMS)

Supervisors University of Twente

1. Dr. Ir. E.A Lalla
2. Dr. Ir. W.J.A. van Heeswijk

Supervisor Districon

1. Olaf Witsen

UNIVERSITY OF TWENTE.

DISTRICON advisory solutions professionals

Acknowledgement

Dear reader, you are about to read my thesis that completes my Bachelor of Industrial Engineering and Management (IEM) at the University of Twente. This research has been conducted at Districon. However, before you are going to start reading the thesis, I would like to thank some people who were very important to me while writing this thesis.

First of all, I would like to thank Sebastien Piest for bringing me in contact with Districon because without him, this opportunity would have never been there. Secondly, in the first meeting with Districon, Niels Veenman introduced me to the company and offered me this great experience to perform a thesis assignment at Districon. Above that, I would like to give a special thanks to Olaf Witsen, who was my supervisor at Districon. Olaf was always available for questions and provided lots of helpful information and guidance at every stage of the research process. I am very grateful for the support, he gave me in this period. Furthermore, every meeting with an employee of Districon was pleasant and interesting, including the weekly meetings with Olaf. Although, I have not been able to visit the company that much due to the corona measures, I had a very nice time at Districon from short chats at the office till online team events. For this reason, I would like to thank every employee of Districon.

Secondly, I would like to thank Eduardo, who was my supervisor of the University. From the project plan until the final version of this thesis, he was always extensively guiding and supporting me. His feedback was very helpful, besides that he inspired me with his ideas and experience in this field of research. Moreover, I would like to thank Wouter for being my second supervisor

Finally, I would like to thank my family for taking into account that I had to write this thesis at home while they were free from school or work.

Tom Huizingh, Nieuwleusen, July 2021

Management summary

Districon is one of the world's leading consultancy firms when it comes to the area of supply chain and the application of analytics and decision support tools. Districon is located in Maarssen and has departments in the USA and Asia. Districon not only offers consultancy but also off-the-shelf IT solutions, interim project management, professionals and recruitment.

A company denoted as 'Company A' requested Districon to come up with a solution to automate their routing scheduling activities. Company A is the pioneer in producing a food ingredient with more than 50 years of experience, the highest quality products at minimal costs are provided to the customer. Company A uses two huge production factories that are located in two different places. Furthermore, the produced ingredient is distributed by Company A over the depots that are divided over the regions in the country. The customers of Company A are mainly supermarkets and convenience stores.

For the distribution of their products from the depots to the customers, Company A would like to have a Routing Planning Software (RPS) that produces high-quality routes in a relatively short computational time. Districon had already constructed a model in Python, from a former project, that is able to plan trips. This specific problem of Company A is denoted as a Heterogeneous Vehicle Routing Problem with Hard Time Windows (HVRPHTW). The problem is characterised as an NP-hard problem, implying that the optimal solution cannot be guaranteed within polynomial time. For this reason, Districon used a Tabu Search (TS) algorithm to find high quality (not optimal) solution in a relatively short time. However, the current model of Districon is constructed to solve Vehicle Routing Problems (VRPs) including homogeneous vehicles. Therefore, the goal of this research is to extend the current model such that it is able to solve a HVRPHTW.

Literature has been conducted in order to learn more about the HVRPHTW. The algorithm proposed in the paper of Molina, Salmeron and Eguia (2020) has been used to extend the current model of Districon. Firstly, the current model was adapted such that it could handle heterogeneous vehicles and afterwards the algorithm of Molina, Salmeron and Eguia (2020) was implemented. This paper introduces a Variable Neighborhood Tabu Search (VNTS), the main notion behind this VNTS algorithm is that a local optimum for one neighborhood does not necessarily have to be a local optimum for another neighborhood.

The new proposed algorithm consists of 3 phases: creation of the initial solution, improvement phase and the post optimization phase. The initial solution has been constructed by first allocating the customers random over the available vehicles and afterwards with help of the VNTS, the vehicles used in the solution will be minimized. This completes the initial solution and this initial solution will be improved by the VNTS in the improvement phase. This VNTS uses 6 different neighborhood structures: Relocate (inter- and intra-route), Exchange (inter- and intra-route), Cross-Exchange and the GENI-insertion. The algorithm terminates its search when 30 consecutive iterations without an improvement have been made. Subsequently, the best-found solution will be used as input solution for the post-optimization method. The post-optimization method attempts to split a route into two routes to evaluate if it is more cost-efficient to use two small vehicles instead of 1 big one. Finally, the VNTS will be applied once again to the output of the post-optimization method since new routes could be constructed by the post-optimization method which implements a new search area for the VNTS. In this way, the algorithm is able to tackle heterogeneous vehicle routing problems and make use of various different vehicles within a vehicle fleet.

This VNTS algorithm was implemented in the model of Districon, experiments have been executed in order to set every parameter value of the algorithm. Examples of parameters are the termination criterium and the minimum and maximum length of the Tabu List. Subsequently, this new proposed algorithm has been compared against the current algorithm and it could be concluded that on average the newly proposed algorithm outperforms the current algorithm. The new algorithm outperforms the current algorithm on 16 out of 24 instances. On average the objective value is 7.1% lower and even the computational time is on average 382 seconds lower. Furthermore, the average truck capacity utilization is 0.95, while the current algorithm only has a truck capacity utilization of 0.89. However, in some cases the new proposed algorithm is struggling with the provided initial solution and is not able to find a feasible solution. Since the computational time is, in most cases, well within the 15 minutes this causes not always trouble. By implementing the new algorithm into the model, the requirements of Company and the goal of this thesis are fulfilled.

The main recommendation to Districon is to conduct further research since the new algorithm has a high potential and due to the limited time in this research, it could not be fully exploited. Firstly, it is recommended to Districon to investigate why the VTNS is not able to find a feasible solution in a few cases or a new creation method for the initial solution could be constructed. Furthermore, to enlarge the search area of the algorithm, Districon should enable the algorithm to create and remove trips while iterating. Finally, it is recommended to extend this single depot model to a multi-depot model since Company A serves its customers from multiple depots. Besides, this extension could be used for future projects as well, since many companies that consult Districon have multiple depots.

Contents

1. Introduction

In this chapter, the plan of approach of my bachelor thesis assignment, where a core problem within a company will be solved, will be provided. This assignment has been offered by Districon. This assignment will be carried out in order to graduate from the bachelor program of Industrial Engineering and Management (IEM) at the University of Twente.

1.1 Description of the involved companies

1.1.1 Districon

Districon is one of the world's leading consultancy firms when it comes to the area of supply chain and the application of analytics and decision support tools. Districon is located in Maarssen and has departments in the USA and Asia. The name Districon is based on the keywords **Distri**bution and **Con**sultants. Districon not only offers consultancy but also off-the-shelf IT solutions, interim project management, professionals and recruitment. These services are divided into three working units: advisory, professionals and solutions. The current trend in clients and projects show that the projects become more international orientated and that the demand for off-the-shelf and tailor-made IT solutions is rising. The unit Solutions responds to this trend. Districon Solutions are developers of key supply chain analytics capabilities by enabling their customers to extract and translate the right supply chain data, to design and implement smart supply chain models and planning applications, and to self-enable users to create business value.

1.1.2 Company A

Company A does only sell a few sorts of products however these products have multiple different product types. For each product sort, a subsidiary has been founded to make the distinction between the products since they are specialist in their specific domains. The core business of the parent company is an ingredient of food, due to confidentially issues the name of the product is not mentioned. Company A is the pioneer in producing this ingredient with more than 50 years of experience, the highest quality products at minimal costs are provided to the customer. Company A uses two large production factories that are located in two different places. Furthermore, the produced ingredient is distributed by Company A over the depots that are divided over the regions in the country. The customers of Company A are mainly supermarkets and convenience stores.

1.2 Problem identification

In this section, the analyse of the core problem is documented. Firstly, the current situation is described in Section 1.2.1. Secondly, the action problem specified by Districon is stated in Section 1.2.2, followed by a problem cluster in Section 1.2.3 that is used to establish the core problem. The core problem is constructed in Section 1.2.4 and finally in Section 1.2.5 the research motivation is provided.

1.2.1 Current situation

This project is empowered by Districon, specifically the solutions unit, however, it is mainly executed for the sake of a customer of Districon which is called Company A. They provide their customers with their products which are stored in warehouses. To complement the stocks of the clients of Company A uses multiple trucks with varying maximum speed limits and capacity. Furthermore, the warehouses or stores which should be supplemented all have a different time window in which the products should be delivered. This results in a very complicated form of a Vehicle Routing Problem (VRP), namely a Vehicle Routing Problem with Time Windows (VRPTW) including heterogeneous trucks, which is an NP-(nondeterministic polynomial) hard problem (Khachay & Ogorodnikov, 2019). Currently, human planners are scheduling the trips for every truck through the depots without the use of a model or decision helping tool. Company A contacted Districon to come up with a model which automates the planning of the routing schedule for the trucks of the last mile. The last mile includes the delivery of goods from depot to clients. The first mile, direct or milk run, includes the replenishment of stock from production factory to depot. The first mile is outside the scope of this research since optimizing this routing schedule was not requested by Company A. Finally, Company A provided a dataset to Districon that represents the data of one region and is analysed in Section 3.4. This dataset contains a bit more than 100 vertices. Currently, the human planners need 8 vehicles to supply every customer.

Districon's model

Districon has already made a start with this project and designed a model which solves a VRPTW but does not distinguish between the different driving speeds and capacities of the trucks. This VRP is solved with the use of the nearest neighborhood heuristic in combination with a Tabu Search heuristic. The input variables are listed in excel and then implemented in the model in Python. Examples of input variables are truck capacity, travelling time and distance between Depot A and a customer, length of the Tabu List and the maximum iterations, identifying when the algorithm must stop iterating.

This model has been applied to the dataset of Company A and the outcome of the model was a routing schedule including 7 routes. Consequently, the model constructed by Districon uses one vehicle less than the solution of the human planners. This means that Company A could economize on the costs of one vehicle. Detailed information about these constructed routes can be found in Appendix B-5.

1.2.2 Action problem

The action problem, which is given by Company A and what needs to be solved by Districon, is that the current transporting routes of Company A could be more efficient and therefore less costly and more profitable. The current model of Districon cannot be used by Company A since this model excludes the input of heterogeneous vehicles which are used by Company A.

1.2.3 Problem Cluster

We have made a problem cluster in order to get a clear overview of all the problems and consequences. A problem cluster is used to map all problems with their connections and to identify the core problem (Heerkens & van Winden, 2017). This problem cluster has been made with the use of interviews with employees of Districon who are in direct contact with Company A. The action problem can be found in the green box in Figure 1.

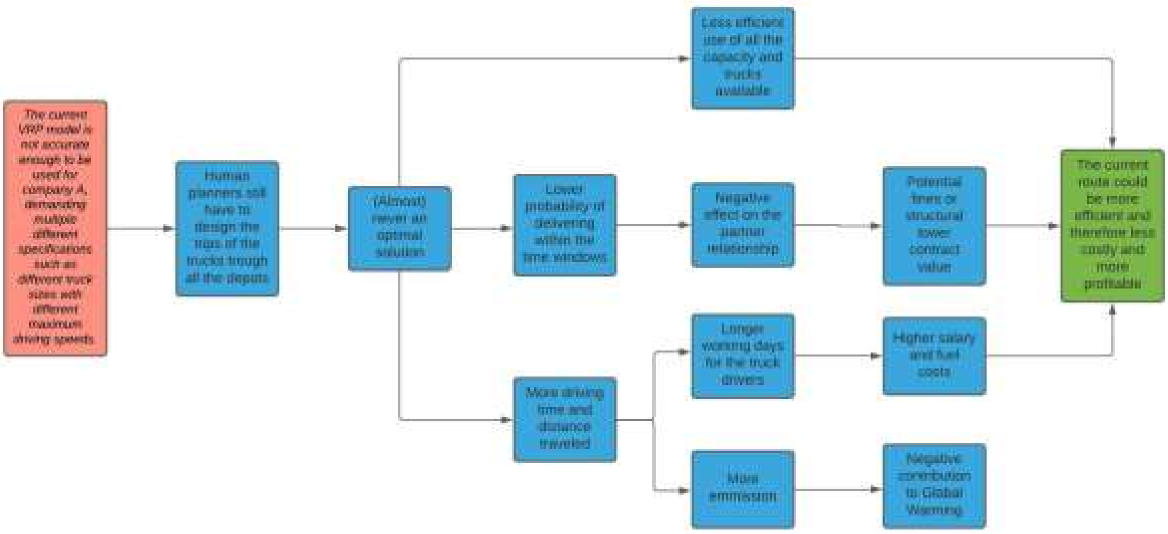


Figure 1: Problem cluster of Company A's core problem

From this problem cluster, it could be obtained that an inefficient transport schedule implements that the total distance travelled by all the trucks together could be decreased. This means that trucks will

drive more than required which results in longer working days for the truck drivers and more fuel. Consequently, the fuel cost and salary costs will increase.
Secondly, a less efficient solution to this VRP will result in a lower probability of delivering within the time window. As a result, the reliability of Company A to its customers will decrease and negatively impacts the relationships with their partners. Furthermore, not delivering in time could end up in fines or even in a structural lower contract value.
Finally, an inefficient solution leads to less efficient use of all the capacity and trucks available. For example, in a route schedule are 11 trucks needed and it is possible to use 10 trucks, one truck could be saved. This will decrease all the costs of this specific truck and moreover, the truck and/or driver can be used for other activities.
The cause of this inefficient solution is the fact that human planners are designing the whole routing schedule for every truck. Although they are experienced and trying their very best, they could never outperform a computer that uses algorithms and heuristics as could be seen in Section 1.2.1 where the model uses one less vehicle than the human planners.
The reason why human planners still have to calculate the routing trips per truck is that the model of Districon is not well enough developed. Therefore, Company A cannot make use of the model of Districon and is forced to plan the routes by human planners.

1.2.4 Core problem

In the problem cluster (Figure 1) can be seen that the action problem, in the green box, is caused by several problems: higher fuel and salary costs, potential fines or structural lower contract values and less efficient use of all the truck capacity available. This has been caused by inefficient solutions of the VRPTW with heterogeneous trucks which has been made by human planners. All in all, from the problem cluster can be obtained that all the problems, in the end, start with the model of Districon which is not applicable to this case of Company A (red box in problem cluster). This has resulted in the following core problem:

The current VRP model is not accurate enough to be used for Company A, demanding multiple different specifications such as different truck sizes with different maximum driving speeds.

1.2.5 Research motivation

Automating the planning process of this customer of Districon has many positive consequences. First of all, it will save the human planners plenty of time since they do not have to come up with a self-made solution of the VRPTW with heterogenous trucks. Planning algorithms can do this significantly quicker with support of computers. Furthermore, the output of the model will be closer to the optimal solution which implements less driving distance and time. Moreover, it will improve the relationship with the partners of Company A since the reliability of the supplementation of the products will be delivered more often in the time windows. For these reasons, the company's value increases, and the costs decreases. Besides, employees of Districon will be saved a lot of time since I am going to extend the model with the distinction between heterogeneous vehicles. Finally, this new model can be used for many more cases in the future which implies potential new profits.

1.3 Research design

After having determined the core problem and the research motivation is formulated, in Section 1.3.1, the way of solving the action and thus the core problem will be discussed on the basis of the stated research questions. Finally, in Section 1.3.2 the deliverables provided at the end of this research are composed.

1.3.1 Research questions

In order to solve the core problem, the following main research question should be answered: *How can the current model of Districon be adapted in order to fulfil all the needs of Company A?*
This main research question has been split up into six research questions which will all contribute to the new extended version of the VRP model.

1. What can literate tell us about ….

1. *the various forms of a vehicle routing problem?*
   In this thesis assignment, multiple variants of VRPs are mentioned and the most relevant forms of a VRP to this research will be elaborated on, in order to increase the understandability.
2. *the possible solving methods for a VRP and what is a Tabu search?*
   While identifying the core problem by interviewing employees of Districon, it was noticed that a Tabu Search (TS) was used in the current model of Districon. Thus, a literature study will be conducted in order to fully understand the algorithms used in the model and to be able to implement the heterogeneous vehicles in the TS.
3. *how VRPTW with heterogeneous vehicles are solved with help of a Tabu Search?*
   The goal of this research is to properly implement the distinction between heterogeneous vehicles into the existing model of Districon. However, before the implementation of this feature can be applied, we need to know how VRPTW with heterogeneous vehicles are solved with help of a TS. A literature study will be conducted in order to obtain this essential information.

2. What is Company A exactly demanding from Districon and what is Districon's intention with respect of the model?

Before researching on how to solve the action problem, it is necessary to know what is expected from the solution. In other words, what exact requirements does Company A want to be included in the solution and thus in the model? Furthermore, the current situation needs to be identified in more detail. Besides, what are Districon's interests in the model after this project is finished? A context analysis will be conducted with use of interviews with the different stakeholders to obtain this information.

3. How is Districon's model currently solving a VRPTW?

Since in previous sections is obtained that the core problem is that the model is not appropriate for this case of Company A, understanding the current model is indispensable. This model of Districon is able to solve a VRPTW without heterogeneous vehicles. As this will be the foundation of my solution to the core problem it is essential to thoroughly study this model. An analysis will be executed to find out what algorithms and heuristics are used and how they are used together. Furthermore, the motivation behind the choices of those algorithms and heuristics will be gathered by interviewing the designers of the model.

4. How can the distinction between heterogeneous vehicles be implemented and evaluated in the current model in Python?

After the way of solving a VRPTW with heterogeneous vehicles is obtained in the literature study, the model needs to be designed and build in Python. Firstly, the current model of Districon will be extended such that it is able to solve a heterogeneous VRPTW. Secondly, a new algorithm based on the literature research will be implemented in a new model. Furthermore, the parameters of the new algorithm need to be set. Examples of parameters of the algorithm are the maximum number of iterations and the length of the Tabu List. This will be done by use of experiments. Finally, the new model should be evaluated against the current model by applying both to different data sets of Company A and observe which model produces the best outcomes.

5.What recommendations and conclusions can be made after executing this research at Districon?
Finally, after all the research has been conducted and the solution has been implemented and evaluated, conclusions can be made and recommendations for further research will be provided.

1.3.2 Deliverables
This section contains the main deliverables that we are going to provide at the end of this bachelor thesis assignment which is performed at Districon.

- A VRP model programmed in Python which includes the distinction between different truck sizes. The output of the model is an efficient as possible transport route which has the lowest objective value.
- The whole process will be documented in a report where every decision will be explained and recommendations for further extensions of the model will be discussed. This report includes the theoretical framework, problem-solving approach and the implementation and evaluation of the solution.

2. Literature review

This chapter contains the theoretical framework of this research about the core problem and the implementation of the solution. In Chapter 1, the problem statement is defined and the research questions have been formulated. For the following research questions, a literature study is conducted in this chapter to gain knowledge about these specific subjects:

What can literate tell us about ….

1) *the various forms of a vehicle routing problem?*
2) *the possible solving methods for a VRP and what is a Tabu search?*
3) *how VRPTW with heterogeneous vehicles are solved with help of a Tabu search?*

In Section 2.1 various forms of the VRP relevant to this research are provided in order to increase the understandability of this thesis assignment. Section 2.2 discusses some solving methods for vehicle routing problems and dives deeper into the definition of a TS. Finally, in Section 2.3 four articles that present a solving method that includes the TS for the HVRPTW will be reviewed. One of these articles will be selected and used in Chapter 5 to extend the model such that it is able to distinguish between heterogeneous vehicles. Finally, the research questions are answered in Section 2.4.

2.1 Various forms of a Vehicle Routing Problem

In this section, an overview of various forms of a VRP is provided. Firstly, the classical VRP model will be explained and afterwards relevant extensions of the VRP for this research are described. The extensions are selected from the taxonomy of Lahyani, Khemakhem and Semet (2015) which can be found in Appendix A-1.

2.1.1 Classical VRP

The vehicle routing problem appeared first in the paper of Dantzig and John Ramser (1959) and it is a generalization of the Traveling Salesman Problem (TSP) which is described by Noon and Bean (1993). In the classical VRP model, a fleet of identical vehicles supplies the customers starting from a depot where the resources are stored. Each customer has a certain demand that must be satisfied. In addition, a cost matrix is defined that stores the associated costs from traveling from a customer to another customer. The aim of a VRP is to optimise the routing design thusly that each customers' demand of goods is satisfied without violating any problem-specific constraint, in this case, the maximum capacity of the vehicle cannot be exceeded (Caceres-Cruz et al., 2014). In Figure 2, a solved example of the classical VRP with three different routes, indicated by the colours, is provided where each circle represents a customer and the box in the middle is the depot. Furthermore, the numbers along the arcs represents the costs from the cost matrix. The demand of each customer is denoted as the black numbers within a circle.

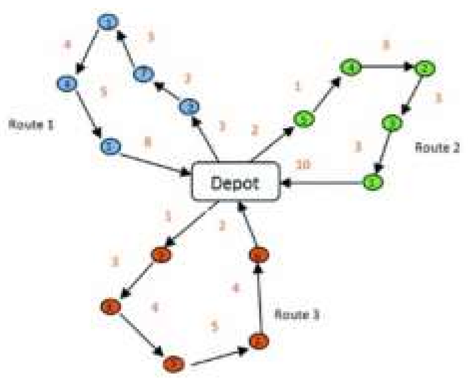


Figure 2: Representation of the classical VRP by Dixit, Mishra & Shukla (2019)

There exist many extensions to this classical VRP depending on the parameters and constraints considered and some of them will be presented in the next Sections. The classical VRP model and all its variants that include extensions, are NP-hard. Consequently, the optimal solution to the VRP cannot be guaranteed in polynomial time according to Lenstra & Kan (1981).

2.1.2 Input data

As indicated by Table 1, based on the input data the VRPs could be taxonomized into four categories. The evolution of information indicates whether the information provided to the planner can change while executing the routes, for instance, with an arrival of a new customer request. Whenever the VRP allows the possibility of changing some problem parameters during the execution of the routes, the input data is considered dynamic. On the other hand, if the vehicle routes are not adapted once they started, the input data is considered static.

The quality of the information relates to the potential variability of the input data, think of the travel time that can take longer than expected due to a traffic jam. Deterministic routing problems assumes that the parameters such as demand and service time are known with certainty. Stochastic data assumes that the parameters are related to probability distributions and thus that the input data is uncertain. The three most common stochastic parameters studied in the literature are: customers' demands, service times and travel times (Lahyani, Khemakhem & Semet, 2015).

		Information quality	
		Deterministic input	Stochastic input
Information evolution	Input known beforehand	Static and deterministic	Static and stochastic
	Input changes over time	Dynamic and deterministic	Dynamic and stochastic

Table 1: Taxonomy of vehicle routing problems by information evolution and quality (Pillac et al., 2013)

2.1.3 Time related constraints

In many examples, delivering the goods or providing the service takes some time. In these cases, the length of the service, service time, is added as a parameter of the vehicle routing problem.

The VRP with time windows (VRPTW) impose that the service should start and end within a predetermined interval by the customer. There are two variants of time windows, namely soft and hard time windows (Boujlil & Elhaq, 2020). If a customer has soft time windows, then the time windows may be violated by the vehicle. However, this violation of the time window brings penalty costs with them. In case, a customer has hard time windows, then the time window constraint cannot be violated. Whenever a vehicle arrives after the end of the time window, the vehicle cannot deliver its goods and should leave the customer's site. Moreover, if the vehicle arrives before the start of the time window, the vehicle should wait until the time windows allows the start of the service. Sometimes, the waiting time is penalised with waiting costs since the vehicle and the truck driver are idle while waiting.

2.1.4 Vehicles related features

Another extension of the simplified classical VRP is the usage of heterogeneous vehicles (HVRP) instead of homogeneous vehicles. Homogeneous vehicles are identical thus have all the same capacity, maximum driving speed, height etc. In case, heterogeneous vehicles are used in the VRP, all these characteristics could diverge between the trucks.

Furthermore, the VRP has a variant that allows the use of multiple trips by the same vehicle during the planning period as long as every constraint is respected, contrary to the routing problems that only allow a vehicle to perform one single trip.

Finally, there is an extension that permits a customer to be served by multiple vehicles (load splitting) contrary to the classical VRP where each customer must be served by only one vehicle. The specific form of a VRP that allows the possibility of multiple visits to the same customer is called a VRP with Split Deliveries (SDVRP).

2.1.5 Depots

In the classical VRP model, a single depot is used in order to serve every customer. However, in real-life applications, companies often have multiple depots and different starting and final locations for vehicles (Lahyani, Khemakhem & Semet, 2015). Sometimes, the customers are already allocated to the appropriate depot and sometimes this has still to be done in the VRP. Furthermore, the characteristics of the depots may differ regarding their capacities, locations and associated costs. This extension of a VRP is called a Multi Depot Vehicle Routing Problem (MDVRP). This specific problem is addressed in Cordeau, Gendreau & Laporte (1997). The extension with heterogeneous vehicles (MDHFVRP) instead of homogeneous vehicles is described in Salhi, Imran & Wassan (2014). Finally, the Multi-Depot Cumulative Capacitated Vehicle Routing Problem (MD-CCVRP) is solved in Lalla-Ruiz & Voß (2020) using a matheuristic approach where the objective is to minimize the sum of arrival times at customers for providing service.

2.2 Solving methods

There are many ways to solve a vehicle routing problem and, in this section, a few solution methods will be shortly focused on. As can be obtained in the taxonomy of solving methods to VRPs designed by Goel et al. (2019), there are three main approaches to solve a VRP, namely exact methods, approximate methods and hybrid methods. In Section 2.2.1, there will be elaborated on the exact methods, followed by the heuristics in Section 2.2.2 and metaheuristics in Section 2.2.3. Moreover, in Section 2.2.3.1, a closer look will be taken to a specific metaheuristic, the Tabu Search (TS). As mentioned before the current model of Districon uses a TS to solve the routing problems, therefore the choice has been made to dive deeper into the TS and generally discuss other solving methods. Finally, a description of matheuristics is presented in Section 2.2.3.2.

2.2.1 Exact Methods

As the name already suggests, exact methods guarantee that their solution is optimal. These methods are mainly applied to small-size VRPs (Jourdan, Basseur & Talbi et al., 2009). Exact algorithms are able to solve Traveling Salesman Problems containing hundreds or thousands of vertices, however these exact methods cannot solve VRP including more than 135 vertices (Vidal et al., 2012). “In the literature, families of exact methods are based mostly on integer linear programming, branch-and-bound or branch-and-cut algorithms or also on dynamic programming.” (Boujlil & Elhaq, 2020). An overview of these exact methods is presented in Semet, Toth & Vigo (2014).

2.2.2 Heuristics

In case finding an optimal solution is impossible or very time consuming, a popular approach to solve the problem is to use heuristics. “A heuristic, or a heuristic technique, is any approach to problem-solving that issues a practical method or various shortcuts in order to produce solutions that may not be optimal but are sufficient given a limited timeframe or deadline.” (Chen, 2021). Heuristics are as old as the VRP and since then many heuristics have been proposed, either constructive and improvement heuristics. Constructive heuristics are used to produce an initial solution and improvement heuristics are utilized to improve existing solutions. Examples of heuristics can be found in Laporte et al. (2000).

2.2.3 Metaheuristics

The term metaheuristic was introduced by Fred Glover and its definition is as follows: “A meta-heuristic refers to a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality. The heuristics guided by such a meta-strategy may be high-level procedures or may embody nothing more than a description of available moves for transforming one solution into another, together with an associated evaluation rule." (Laguna, 2000). Metaheuristics are developed to find a "good enough" solution in a "small enough" computing time. Contrary to the exact methods, where a proof is provided that the optimal solution will be found in a finite amount of time, although this can be prohibitively long for VRP with many vertices (Glover, 2015). According to the taxonomy of Goel et al. (2019), metaheuristics are distinguished in two types: trajectory-based and population-based. This distinction has been made according to the number of solutions used at the same time. Trajectory methods iteratively make small changes to a single solution and population-based algorithms iteratively combine solutions from multiple initial points into new candidate solutions. An overview of the different sorts of metaheuristics is given by Vidal et al. (2013).

2.2.3.1 Tabu search

The Tabu Search (TS) is a trajectory-based metaheuristic that can be viewed as an improvement method and was introduced by Fred Glover in 1986. In the book of Burke & Kendall (2014), the TS has been described and explained. TS prevents an algorithm or model to be stuck in a local optimum by allowing non-improvement moves and preventing it to visit a recently visited solution by use of a Tabu List (TL). The TL stores recently visited solutions by the use of adaptive memory structures. Solutions that are included in the TL cannot be visited, in this way cycling back to recently visited solutions is not possible. The length of the TL decides how long a certain solution stays 'tabu'. "At each iteration of TS, the local transformations that can be applied to the current solution, denoted S, define a set of neighboring solutions in the search space, denoted N(S)." (Burke & Kendall, 2014). The best or least bad solution of this neighborhood set will be selected. In some cases, the TL is too powerful such that it may prevent attractive moves to be made, while there is no distress in being stuck in a local optimum. For this reason, aspiration criteria have been introduced, whenever this criterion is met, the TS can make use of the forbidden solution on the TL. The most commonly used aspiration criterion allows a tabu move in case that it results in a better solution than the current best-known one. Furthermore, a termination criterion should be defined otherwise the TS will run forever. This can be done after a fixed number of iterations, after reaching a pre-specified objective value or after a specific number of consecutive iterations without improving the objective value. Moreover, to enlarge the search space, infeasible solutions could be allowed by violating some constraints and compensate the violations with penalties. Finally, the memory usage of the TS from short term to long term could be used to implement search strategies for intensification and diversification. "Intensification strategies reinforce move combinations and solution features historically found good, while diversification strategies drive the search into new regions." (Glover, 1990).

2.2.3.2 Matheuristics

Recently, matheuristics became popular among researchers that are interested in solution generation for vehicle routing problems. The name 'matheuristic' is a combination of mathematical programming and metaheuristic. Matheuristics make use of both exact and metaheuristic techniques such that competences of both methods could be jointly utilized. "The integration of exact approaches within a metaheuristic or vice-versa can lead to higher computational times than heuristic methods, but may also lead to a better performance robustness and quality of the solutions in some applications." (Kramer et al., 2019). The papers of Lalla-Ruiz & Voß (2020) and Kramer et al. (2015) present the application of matheuristics for solving vehicle routing problems.

2.3 Review on papers solving HVRPTW

In this section, a review will be provided on four literature papers that present a solving method including the TS for a Heterogeneous Vehicle Routing Problem with Time Windows. These four papers have been selected since they were the only ones that fulfilled every requirement of the search. The papers needed to propose a solving method for the Heterogeneous Vehicle Routing Problem with Time Windows, preferably hard time windows however this is not necessarily. Furthermore, the solving method should include a Tabu Search algorithm. Lastly, no incompatible demand constraint should be included in the model assumptions.

The articles will be discussed in the same manner. Firstly, a small introduction about the case of the paper will be provided. Afterwards, the algorithm will be explained. Finally, the performance of the algorithm will be discussed.

2.3.1 A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows

The first article that will be reviewed is the article from Paraskevopoulos et al. (2007). They used a two-phase approach to solve the HVRPTW.

In the first phase, a few initial solutions are designed using a semi-parallel construction heuristic. The initial solutions are constructed based on the Average Cost per Unit Transferred ($AKUT_k$) , where k stands for a particular vehicle type. $AKUT_k$ is expressed as the ratio of the total travelling time plus the fixed costs over the demand carried by vehicle k. In case customers are not assigned to a route, additional vehicles will be created to make sure that every customer is 'served'.

To reduce the number of vehicles used in the different solutions and increase the capacity utilization of the vehicles, a route elimination method is employed. This route elimination method is based on the Ejection Chain (EC) framework and the basic ideas of IR-insert, which is an intelligent reordering mechanism. Finally, some high-quality solutions are selected to proceed to the second phase for further improvement.

In the second phase, a Reactive Variable Neighborhood Tabu Search (ReVNTS) hybrid metaheuristic method is used to minimize the objective value. A solution s' in the yth neighborhood of the current initial solution s is randomly selected. The TS will generate iteratively new solutions. In case a generated solution is better than the current solution, the neighborhood index y is reset to 1 and a new random solution s' will be selected. This procedure stops when all possible neighborhood structures have been examined, then the best solution found will be stored. Afterwards, the ReVNTS will be applied to another initial solution, generated in the first phase, until all selected initial solution has been examined. The best solution out of the stored final solutions of each initial solution is the outcome of the model. This procedure has been visualized in Figure 3.

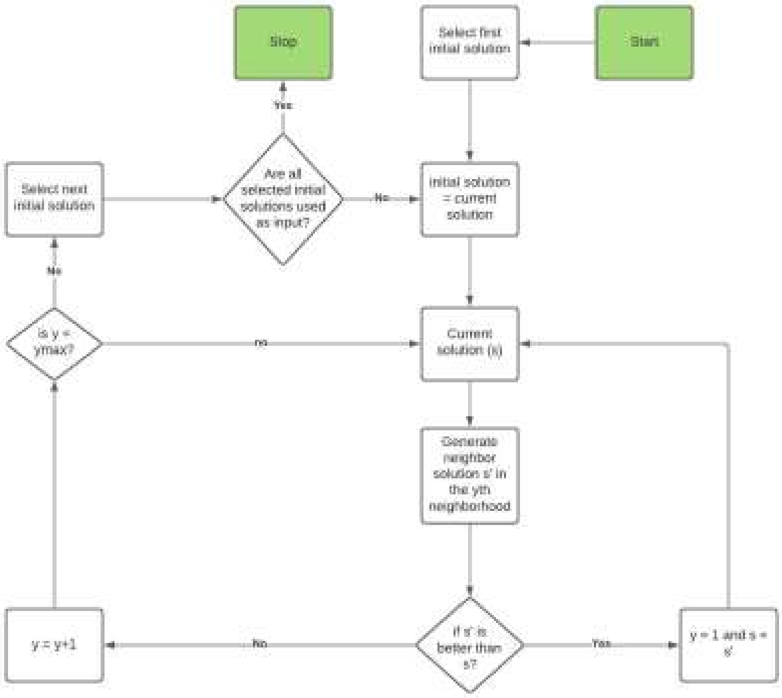


Figure 3: Flow chart ReVNTS of Paraskevopoulos et al. (2007)

This solution method has been applied to the datasets of Liu and Shen (1999) and provided high-quality solutions. These datasets contain 100 vertices. The datasets differ in the way how the coordinates of the customers are generated and the length of the time windows of the customers. For each data set, a different subclass of heterogeneous vehicles is defined. The proposed algorithm has been applied to these different datasets and compared to the algorithm of Liu and Shen. The average reduction of the ReVNTS over all datasets compared to the algorithm of Liu and Shen is 12.69%.

2.3.2 Vehicle routing problem with a heterogeneous fleet and time windows

The second article of Jiang et al. (2014) uses a two-stage Tabu Search heuristic without having to rely on a construction heuristic. In order to produce an initial solution, the customers and vehicle are sorted. In the article, multiple possible sorting rules can be found. In the algorithm, the greatest distance rule and the greatest capacity rule will be used to sort the customers and vehicles. The greatest capacity rule is used such that in phase 2 the usage of larger vehicles can be split up into smaller vehicles to improve the solution. Furthermore, this algorithm makes use of a holding list which is a “phantom” route. In phase 1, there are five possible moves for the neighborhood of a solution:

- Customer is moved from a route to another route
- Customer is exchanged with another customer of another route
- Customer is transferred from holding list to a route
- Customer is moved from a route to a holding list
- Customer is exchanged with another customer in holding list

When producing the initial solution, the holding list is filled with every customer based on the sorting criteria mentioned above. The TS will divide the customers over the first vehicle by iteratively running one of the five above mentioned moves until an x number of non-improving iterations occurs. In that case, a new vehicle will be added to the solution set which is then allowed to be filled by customers. This procedure will go on until either the maximum number of vehicles has been reached or the holding list is empty. Consequently, the initial solution has been produced. This completes phase 1 of the

proposed solving method.
Phase 2 is a post-processing procedure that includes a TS that is defined to split the usage of larger vehicles into smaller vehicles in two ways. Firstly, a part of a route can be transferred to a smaller empty route if the vehicle is not fully loaded. Secondly, a route could be split into two smaller empty routes. For both procedures, the order of the customers remains unchanged. These two procedures are executed until an x amount of consecutive non-improving iterations arise. This concludes the whole algorithm and the best solution found so far will be the final solution of the proposed solution method of this paper.
Finally, this algorithm has been applied to the dataset of Liu and Shen and compared to the MSDA Medium algorithm presented by Bräysy et al. (2008). The average computing time of Bräysy is 211 seconds and the average computing time of the proposed algorithm by Jiang et al. is 32 seconds. The algorithm of Bräysy presents at almost every case a better result, however the average deviation compared to the algorithm of Jiang et al. is only 0.84%. Consequently, this relatively simple solving method shows that in a very short computing time, decent solutions could be obtained. Finally, in the conclusion part of the paper is stated that the searching process is sometimes not able to avoid the local optimum.

2.3.3 An ACS-based memetic algorithm for the heterogeneous vehicle routing problem with time windows

The third paper that will be reviewed in this section is the article of Molina, Salmeron and Eguia (2020). They propose a hybridized Ant Colony System (ACS) with a local search that is performed by a Variable Neighborhood Tabu Search (VNTS).
The initial solution starts with randomly choosing a vehicle and inserting the furthest customer into the route. At each solution step, a probability function represents the probability to a specific customer to be inserted between two other customers. This probability function takes the attractivity of the demand requested by the customer and the time window into account. The customer with the highest probability is inserted in the route in most instances. Finally, the customers that are not included in a route, are put on the holding list.
The initial solution will be improved by the VNTS that uses seven neighborhood structures. The Tabu Search (TS) starts iterating in the first neighborhood until the maximum iterations for a neighborhood have been reached. Similarly, this procedure will apply for the other neighborhoods. However, in case a new best solution has been found the TS returns to the first neighborhood where the new best solution will be used as starting solution. This process will go on until a local minimum solution for all the neighborhood structures is reached. The size of the TL is dynamic and has a lower and upper value. This enables the ability to use diversification and intensification in the search strategy. After each iteration where no improvement is observed, the size of the TL is increased by one unit up to the upper bound value. In case a new best solution has been detected, the size of the TL resets to the lower bound value.
This solution method shows high quality results on the instances of Paraskevopoulos et al. and Jiang et al. that are described in previous sections. On the latter, this algorithm improved 55 out of the 56 solutions and 10 out of the 24 instances of Paraskevopoulos et al. are improved while 5 solutions matched the best solution found so far in the literature. This shows that this algorithm provides feasible solutions of good quality to all types of HVRPTW instances.

2.3.4 A granular tabu search algorithm for a real case study of a vehicle routing problem with a heterogeneous fleet and time windows

In the article of Bernal et al. (2017), a two-phase heuristic algorithm including a granular Tabu Search (MGTS) is presented.
The first phase consists of the generation of the initial solution that will be improved in the second phase by a granular TS algorithm. To create the initial solution, two-node routes are created with the depot added twice to the route as a starting and ending point. The vehicles are assigned to the routes

created sorted on the capacity. Thus, the vehicle with the highest capacity is assigned to the first route. The algorithm inserts as many customers as possible while ensuring that the parameters as maximal length of a route, maximum capacity of a truck and the time windows constraints of the customers are not violated. Whenever some customers are not added to a route at the end of the process, the tolerance threshold of the parameters will be increased.
After the initial solution has been found, in the second phase, the granular TS is applied in order to improve the solution until a maximum number of iterations are executed. The MGTS considers infeasible solutions regarding the violation of the parameters mentioned above by compensating them with penalty costs. These penalty costs are calculated by penalty factors that are updated dynamically. Furthermore, a dynamic parameter, beta, allows the algorithm to alternate between diversification and intensification stages. The MGTS algorithm will be applied until a local optimum has been found. For this local optimum, the algorithm iteratively moves from a solution to the best solution found in the neighborhood, feasible or not. This move is set tabu for a random integer between a pre-defined lower and upper bound.

The application of the granular TS on the dataset, provided by the company wherefore this algorithm is constructed, shows some interesting results. However, on some occasions, the current solution of the software of the company finds solutions that are much better than the solution provided by the algorithm of Bernal et al. (2017). Out of the 12 cases, the algorithm was only able to outperform the current solution on 5 occasions. In many cases, the number of vehicles used by the algorithm is higher than the real solution. The fleet sizes are varying from 7 to 16 vehicles depending on the case.

2.4 Conclusion

After the literature study has been conducted, the answers to the research questions mentioned in the introduction of this chapter could be answered.
Section 2.1 described what a Vehicle Routing Problem (VRP) is and some relevant extensions have been explained. In a VRP, a fleet of vehicles departs from a depot and supplies the customers allocated to it. Each customer has a certain demand that should be satisfied. Examples of extensions of the VRP are the different sorts of input data, in- or exclusion of time windows and the usage of homo- or heterogeneous vehicles. Most of these extensions could be combined into one complex form of a VRP. The problem considered in this research is a heterogeneous vehicle routing problem with hard time windows including static and deterministic input data.
In Section 2.2 is shown that exact algorithms are able to guarantee the optimal solution to small and simple vehicle routing problems although this could be very time consuming. For problems that need to be solved relatively quickly or more complex problems, (meta) heuristics could be used as they will find a good, not optimal, solution within a short computing time. Metaheuristics iteratively make small changes to an initial solution. An example of a metaheuristic is TS that is used to prevent the algorithm to be stuck in a local optimum. This is done by putting recently visited solutions on a TL. Meaning that this solution cannot be visited until an aspiration criterion is met or it is removed from the TL. Moreover, another example of a metaheuristic is the matheuristic. Matheuristics make use of both exact and metaheuristic techniques such that competences of both methods could be jointly utilized.
Four papers that provide a solution method for the HVRPTW including a TS algorithm are reviewed in Section 2.3. The proposed solution method of Molina, Salmeron and Eguia (2020) has improved almost all best-known solution in literature of the Jiang et al. instances and improved 10 out of the 24 best known solutions in literature of the instances of Paraskevopoulos et al. (2007), thus it outperforms the proposed approach of Paraskevopoulos et al. The paper of Jiang et al. (2014) produces mediocre solutions in a very short computational time, however, this is not the objective of Company A because they want high quality results. Finally, the article of Bernal et al. (2017) did not present high quality solutions to the instances of the company for which the algorithm was constructed. Moreover, in many cases, the number of vehicles used by the algorithm is higher than the real solution. Company A desires to maximize truck capacity utilization and to minimize the number of trucks used, thus this algorithm

is not useful for the model of Districon. For these reasons, the TS algorithm of Molina, Salmeron and Eguia (2020) will be used to extend the model of Districon for the application to the case of Company A. Besides, it is a recent paper thus the solving method is constructed based on the newest findings in research.

3. Context Analysis

The aim of this chapter is to create a better insight of the context of the problem. In Section 3.1 the reasoning behind the choice of Company A to have a model that automatically schedules their trips is reported. In Section 3.2 the relevance of the model from the perspective of Districon is elucidated on. Furthermore, in Section 3.3 the problem definition of the specific form of the VRP corresponding to the PoC is formulated. Section 3.4 provides a thorough analysis of the data provided by Company A. Furthermore, in the Section 3.5 the most important Key Performance Indicators (KPIs) are discussed. Moreover, in Section 3.6 potential alternative applications of the model for Company A as well as for Districon will be discussed. Finally, requirements that should be included in the new model are listed in Section 3.7 which concludes this context analysis and answers the first research question:

> *What is Company A exactly demanding from Districon and what is Districon's intention with respect of the model?*

The information provided in this chapter has been gathered by interviewing employees of Districon and studying the data that Company A sent. Qualitative data about Company A has been obtained by interviewing an employee of Districon who is in direct contact with Company A. The quantitative data of the model has been gathered by speaking to one of the designers of the model. Finally, the information about the future interest of Districon in the model has been acquired by a conversation with the leader of innovation projects to where this model concept belongs to.

3.1 Motivation Company A

In this section, in-depth information about Company A is provided especially about why the model is so important to Company A and some key features which cannot be ignored in the model.

3.1.1 Current planning process

Currently, every route is planned by human planners and the customers set with their demand is relatively similar every day. Consequently, the planners use the route of the day before and implement the contingent change in the customers set. However, this can get quite complicated when a new customer is added near a cluster of a trip whose truck is already completely loaded. In this case, the planners should adapt multiple trips in order to find a feasible routing schedule which is fairly time consuming. Additionally, formerly when a new depot was opened, it took the human planners plenty of time to design the route schedule for this concerned depot. Occasionally, last-minute changes could be observed. For example, whenever a customer is not able to pick up the delivery at the agreed-upon time for some reason. This implements that the time window of a customer is changed, consequently the route must be adapted. This implies that running the model should not take hours otherwise last-minute changes could not be included in the RPS. The maximum running time of the model is set to 15 minutes by Company A. A Route Planning Software (RPS) will resolve these issues since the data set could be updated and implemented in the model that will find a feasible and efficient solution in a relatively short time. A RPS is a system that automatically plans a routing scheme that involves the lowest costs based on the input data provided.

3.1.2 Solution approach Company A

For the mentioned reasons in the previous section, Company A is planning to apply a Route Planning Software (RPS) to their VRP in the last mile. Company A already orientated on software that could automate their planning of the trips. They experienced Software 1, however, this operating system was too advanced. Software 1 is more or less a Transportation Management System (TMS). A TMS records the whole flow from order to delivery and for example sends confirmation emails to customers. Therefore, Software 1 includes boundless unnecessary features, resulting in that this software is too expensive to operate with for Company A. Moreover, Company A tested Software 2 but this tool was not advanced enough. The outcome of this model still needed manual adaptions to obtain realistic solutions. Company A needs something in between the previous two mentioned software, a boutique solution. A boutique solution is a tailor-made or custom-made solution to the particular problem. Therefore, they asked Districon to come up with a Proof of Concept (PoC). Districon received a dataset concerning the PoC that contains all the data for one region. However, If Districon receives the assignment to automate the routing planning, the model needs to produce routing schemes for all the regions. Information about the dataset will be provided in the fourth section of this chapter.

Company A desires the model to find the solution with the most optimized load capacity of the trucks in other words the number of trucks utilized should be minimalised. The objective is to obtain a higher truck utilization.

3.2 Districon

In this section, more information on why this model is important to Districon is provided. In Section 3.2.1, it is shortly explained how this project came about and the ambitions of Districon with the model are described in Section 3.2.2.

3.2.1 Innovation projects

Since last year, Districon is focussing more and more on innovation projects. Innovation projects are executed without a direct order of a customer that anticipates on future needs of companies. These days, Districon is structurally investing in innovation projects, especially at times where some employees are idle, contrary to back in the days where occasionally time was spent on innovation projects. Above all, the customers remain the major centre of attention. This VRP model is a running innovation project and at the same time it is also focused on a case of a client, namely Company A.

3.2.2 Future application of the VRP model

The objective of this specific innovation project of Districon is that this VRP model should be a quality foundation for a RPS which could be used for many different cases. Districon spends much time in network studies regarding finding out what the best location is to build a warehouse for a company depending mostly on their customers locations. Districon starts the analysis with building a model that represents the current routing schedule, where every single trip of the last months are included. The outcome of the model, the total costs, will be compared with the actual realised costs to check whether the model fits the network well. Whenever the model outcome is similar to the actual outcome and the transportation network is represented well in the model, the search for the new location of a warehouse can start.

3.3 Problem definition

After the needs and requirements have been identified, the vehicle routing problem including their constraints and objective is described.

3.3.1 VRP description

The future problem of Company A and the new desired model of Districon describes a VRPTW with heterogeneous vehicles. This problem will be discussed in chapter 5. However, the problem of customer A provided in the PoC is a Vehicle Routing Problem with Time Windows (VRPTW). The representation of a VRPTW given by Cordeau et al. (2001) is used as a guideline to define the VRPTW. A VRPTW is defined on a graph $G = (V, A)$ where $V = \{v_0, v_1, \dots v_n\}$ denotes the vertex set. This set represents every location of the dataset, where v_0 stands for the depot and the remaining vertices for the customers. At the depot is a fleet of m vehicles deployed. The arc set $A = (\{v_i, v_j\},\ v_i, v_j \in V, i \neq j)$ denotes direct connections between the depot and the customers and among the customers. For each arc (v_i, v_j), nonnegative costs c_{ij} and traveling time t_{ij} are associated. For every vertex $v_i \in V$, a nonnegative demand q_i is requested ($q_0 = 0$, since the depot has no demand), a nonnegative service time s_i and a time window $[e_i, l_i]$, where e_i and l_i are nonnegative integers. e_i stands for the opening of the window and l_i denotes the end of the time window. These are hard time windows, implementing the inclusion of waiting costs w_i whenever a vehicle arrives before the opening of the time window. The maximum duration and capacity of a route k are denoted as D_k and Q_k.

3.3.2 Constraints

The VRPTW explained in the previous section contains the definition of a normal VRPTW, however the complete description of a certain vehicle routing problem differs per case. Consequently, the following constraints have been formulated to fulfil all the requirements and desires of Company A:

- Maximum capacity of trucks cannot be violated
- Every customer has to be served within its time window
- The depot is the start and end location of every trip
- Every customer has to be served at once so load splitting is not allowed, i.e., every customer is included in only one trip
- Total duration of a single trip does not exceed the maximum duration of a trip
- Trucks start loading and departing from 6 am
- Trucks completes their last delivery no later than 6 pm
- Trucks' capacity utilization should be optimized

3.4 Data

In this section, the data of one region provided by Company A for the PoC is analysed. Starting with appointing the parameters mentioned in the VRP description (Section 3.4.1). Followed by a map which displayed all the locations of every customer of Company A (Section 3.4.2). In Section 3.4.3 the demand of the customers is analysed. Finally, the time windows and service times of the customers are examined in Sections 3.4.4 and 3.4.5.

3.4.1 Model parameters

Table 2 shows the model parameters, introduced in the previous section, that are specific for the POC data of Company A. The input data is static and deterministic (see Section 2.1.2).

Model parameter	**Value**
Number of depots	1
n (number of customers)	n (a bit more than 100)
m (number of vehicles)	8
Types of vehicles	1
Qk (maximum capacity of the vehicles)	Qk (about 30.000 kg)
Dk (maximum duration of a trip)	12 hours exclusive trip from last stop to depot
Wi (waiting costs per hour)	0,5
Maximum number of stops per trip	Unlimited

Table 2: Model parameters

For the first mile (direct or milk run) from plant to depot, a truck with a net weight capacity of about 80.000 kg is utilized contrary to the last mile where reefer trucks with a net weight capacity of about 30.000 kg are utilized to serve the customers. Since the unit of the capacity of trucks and demand is not specified in the data, we have assumed that the demand is denoted in kg. This assumption holds for the whole report. In the dataset of the PoC, there is only one vehicle type but in the future Company

A will utilize more than 7 different types of trucks to obey the clients. The heterogeneous datasets are introduced in Chapter 6. There is no maximum number of trip stops defined since the capacity constraint will make sure that a vehicle will not have to stop plenty of times. As mentioned in Section 3.3.2, trucks start loading and departing at 6 am and must complete their last loading activities at 6pm. Consequently, the maximum duration of a trip equals 12 hours plus the traveling time from the last stop back to the depot. Additionally, Company A allocates costs if a vehicle arrives before the opening of the time window of a customer, a penalty cost of 0,5 per hour will be included to compensate for this idle time. Furthermore, for every driven kilometre a cost of 0,001 will be considered. Moreover, the involved cost for every working hour is equal to 1. Finally, the set-up costs per vehicle are 100. Note that the costs have no currency since this is unknown, consequently the objective value will not have a unit either.

3.4.2 Customers map

In this section, an overview of the customers of Company A is given on a map. The map denoted in Figure 4 is a heat map where the small black circles denote a location of a client. The depot has been visualized by a red triangle, that is indicated by the red arrow painted at the map. The amount of heat is represented by the total demand in that specific area, the more demand the more heat is visualized in that specific area. The amount of heat is represented by the legend of the right of the graph. The more demand in a region the redder the colour is in the graph. From Figure 4, it can be obtained that most customers of Company A are located near the depot. However, there is one major outlier in the north of the region. Furthermore, there is a small cluster of customers in the south of the region.

Figure 4: Heat map of the customers in the region

3.4.3 Demand

As indicated before in this specific region, Company A serves n customers in total which are all demanding products from Company A. The amount of demand in kg distributed per store is displayed in Figure 5. The demand is varying from 100 kg to 5.000 kg which are respectively the smallest and largest demand. The total demand of the n customers is about 180.000 kg which implements that the average demand requested by a customer of Company A is about 1.000 kg. This means that 6 trucks would be needed to serve every customer in case the load could be perfectly spread over the trucks, without considering time window restrictions.

Figure 5: Demand per customer/store

3.4.4 Time windows

The start and end of every customer's time window can be obtained in Figure 6, where the blue line represents to the start of the time window and the orange line to the end of the window. The left vertical axe corresponds to the unit of the time windows. From Figure 6 can be concluded that the time windows are relatively large which results in a less complex problem since the customers are flexible. The average time window length is about 8 hours, which is equal to almost a whole working day. Furthermore, the largest time window length is 10 hours and the shortest is equal to 3 hours which is still respectively long. From the Table 3 and Table 4 in Appendix B-2 can be obtained that most time windows start at 7 am and end at 6 pm. Moreover, in Figure 6 the demand for every customer is also specified by the grey bars which corresponds to the right vertical axe. Finally, it can be concluded from Figure 6 that the customers who have a relative short time window also have a relatively low amount of demand, so the short time windows should not cause much trouble.

Figure 6: Demand in the time windows

3.4.5 Service time

The last parameter of the model which has not been discussed in this chapter is the service time. From Table 3 can be obtained that the service times vary between 5 and 15 minutes. Furthermore, it can be concluded that the total service time is about 1400 minutes and the average service time is about 10, minutes. The duration of the service time looks relatively short on the surface. However, the total service time is about 1400 minutes divided over 8 trucks which is on average 180 minutes per truck driver. This means that on average at every route, the driver is about 3 hours busy with unloading products.

[redacted]

3.5 Key Performance Indicators

In this section, the key performance indicators (KPIs) used by Company A are discussed. A KPI measures the performance of a company in business activities. The KPIs used by the company can be classified as follows:

- **Total costs (objective value)**
  This KPI is calculated by summing up the driving costs, set up costs, working hour costs and waiting costs of every trip.
- **Truck capacity utilization (%)**
  The truck capacity utilization is calculated by dividing the total demand by the total capacity of the trucks used. For instance, if 5 trucks with a capacity of 500 kg are used to serve the customers that have a total demand of 400 kg. The truck capacity utilization is 80%.

These two KPIs have been selected since Company A explicitly mentioned that they would like to reduce their costs and improve their truck capacity utilization. From the two objectives, the cost related KPI is the most important one since in the end reducing costs is the main goal of Company A. However, these two KPIs are closely related to each other, since improving the truck capacity utilization results in needing less trips to serve the customers, which results in less costs.

3.6 Alternative applications of the model

Company A is considering many innovation projects nonetheless initially it needs to be identified whether a specific project is genuinely worth it. The model can be used for innovative projects that are related to the last mile to indicate the impact on the routes and whether it is profitable or not. Company A is examining opening a new depot and changing some allocations. These innovations could relatively easily be analysed by adapting the dataset that contains the data about locations and running the model frequently with various new depot locations and different allocations to the depots. In this way, Company A could analyse if a certain change is beneficial.
Furthermore, the model could be used for determining whether it is more desirable to utilize fewer trucks with higher capacity or using more trucks with less capacity. Moreover, various scenarios could be analysed with help of the model. For example, what if the time windows of the customers change or what if Company A exploit the evening as well to distribute their products.

3.7 Conclusion

To conclude, a list of requirements that the new model (RPS) needs to fulfil has been made regarding the dialogue with the experts of Districon and indirectly Company A:

- Maximise truck capacity
- Deliver within the time windows
- Serve every customer at once unless demand exceeds maximum truck capacity (not the case in this dataset)
- Feasible solution
- Relatively short running time (max 15 minutes)
- Ability to use heterogeneous vehicles

Value of RPS for Company A

All in all, the model could be of a high value to Company A since innovations and scenarios could be analysed rapidly. Besides, the model can handle dynamic customers sets in a superior way to the human planners. Finally, in case a new depot is opened, as well abroad as domestic, the RPS could construct the routes instantaneously and more efficient than currently, the human planners are performing.

4 Current model

In this chapter, the current model of Districon will be analysed. Since this model will be the foundation of the extended model, understanding the model is indispensable. In Section 4.1, general information about the model is provided. The algorithms used in the model will be detailly analysed in Section 4.2. Finally, Section 4.3 summarizes the full analyse of the current model of Districon.

4.1 General information

In this section, the origin of the model will be discussed. This model is not created from scratch, but this model was constructed in purpose of a former project. The information about this project has been obtained by interviewing the team-leader of this former project.

4.1.1 General description

The model of Districon was initially developed a year ago for another project containing a vehicle routing problem. The constructed algorithm has been programmed in Python. This model was firstly partially programmed in AIMMS, however, the solvers of AIMMS are constructed to solve to optimality. However, in this case, it is too complex since this VRP is a NP-hard problem thus the problem could not be solved to optimality in polynomial time. For this reason, the model has been programmed in Python because it is able to solve the problem quicker since this model does not solve to optimality.

4.1.2 Objective of the former project

In this project, the goal was to analyse the influence of certain scenarios of the company whether they increased or decreased the costs. The scenarios were designed based on two input values, namely the time windows and classifications of the shops. Shops are classified depending on their lead time. Based on the lead time, time windows are assigned to the shops. In the scenarios, different classification rules were used in combination with different time windows. For example, the classification was based on the volume of the store and the stores could be supplemented within the corresponding opening hours of that classification. In this way, many different scenarios have been run. The objective was to economize on the total driven distance, total working hours and to improve truck capacity utilization.

4.1.3 Constraints

The trucks of this company can only serve 2 or 3 customers per truck due to the large orders contrary to the trucks of Company A that are able to serve 15 to 25 customers. Furthermore, the stores were already allocated to the multiple warehouses and this grouping could not be changed. Unfortunately, this resulted in less room for improvement since not that many swaps could be made. Since the allocation of customers to warehouses could not be changed, a single depot algorithm was constructed instead of a multi depot algorithm.

4.1.4 Difference between former project and the case of Company A

This model has been modified and extended for the case of Company A such that it perfectly fits the data set of the proof of concept. The model is now able to solve a homogeneous single depot Vehicle Routing Problem with Time Windows (VRPTW). The description of this specific form of a VRP is provided in Section 3.3.

A major difference between the former project and the case of Company A is that the trucks in the former project were only able to serve 2 or 3 customers, while trucks of Company A could easily serve 20 customers. Furthermore, Company A uses no classification per customers and the customers are not related with each other. Moreover, the dataset that Company A provided for the PoC is not variable, while in the former project some variables could be experimented with.

4.1.5 Input and output data

The input data such as the model parameters and the customers' locations with their corresponding demand are imported via csv files. The dataset of Company A only contains coordinates. Consequently, Districon used OpenStreetMap with truck profiles to calculate the distances between the vertices. In OpenStreetMap there is the possibility to customize the truck profile by adding e.g., the height and weight of the truck and many more characteristics that could be implemented. Consequently, OpenStreetMap can calculate the best route for every truck type of Company A if all characteristic and desires are known. The proposed solving method for a VRPTW consists of generating an initial solution that will be improved in the second phase by means of a TS algorithm. In the next section, the construction of the initial solution as well as the functioning of the TS will be thoroughly investigated.

4.2 Algorithms explanation

The algorithms that Districon constructed for solving the PoC will be explained and analysed in this section. This algorithm has been built considering the paper of Cordeau et al. (2001). The problem description of this specific case of a VRP can be found in Section 3.3. First of all, the creation of the initial solution will be explained in Section 4.2.1. Afterwards, in Section 4.2.2, the TS algorithm will be explained. Finally, in Section 4.2.3, the convergence of the algorithm is discussed.

4.2.1 Creation of the initial solution

In this section, the algorithm that constructs the initial solution of the current model of Districon will be analysed. In order to do this, the flowchart in Figure 7 has been used.

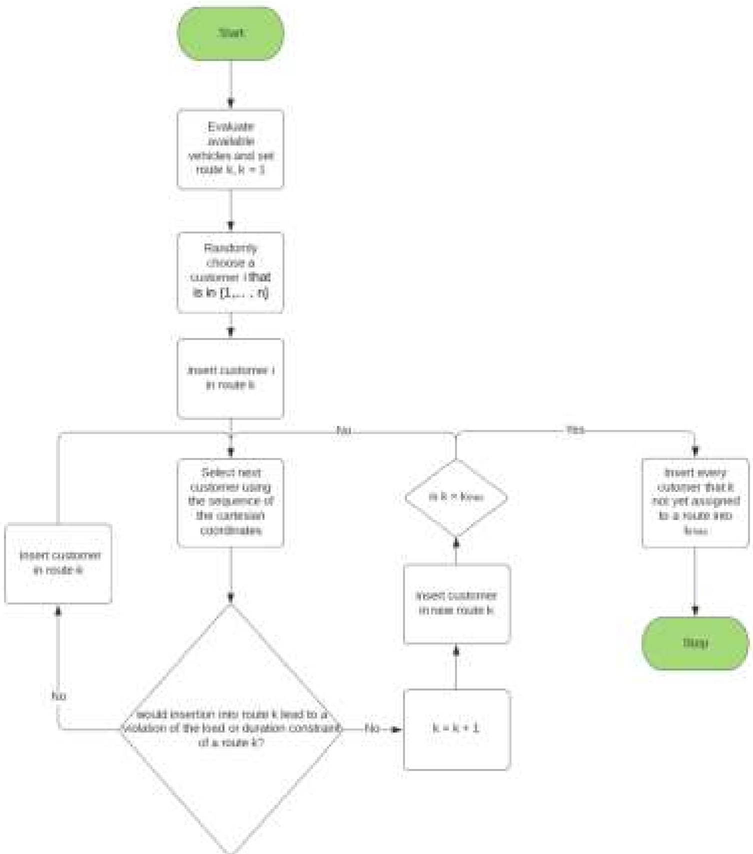


Figure 7: Flowchart of the creation of the initial solution of the current model

The cartesian coordinates of the customers are calculated with the depot as reference point. Subsequently, the customers are sorted in increasing order of the angle they make with the depot. At most k_{max} routes will be constructed, k_{max} is equal to m (see Section 3.4.1). The following procedure will be used in order to divide the customer over the routes:

1. Randomly select a customer $i \in \{1, \dots, n\}$
2. Set k = 1
3. Insert customer i into route k while minimizing the increase in the total travel time of route k while using $i, i+1, \dots, n, 1, \dots, j-1$ as a sequence of customers.

In case, the insertion of customer i into route k results in the violation of the load or duration constraints or maximum number of stops, customer i is inserted into new route (set k= min{k+1, m}). In case k = m and there are still customers left to assign to a route, they will all be inserted into route m. In step 3, a customer i can only be inserted between two successive customers if the time windows of all 3 involved customers are not violated. Otherwise, the customer i will be inserted at the end of the route. By following this procedure, all routes apart from the last route m, satisfy the load and duration constraints but may violate some time windows while route m may violate all three types of constraints. In this way, the initial solution is constructed and in the next phase of the algorithm, the TS will iteratively improve the solution.

4.2.2 Tabu Search algorithm

In this section, the Tabu Search algorithm of the current model of Districon is explained. First the objective value function will be explained. Afterwards, the operator used in the TS to create new neighbor solution will be elaborated on. Finally, the overall working of the TS is presented in a flowchart. Additionally, the convergence of the algorithm is discussed.

4.2.2.1 objective value function

The TS that will be used to improve the initial solution uses anti-cycling rules and diversification mechanisms. Furthermore, the TS allows the possibility to explore infeasible solutions during the search. The total costs value, c(s) is defined as the sum of the waiting costs, driving costs and violation costs. The violation costs are costs that compensate for violating the constraints and are denoted as positive parameters that will be multiplied per unit of violation: α (load constraints), β (maximum duration constraint) and γ (time windows constraints). The values of α, β and γ are initiated at 1. Note that the set-up costs per vehicle and working cost per hour are not included in this cost function and model.

4.2.2.2 Creation of neighbor solution

The solutions of S are characterized by an attribute set. Each solution $s \in S$ is associated with the set B(s) = {(i, k): customer i visited by vehicle k}. The local search heuristic explores the solution space by moving from the current solution s' to the best solution in its neighborhood N(s) at each iteration. The N(s) is defined by a simple operator that removes an attribute (i, k) from B(s) and replaces it with an attribute (I, k'), where $k \neq k'$. The route where customer i is removed from, is reconnected by linking the predecessor and successor vertices. Customer i is placed in between two vertices in route k' that minimizes the objective value. The visualization of this operator is represented in Figure 8 where the circles with numbers represent a customer and the depot is represented by the rectangle with DC. The operator starts at step 1 with a complete solution that is denoted as the current solution s' in the algorithm. Subsequently, in step 2 a single store will be removed from a route. In step 3, the operator starts checking every single inserting option in every route except in the route where the store is removed from. Finally, in step 4, the best insertion option will be implemented into the route. This procedure will be repeated for every single store in the dataset. The insertion option out of all of these possibilities will be implemented in the current solution. In this way, a new solution s' is created at every single iteration.

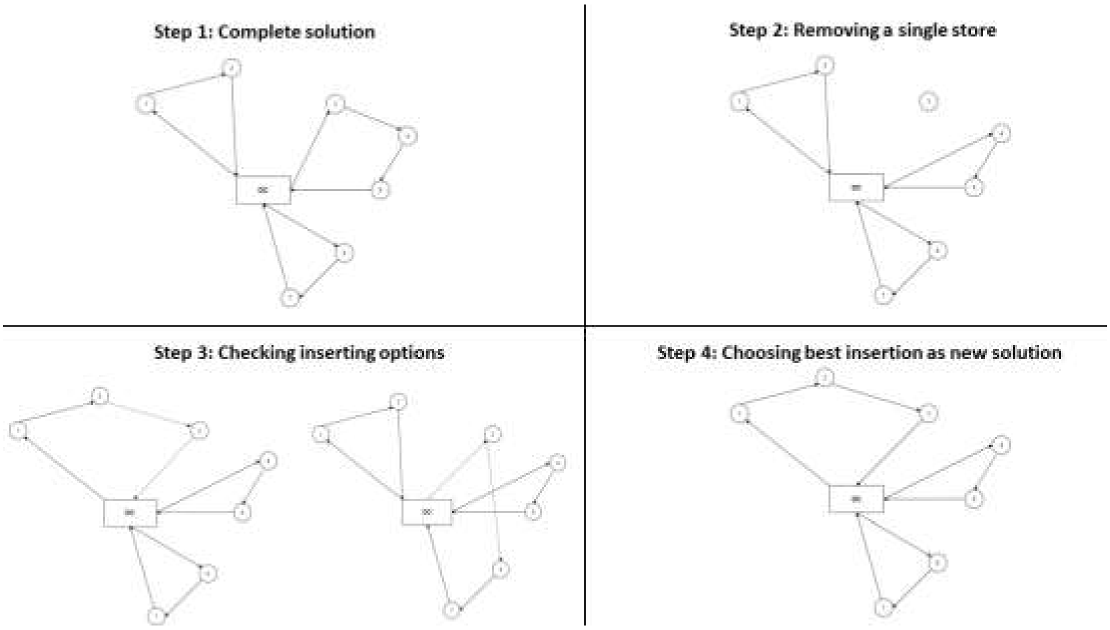


Figure 8: Visualization of operator in Districon's model. Source: Districon

4.2.2.3 Tabu Search

The overall working of the TS has been visualized in Figure 9.

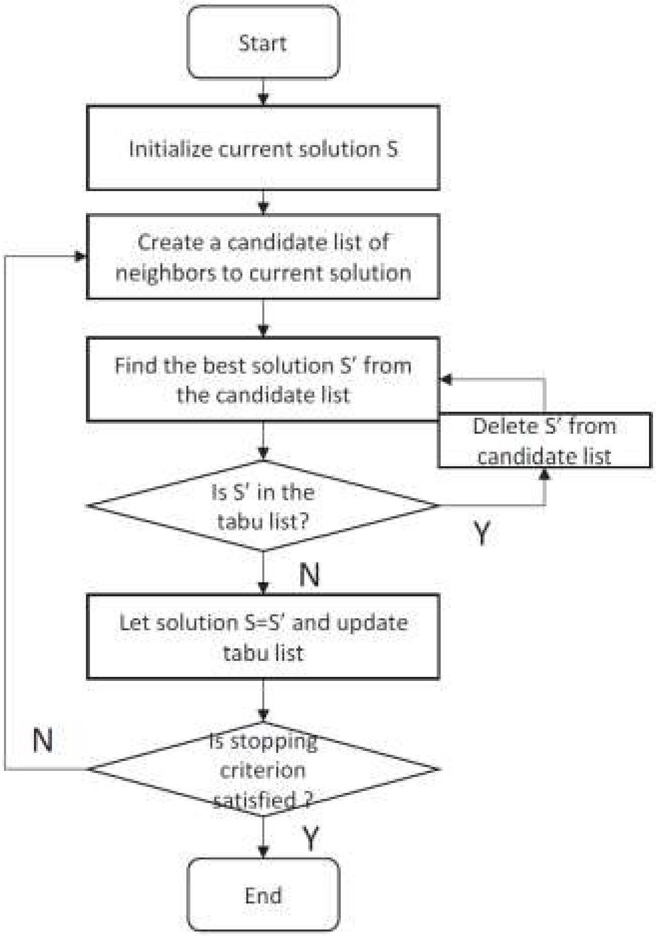


Figure 9: Flowchart Tabu Search current model (Hao et al., 2017)

The TS starts by importing the initial solution that has been constructed. Afterwards, the operator that is used to create neighbor solutions is used to create a candidate list of neighbor solutions. The best solution out of these candidates that are not included on the Tabu List (TL) is selected and implemented in the current solution. Furthermore, the attribute (i, k) is set on the TL that has a length of 10. In this model, no aspiration criterion has been formulated so the attribute will stay tabu for the next 10 iterations.
In order to diversify the search of the TS, any solution $s' \in N(s)$ that has a higher objective value than the current solution is penalized. The value of the penalty cost is based on the additional frequency of its attributes and a scaling factor. In more detail, ρ_{ik} denotes the number of times that attribute (i, k) has been implemented to the solution while executing the local search. The penalty cost is defined as $p(s') = \lambda * c(s') * \sqrt{nm} * \rho_{ik}$ and is added to the objective value. The objective value is calculated by the summation of the total costs and the penalty costs. The scaling factor $c(s') * \sqrt{nm}$ corrects the penalty costs with respect to the total solution costs and the size of the problem. The parameter lambda controls the intensity of the diversification and is initiated at 0,001. In case, the solution s' has a lower objective value than s, the penalty costs are equal to 0.
To explore the search space, the values of α, β and γ will be dynamically adjusted. After each iteration, these parameters will be updated. If the solution is feasible with respect to the constraint of the corresponding parameter, the value of the parameter will be divided by 1 + delta, otherwise, it is multiplied by 1 + delta. For example, if a solution violates the load constraints, α will be updated by multiplying it with 1 + delta. The parameters α, β and γ are bounded between -10 and 10. The complete algorithm works as follows:

1. Set α, β and γ = 1. Set delta = 0,25. If s (initial solution) is feasible, set s* = s and c(s*) = c(s), otherwise c(s*) = ∞.
2. While max iterations or max running time conditions are not exceeded:

- Choose a solution $s' \in N(s)$ that minimizes c(s') + p(s') and is not tabu
- If solution s' is feasible and c(s') < c(s*), set s* = s' and c(s*) = c(s')
- Update α, β and γ according to the above-mentioned procedure
- Set s = s'

Finally, after one of the termination criteria has been met, s* is the outcome of the model. This whole algorithm is presented in Figure 9 where N stands for 'no' and Y for 'yes'.

Note that the programmed model of Districon allows a truck to depart after the end of the time window of a customer when the truck arrives within the opening and end of the time window. In this, the constraint of hard time windows could be violated.

4.2.3 Convergence of the algorithm

The convergence of the algorithm can be found in Appendix B-4 for the first 1000 iterations. Note that the set-up costs per truck and working costs per hour are not included in the objective value. The objective value consists of the waiting costs, driving costs and costs per working hour. From the figure could be obtained that the first iterations are highly effective and improve the current solution often. However, after about 200 iterations, the algorithm barely improves the current best solution.

4.3 Conclusion

In this chapter, the current model of Districon, that solves a single depot homogeneous vehicle routing problem with time windows, has been scrutinized. In Section 4.1, the reason why the model initially has been built is discussed. This model has been constructed to test multiple scenarios in a former project. Furthermore, the model was programmed in Python since the algorithm in that language was able to solve the problem quicker than the solvers in AIMMS according to the employees of Districon. Moreover, the traveling time from vertice A to B has been calculated via OpenStreetMap with truck profiles including many customization options. The input data are imported via csv files and inherently the output of the model is presented in a csv file as well. Finally, it could be obtained that the set-up costs per truck and working costs per hour are excluded from the model.

In Section 4.2, the algorithms used to solve this specific from of a VRP are explained. The initial solution has been created by sorting the customers by their cartesian coordinates and insert them into routes until the limit of the capacity of a truck has been reached. This initial solution is improved by a TS that selects at every iteration the move of a customer that has the best effect on the total costs of the current solution. This move will then be set on the TL to prevent the algorithm to be stuck in a local optimum. Furthermore, to diversify the searching procedure, penalty costs are assigned to moves that already has been made in the past. The algorithm allows the search of infeasible solutions and the violations are penalized by dynamic parameters.

5. Solution design

In this chapter, the new model that is programmed in Python and the new proposed algorithm will be introduced and explained. This model consists of 3 phases: creation of the initial solution, improvement of the initial solution and a post-optimization method. In Section 5.1, the production of the initial solution and the design of the post-optimization method will be explained. Furthermore, additional changes to the model in order to fulfil all the requirements of Company A will be discussed. In Section 5.2, the new proposed algorithm that will be applied in the improvement phase will be discussed. Note that the improvement phase of the model of Districon is already elaborated on in Section 4.2. Finally, in Section 5.3, the chapter is summarized with the most important characteristics of this new model and the differences with the old model.

5.1 Adaptation of current model from homogeneous to heterogeneous

In order to perform valid and fair experiments between the new proposed algorithm and the current algorithm, both algorithms should be able to handle heterogeneous vehicle routing problems. Consequently, the current algorithm that is explained in Chapter 4, should be adapted such that it is able to solve VRP's that include heterogeneous vehicles. Firstly, the difference between the current problem description and the new one will be explained. Afterwards, the newly proposed algorithms for the initial solution and for the post-optimization method will be presented.

5.1.1 Current vs new problem description

The current problem description is defined in Section 3.3.1. This description concerns a homogeneous vehicle routing problem with hard time windows. The only difference between the new and the current problem description is the fact that the new model uses a heterogeneous vehicle fleet to solve the routing problem. This heterogeneous fleet is defined as K, where a vehicle type $k \in K$. The number of vehicles of type k is denoted as z_k. For every vehicle type, set-up costs w_k are defined. Furthermore, for each arc (v_i, v_j), nonnegative costs c_{ij}^k and traveling time t_{ij}^k are associated. These are the only differences with respect to the current and new problem description. The constraints defined in Section 3.3.2 are also similar.

5.1.2 Construction of the initial solution

First of all, the execution of the initial solution needed to be modified since the current algorithm for the initial solution divides the stores, that are sorted on cartesian coordinates, over the vehicles that are available. The constraint of vehicle capacity, maximum route duration and maximum stops per route are taken into account while inserting the stores over the routes, except for the last trip in case the maximum number of routes are used.

In the case of homogeneous vehicles, the order of the trucks does not matter since all characteristics of the vehicles are equivalent. However, in the case of heterogeneous vehicles, this order of trucks is important since characteristics as set-up cost, variable cost, speed limits (affects the time matrix of travelling from customer A to B) and capacity could be non-identical. For this contrast, the current algorithm that produces the initial solution has been extended and modified. This new algorithm for the initial solution consists out of 2 steps, one for dividing the customer over the vehicles and one for optimizing the available truck fleet utilization. The goal of this initial solution is to find the smallest fleet of vehicles with the highest truck capacity utilization that is still capable of serving every customer without violating any constraint. The reasoning behind this purpose of the initial solution is that it is in most cases better to use 1 large vehicle than 2 small vehicles since two vehicle denotes extra salary costs and potentially extra set-up costs. However, in some instances this is not the case. Therefore, a post-optimization method has been constructed to compensate for these situations. The goal of the post-optimization method is to split large route into 2 smaller routes. This procedure is explained in more detail in Section 5.1.3.

The customers are still sorted based on their cartesian coordinates, however, the trucks are sorted conditioning to their capacity from high to low. By sorting the trucks conditioning on their capacity from high to low, the smallest feasible fleet size will be found. The procedure of inserting the customers into the routes is presented in Figure 7 in Section 4.2.1. In the second step of the algorithm, the capacity utilization of the fleet will be optimized and works as follows as visualized in the following flowchart in Figure 10:

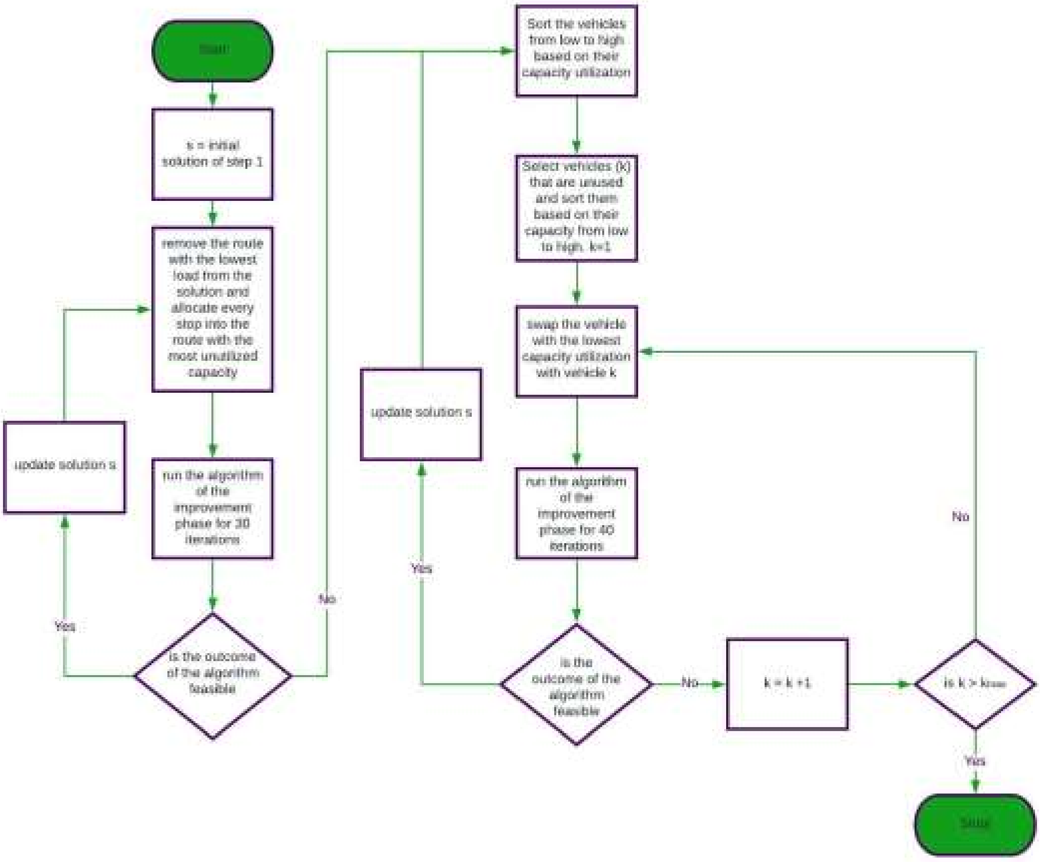


Figure 10: Flow chart of step 2 of the construction of the initial solution

First, the truck with the lowest load will be removed from the solution and every store is placed in the truck with the most unused capacity. The algorithm of the improvement phase will be applied for 30 iterations on this new solution. This number of iterations has been obtained from the paper of Molina, Salmeron and Eguia (2020) where the algorithm is applied for 30 iterations on every single initial solution. The algorithm is applied such that the model will not miss out on feasible solutions after a removal of a truck that, for example, results in just one-time window being slightly violated. In case, a feasible solution is obtained, this procedure will be repeated. Otherwise, the truck with the lowest capacity utilization will be swapped for the smallest truck available and the same operator will be applied. In case, an unfeasible outcome is obtained, the second smallest truck will be assigned to this trip and so on until no available vehicle types are left. In case, a feasible outcome is obtained, this same procedure will be applied on the truck with the second-lowest capacity utilization and so on. This procedure stops when a vehicle could not be feasibly swapped by a vehicle type that has a smaller capacity.

For both procedures, removing and swapping a truck applies the following rule: whenever the new full fleet capacity is lower than the total demand, the procedure will not be executed. This has been done to save time and iterations since the outcome of the procedure would always be infeasible because the total truck capacity is lower than the total demand which means that not every customer could be served.

The reason why this algorithm of the initial solution has been constructed is that the main goal of Company A is to optimize their truck capacity. In this way, the initial solution that will be used as starting solution for both the current algorithm as the new proposed algorithm is constructed.

5.1.3 Post-optimization method

After the algorithm of the improvement phase has finished operating, a post-optimization procedure is applied to attempt to improve the best-found solution of the algorithm. This post-optimization procedure aims to explore the possibility of two smaller vehicles being cheaper than one big truck. These smaller vehicles do not have to be identical. This post-optimization method is presented in the flow chart of Figure 11.

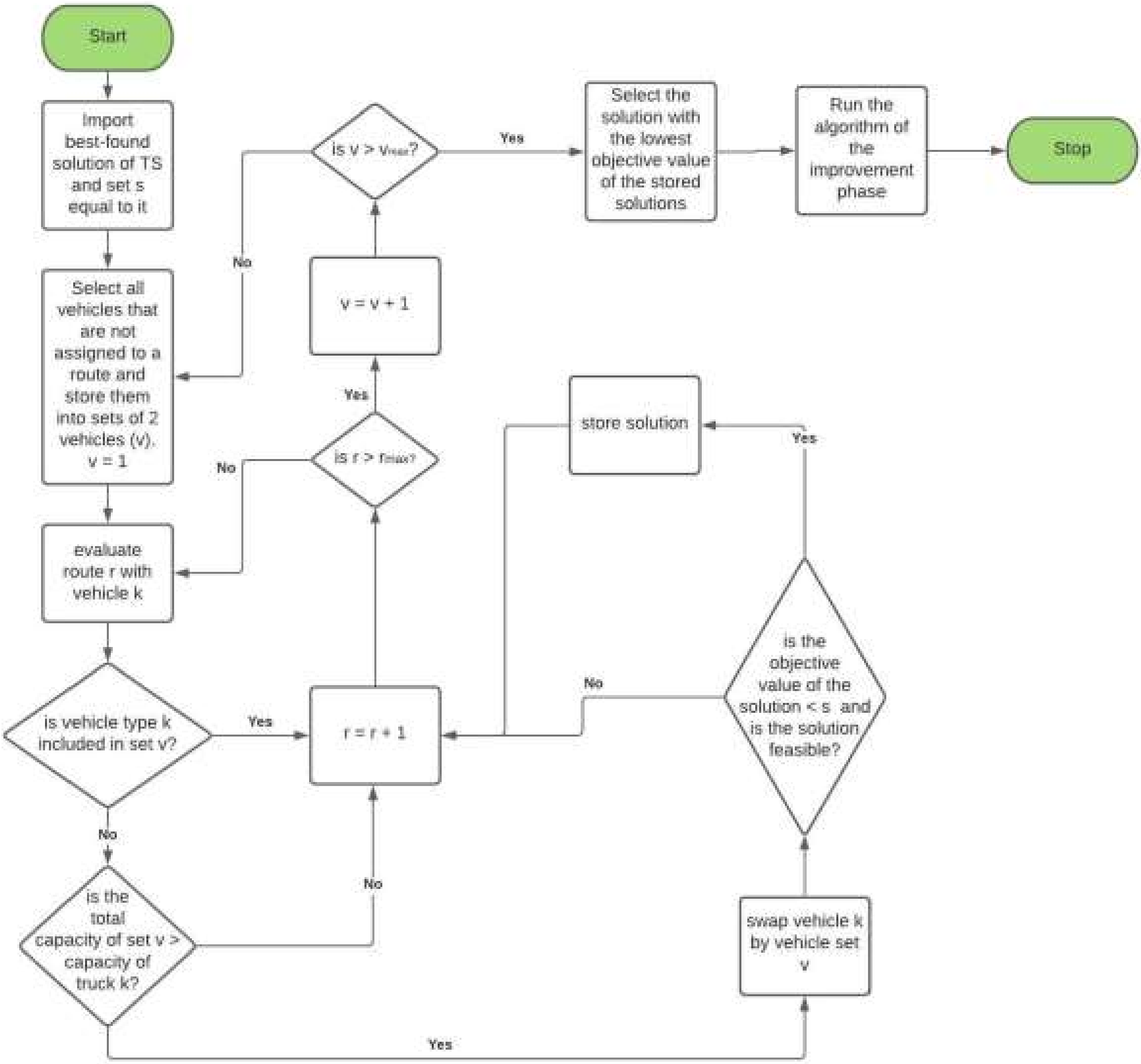


Figure 11: Flow chart of post-optimization method of the new model

The procedure starts by selecting all unutilized vehicles and store them all in unique sets with a length of 2. Every truck of a trip will be swapped by the 2 vehicles in the set under the condition that both vehicles in the set are not of the same type as the truck of the trip and the load of the trip should be smaller than the total capacity of the 2 vehicles in the set. Furthermore, the capacity of the truck of the current route should be higher than both capacities of the vehicles in the set. Otherwise, splitting the route into 2 pieces will in advance not improve the current solution.

The first truck in the set will be swapped with the current vehicle and every stop after the stop that exceeds the capacity of the first truck will be moved to the newly created route with the second truck of the set. In case the first truck has enough capacity to serve every customer of the route, no second route will be created and the second truck remains idle. In case, this new solution is feasible and improves the objective value, the solution is stored. After every swap of the truck sets into the routes has been evaluated, the stored solutions are ordered from low to high based on the objective value. The solution with the lowest objective value will be the new solution. This procedure is repeated until the length of the store solutions is equal to 0.

Finally, in case a new final solution has been created after the post-optimization procedure, the route will be split at the place where the truck capacity of the first truck has been exceeded and this may not be the best place to cut the route. Furthermore, a new route is created which implies a new search area for the algorithm used in the improvement phase that was not available before. Therefore, the solution found after the post-optimization procedure will function once again as the starting solution for the algorithm of the improvement phase in order to improve the newly constructed routes. The solution after this run of the algorithm will be the final solution of the whole model.

5.1.4 Additional adjustments to the model

In the previous sections, newly implemented algorithms have been discussed. However, in order to perfectly fit with the requirements of Company A, some adjustments had to be made to the model in general.

First of all, the set-up costs per truck were not included in the model and thus not in the objective value. Especially for heterogeneous vehicle fleets, it is important to include these costs since this could make the difference in assigning a specific truck to a route or not. Consequently, for every truck type, unique set-up costs could be assigned in the adapted model. These set-up costs were not implemented in the model since for the outcome of the model for the PoC the set-up costs were irrelevant. This was the case because the set-up costs were the same for every truck.

Furthermore, the hard time windows constraints of Company A could be violated in the outcome of the current model. In case that the vehicle arrives within the time window, it was possible for the truck after servicing the customer to depart while the end of the time window was expired. In the new model, this is impossible and the solution will be declared infeasible causing it will never be the outcome of the model.

Moreover, the working costs were not included in the objective value of the model. The model kept track of the total duration of the trips, however, no costs were assigned. In the new model, the working cost per hour, denoted in Section 3.4.1, are multiplied by the total duration of a trip such that the working costs are included in the objective value of the model. The working hour costs were not implemented in the old model since they were irrelevant for the previous PoC dataset since the traveling time matrix was the same for every truck.

5.2 New proposed Tabu Search Algorithm

In this section, the new proposed algorithm that is created to solve a heterogeneous vehicle routing problem including hard time windows will be explained. The problem definition of this specific form of a VRP is formulated in Section 5.1.1. This algorithm is a combination of the current algorithm of Districon that was based on the paper of Cordeau et al. (2001) and the work of Molina, Salmeron and Eguia (2020) where a Variable Neighborhood Tabu Search (VNTS) is introduced and where the VNTS is defined as "a hybrid approach that introduces the use of TS in the local search procedure of a Variable Neighborhood Search (VNS) scheme in order to explore the solution space in a more effective manner". In Section 5.2.1, the functioning of a Variable Neighborhood Tabu Search will be explained. Consequently, in Section 5.2.2, the multiple neighborhood structures used in the VNTS will be explained.

5.2.1 Functioning of the Variable Neighborhood Tabu Search

The functioning of the VNTS is shown in the flowchart in Figure 12, where s denotes the best solution, s' stands for the current solution and s'' for the neighbor solution:

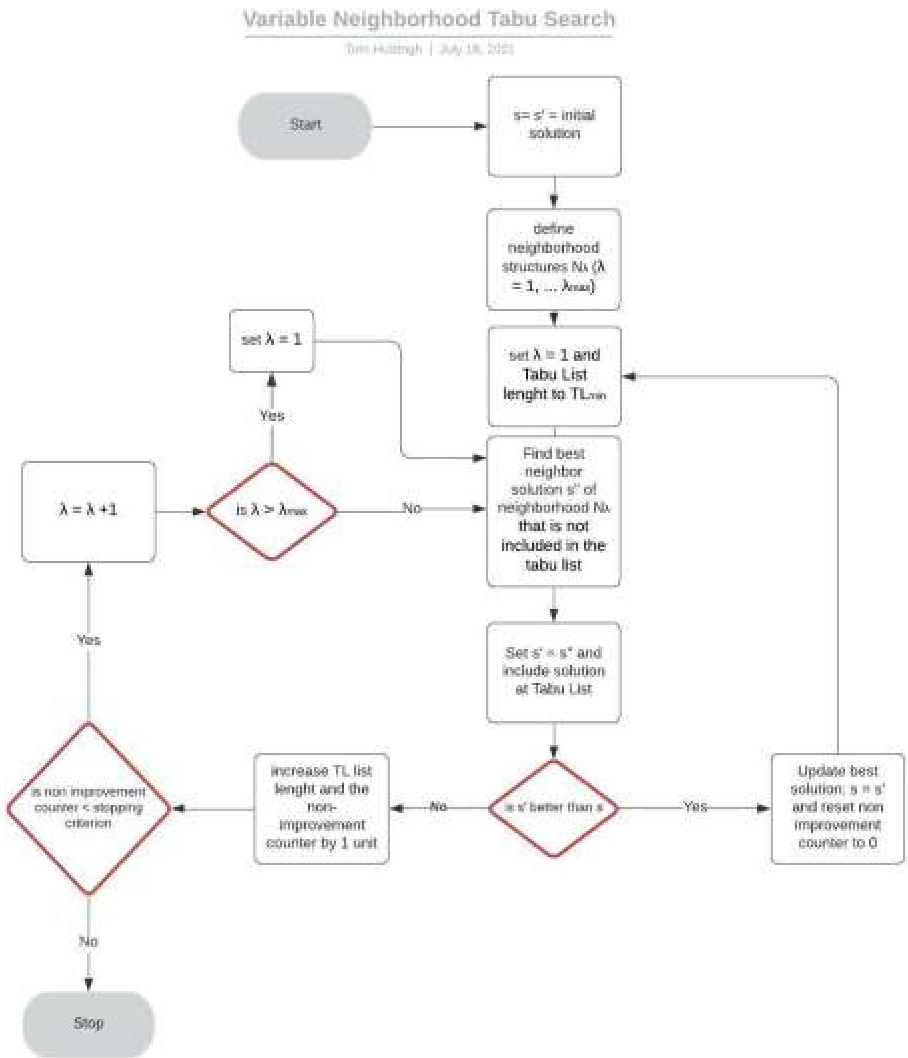


Figure 12: Flowchart of Variable Neighborhood Tabu Search

Firstly, a set of neighborhood structures N_λ ($\lambda = 1, \dots \lambda_{max}$) is defined, where N_λ is the λth neighborhood. The VNTS starts with the initial solution s produced by the algorithm in the previous section and the TS is used to find a new solution s' in N_λ , starting from λ = 1. "The VNS scheme avoids getting stuck in poor quality solutions since a local optimum for a given neighborhood structure is not necessarily so for another." (Molina, Salmeron and Eguia, 2020). In case, solution s' has a lower objective value than s and is feasible, s is replaced by s' and the TS resets to N_1, otherwise the TS will explore the next neighborhood $N_{\lambda+1}$. In the paper of Molina, Salmeron and Eguia (2020), this is done until a maximum of 30 iterations is reached or all N_λ have been explored. However, in the paper multiple initial solutions have been constructed thus to limit the computational time of the whole model in total, these constraints have been set. In this case, the model only has one initial solution, therefore in case $N_{\lambda\max}$ has been reached, it resets to N_1. The stopping criterion will be an x amount of iteration without improvement. This number will be experimented on in the next chapter.

After each iteration, the recently moved customer in combination with its old route is placed on the Tabu List (TL). Furthermore, the length of the TL is variable which allows the local search to diversify and intensify. A small TL will result in a more intensified search because it allows cycling of small periods while a large TL can help the TS escape from a local optimum by preventing the search to return to a recently visited solution which allows a more diversified search on more distant neighbors. The TS starts the search with the minimum length of the TL (TL_{min}) and moves a unit up after each iteration without an improvement up to a maximum of TL_{max} to diversify the search. In case, a new best solution has been found the TL reinitialises to TL_{min} , to intensify the search in this search area, where the oldest solutions will be removed from the TL. Moreover, the TS allows non-improving moves and even non-feasible moves to explore an even bigger area. For the non-feasible moves, violation costs are accounted in the exact same manner as explained in Section 4.2 by use of parameters α (load violation), β (maximum duration violation) and γ (time window violations). In case, the objective value of a solution that includes violation costs is lower than the current solution, the infeasible solution could be selected. However, this infeasible solution could never be selected as the best-found solution so far.

5.2.2 Neighborhood structures

In this section, the neigborhood structures used in the new proposed algorithm will be elucidated. In total 6 different neighborhood structures will be used to escape from cycling in a local optimum. The used neighborhood structures are Relocate (inter- and intra-route), Exchange (inter- and intra-route), Cross-Exchange and the GENI-insertion. Inter-route means that the operator is applied on pair of routes and intra-route infers to the application of the operator on a single route.

5.2.2.1 Relocate (inter- and intra-route)

The Relocate operator removes a customer from a route and inserts it at the best possible place of one of the other routes in case of inter-route relocation. In case of intra-route relocation, the customer will be inserted at the best possible spot in the current trip except for its current place. The Relocate procedure has been shown on the left-hand side in Figure 13, where customer i is removed between customers i-1 and i+1 and is inserted between j and j+1. This visualises the intra-route relocation operator. On the right-hand side of Figure 13, the inter-route relocation procedure is visualised, where the customers denoted by i form a route and the customers of j compose a route. The squares are used as starting and finishing points. Customer i is removed from the route with i-1 and i+1 and is added to the other route with j and j+1. This procedure is executed for every single customer at every iteration and consequently, the relocation that has the best effect on the costs of the current solution will be executed which will result in the new solutions s'. Note that the intra- and inter-route relocation operators denote 2 different neighborhoods.

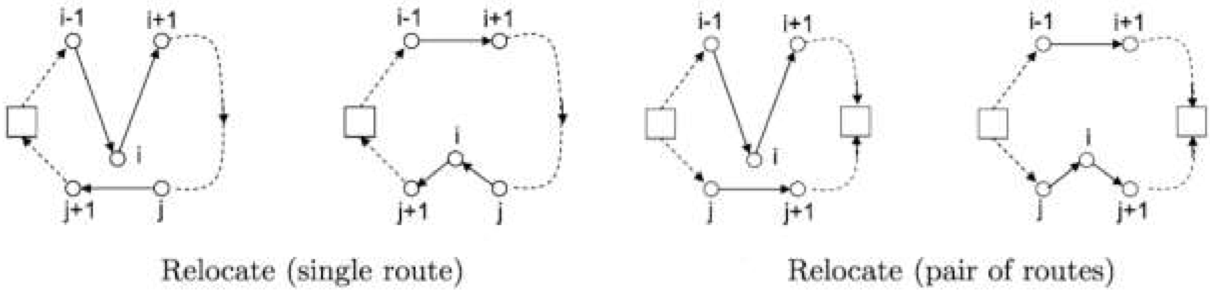


Figure 13: Relocate operator (Paraskevopoulos et al., 2007)

5.2.2.2 Exchange (inter- and intra-route)

The Exchange operator simultaneously swaps two customers. The intra-route Exchange has been visualised on the left side of Figure 14. It could be obtained that the positions of customer i and j within the same route has been exchanged. This procedure will be executed for every route and every set of 2 stores in a route. On the right side of Figure 14, the inter-route Exchange has been visualised where customer j from the upper route is exchanged with customer i from the lower route. An x-% of all possible exchanges is executed in order to decrease the computational time. The best swap in terms of the effect on the objective value will be executed for both neighborhoods which will result in solution s'.

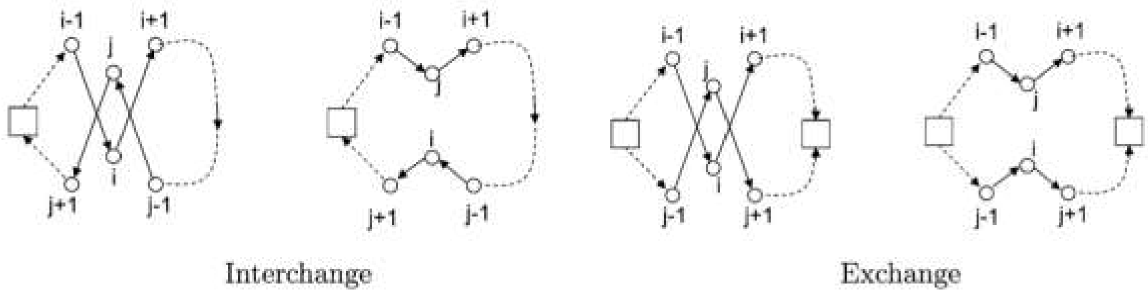


Figure 14: Exchange operator (Paraskevopoulos et al., 2007)

5.2.2.3 Cross-Exchange

The Cross-Exchange is an inter-route operator which means that the exchanges are only executed between two different routes. The Cross-Exchange swaps sets of consecutive customers between two routes. In Figure 15, the Cross-Exchange with two sets of two customers (2-2) is executed and displayed. On the left-hand side of the figure, the sets (j, g) and (i, e) are located in their current route and on the right-hand side of the figure, it could be obtained that these two sets have been swapped and two new routes have been constructed. Similarly, sets of 1-2, 1-3, 2-3 and 3-3 customers are swapped. The execution of this operator is in the same order as the list denoted above, where the set of 2-2 is swapped after the first set (1-2). For every set an x-% of all possible Cross-Exchanges is executed, otherwise, the computational time of the algorithm will be too long. The first time the algorithm visits this neighborhood the swap with sets of 1-2 customers will be executed, the second visit will execute sets of 1-3 customers and so on. The 6th time the algorithm visits this neighborhood structure, the set of 1-2 customers is again used for the Cross-Exchange operator. The swap out of the x-% possible Cross-Exchanges of the corresponding specific set that has the best effect on the objective value will be executed and thus implemented in solution s'.

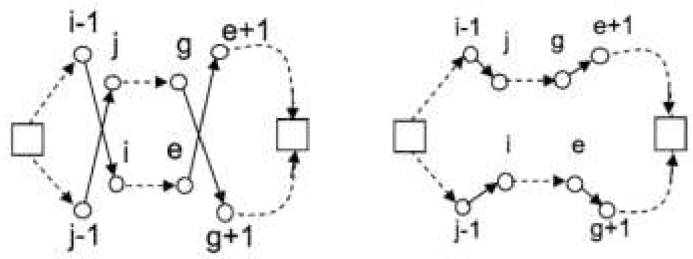


Figure 15: Cross-Exchange operator (Paraskevopoulos et al., 2007)

5.2.2.4 GENI-insertion

Finally, the GENI-insertion operator is used to find the best possible solution to the vehicle routing problem. Like the Cross-Exchange, the GENI operator is only applied on pairs of routes. The GENI operator starts by removing a customer from a route. Thereafter, this customer is placed between two customers, which do not have to be consecutive, in another route. An x-% of all possible customer sets where the removed customer could be inserted in is evaluated to save some computational time. In this new proposed algorithm, the GENI- insertion operator is not allowed to select consecutive customers in a route since then the exact same procedure will be executed as in a Relocate inter-route. Consequently, the iterations of the GENI-insertion operator will be used efficiently. This process is shown in Figure 16. On the left side of the picture customer, i is removed from the upper route and is inserted between customer j and k into the lower route on the right-hand side of the figure. It could be noticed that the order of the customers apart from customer k has remained the same.

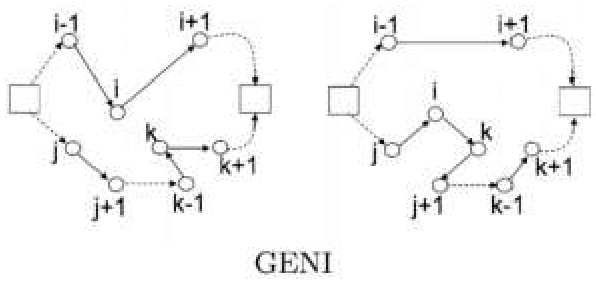


Figure 16: GENI-insertion operator (Paraskevopoulos et al., 2007)

5.3 Conclusion

In this chapter, the new model is discussed including the implementation of the new proposed algorithm to solve the heterogeneous vehicle routing problem with hard time windows. The most important difference between the current model of Districon and the new model is that the new model can handle a truck fleet with different truck types, implementing different set-up costs, variable costs and capacity. This has been done by introducing a new algorithm for the initial solution that aims to find the smallest vehicle fleet with the highest truck capacity utilization while feasibly serving all customers. Furthermore, the new proposed algorithm for the improvement phase of the model is discussed in Section 5.2. The framework of this algorithm is a variable neighborhood TS that uses a TL which length is varying. The new algorithm uses 6 different neighborhood structures: Relocate (inter- and intra-route), Exchange (inter- and intra-route), Cross-Exchange and the GENI-insertion. These neighborhood structures are all explained in Section 5.2.2. The different parameters of the new algorithm will be experimented within the next chapter to obtain a better result. Additionally, a post-optimization method has been included in the model that splits a route into two routes to evaluate if it is more cost-efficient to use two small vehicles instead of 1 big one. The structure of the model in general is visualised in Figure 17, where every step of the model in broad terms is presented.

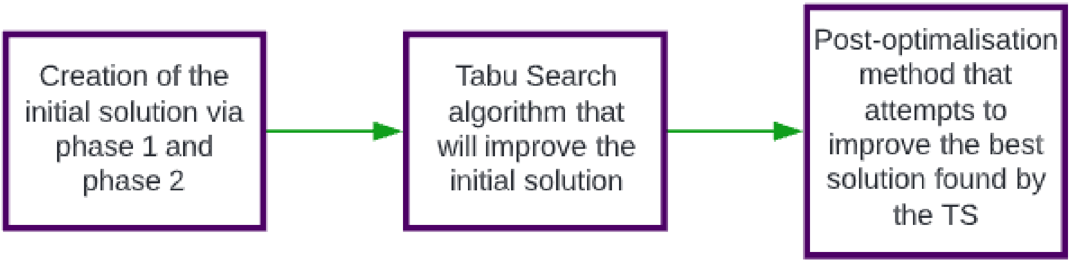


Figure 17: Flowchart framework of new algorithmic framework

Finally, the new model does not accept that a vehicle arrives within the time window and leave after the expiration of the time window. Moreover, the set-up cost per vehicle and the working cost per trip is included in the objective value.

6. Experiments

In this chapter, several experiments will be run in order to adjust the parameters of the model such that the algorithm fits the dataset of Company A well. This will improve the outcomes of the algorithm. In Section 6.1, the experimental framework including the input and output data will be presented. In Section 6.2, the various scenarios that will be experimented with are explained. Consequently, in Section 6.3, the results per experiment are presented and discussed. Afterwards, the best parameters of the model will be selected and implemented into the algorithm. Subsequently, this new algorithm will be tested against the old algorithm over various scenarios to compare both algorithms. Finally, in Section 6.4, a summary of this chapter will be presented.

The experiments are run on an HP Zbook studio G5 laptop with an 8^{th} generation Intel Core i7 processor and as mentioned before, the model is implemented in Python. Consequently, the experiments will be run in Python on the previous mentioned laptop where the executions are limited to one CPU.

6.1 Experimental design

In this section, the framework of the experiments will be presented and discussed. The input data of the various experiments will be discussed. First, the customer's dataset will be discussed. Afterwards, the datasets of the vehicle fleets will be introduced and the KPIs of the experiments will be discussed. Finally, the validation of the experiments will be discussed.

6.1.1 Customers data

Unfortunately, Company A provided only one dataset of customers including their locations, demand and time windows. Consequently, additional datasets have been generated in order to test the new algorithm on multiple instances. However, these new datasets are only used when comparing the newly proposed TS algorithm and the current TS algorithm. For the experiments that are related to finding the best settings of the newly proposed TS algorithm, an adjusted from of the dataset of the PoC has been used in combination with 6 different vehicle fleets that can be found in Table 5. This dataset has been analysed in Section 3.4. From this section, the cost parameters of the objective value can be obtained. This dataset includes one depot with n allocated customers. Furthermore, the constraints such as minimal departure time are defined in Section 3.3.2.

6.1.2 Vehicle fleets

The PoC dataset of Company A involves a homogeneous vehicle fleet, however, the new build model is constructed for a heterogeneous fleet. This means that new vehicle fleets need to be constructed to properly test the new model and algorithms. In total, 6 different vehicle fleets will be used while constructing these datasets, the data of the PoC vehicle fleet was considered and used as inspiration. The new truck fleets will all have about the same total capacity as the homogeneous fleet of the PoC. The characteristics of the heterogeneous vehicles will vary differently in the datasets in order to test the model in various scenarios. Firstly, the vehicle fleets will differ in the amount of different truck types that are available to serve the customers. Secondly, vehicle fleets will be distinguished by set up costs being linearly or non-linearly divided over the capacity of the vehicles. For example, if a truck with a capacity of 100 kg has a set up cost of 100, the set-up cost of a vehicle with a capacity of 50 is in the case of linear division of set-up costs over capacity 50. In the case of non-linearly division of set-up costs, the set-up costs of this vehicle will either be lower or higher. Furthermore, the variable costs can be the same for every truck type, however, they could also differ based on the capacity. Either a higher capacity leads to lower variable costs or the other way around. Finally, the driving time per vehicle could differ, some datasets of vehicle fleets include less driving time for smaller vehicles. The following vehicles are used in the vehicle fleets:

Truck type	Capacity (kg)	Set-up cost (1)	Set-up cost (2)	Set-up cost (3)
A	28.000	100	100	100
B	21.000	75	70	80
C	17.500	62,5	55	70
D	14.000	50	40	60
E	10.500	37,5	25	45
F	7.000	25	15	30
Average	16.333,33	58,33	50,83	64,17

Table 4: Vehicle types including their set-up costs

As mentioned before, the costs are unitless. In Table 4, the 6 different truck types including their capacity and the corresponding set-up costs for 3 scenarios. The header set-up cost (1) stands for the first scenario. Scenario 1 contains the linear division of the set-up costs over the capacity. Scenarios 2 and 3 contain the scenarios of non-linear division of the set-up costs. In scenario 2, the smaller vehicles do have lower set-up costs in proportion with the bigger vehicles. On the other hand, in scenario 3, the bigger vehicles will have lower set-up costs compared with the smaller vehicles. Given these different truck types, the following six vehicle fleets datasets have been constructed:

Dataset	Number of different vehicle types available	Vehicle Fleet	Set-up cost scenario (1,2 or 3)	Cost and time matrix
1	1	A8	1	1
2	4	A4, C3, E3, F4	1	1
3	6	A1, B2, C3, D3, E3, F4	1	1
4	6	A2, B2, C3, D3, E2, F3	2	1
5	4	A3, B3, D3, E4	2	2
6	6	A1, B2, C3, D3, E3, F4	3	3

Table 5: Overview of the 6 different datasets of the vehicle fleets

In Table 5, the vehicle fleets per dataset are been presented, where for example B2 stands for 2 trucks of type B (see Table 4). The set-up cost scenarios explained in the previous paragraph are denoted in the fourth column of Table 5. Finally, the cost and time matrix are denoted by either 1,2 or 3. In case, the cost and time matrix are denoted by 1 then the cost and time matrix made by Districon for the PoC will be used. In case, the number in this column is 2, the bigger trucks will have lower variable costs. Finally, if the number in the last column is 3, the smaller trucks will have lower variable costs and the travel time will be less. The new matrixes will be constructed by multiplying the current matrix by a factor. The factors used in dataset 5 from Table 5 will be 0.85, 0.9, 0.95, 1. Consequently, the cost matrix of vehicle A is multiplied by 0.85 and the cost matrix of vehicle D is multiplied by 0.95 and so on. The corresponding factors for the vehicle used in dataset 6 for the cost and time matrix are denoted as follows 1, 0.95, 0.9, 0.85, 0.8, 0.75.

6.1.3 Key Performance Indicators

In order to compare the change in the dependent variable in each experiment, some KPIs have been formulated. These KPIs both include data related as model related indicators.

Objective value
The objective value denotes the total cost value of the final solution that is the outcome of the model or algorithm. The total cost is the summation of the set-up costs of the vehicle used, the working costs, the total travelling costs and the waiting costs. In case a final solution of the model is not feasible, the violation costs of the solution will be added to the objective value.

Running time
The running time denotes the total time the algorithm or model needs to solve the vehicle routing problem. Since the requirement of Company A was to run the model in a maximum 15 minutes, it is important to keep track of the running time.

Overall truck capacity utilization
This KPI denotes the overall truck capacity utilization over all the trucks that have been utilized in the constructed routes. Since the objective of Company A was to improve the overall truck capacity utilization, it is necessary to keep track of the overall truck capacity utilization.

Utilized vehicles
This KPI denotes the vehicles that the final solution of the model used to solve the VRP. Company A asked Districon to come up with a solution to the PoC that contains as less as possible trucks. Therefore, it is interesting to compare the solutions of the models and algorithms by their truck fleets that are used for solving the VRP.

6.1.4 Validation of the experiments
In this section, the validation of the experiments will be discussed, both external as internal factors on the algorithm will be considered. The experimental design will be critically reflected.

Computational settings
First of all, every experiment has been executed on the same computer, implementing that the processor of all the experiments has the same quality. Consequently, the running times of the different algorithms are of the same proportion.

No invalid feasible outcomes
Before these experiments have been executed, the outcomes of the model have been extensively tested for violations for any constraint. After these experiments, all semi-errors in the model were removed and it is impossible to produce an infeasible solution as the final solution while the model denotes it is feasible.

Assumption
In Section 3.4.1, the assumption has been made to assign the unit 'kg' to the demand of the customers and the capacity of the trucks since no unit was assigned by Company A. This has had no effect on any outcome of the experiments because the unit for both demand and capacity is the same implementing that the proportion of demand to vehicle remained intact.

Experimental set-up
The initial solution algorithm and post-optimization method for every experiment conducted after the long experiment were identical. Furthermore, the dataset of customers is the same for every run of every experiment. Moreover, since the initial solution algorithm and the newly proposed algorithm depends on some randomness, the experiments including these algorithms have been run 5 times. Consequently, the randomness has been controlled.

6.2 Experimental scenarios

In this section, the various experiments that will be conducted will be explained. In total 8 different experiments will be conducted. First of all, the experiment regarding the different initial solution creation methods is elaborated on. Afterwards, the experiments regarding the parameters of the algorithm are explained in Section 6.2.2. Subsequently, the experiment regarding the post-optimization method is explained and finally, the experiment concerning the new proposed algorithm including the best-obtained parameters that will be tested against the old algorithm of Districon.

6.2.1 Initial solution

The newly designed algorithm that produces the initial solution is explained in detail in Section 5.1.1. This algorithm for producing the initial solution will be tested against other procedures. The newly proposed algorithm to produce the initial solution consist of 2 phases. In phase one the customers are sorted by cartesian coordinates and inserted in vehicles that are sorted based on their capacity from high to low. This procedure in phase 1 will be tested against the exact same algorithm apart from that in this case, the trucks are sorted based on the ratio capacity/set-up costs from high to low. Furthermore, an algorithm that allocates the customers random into vehicles will function as a benchmark solution. Since all these algorithms depend on randomness, every experiment will be executed 5 times.

Additionally, all 3 procedures for phase 1 will also be tested in combination with the algorithm of phase 2. So, in total 6 different experiments for the production of the initial solution will be conducted:

- Random division of customers overall available trucks
- Phase 1 of the algorithm of the initial solution
- Phase 1 of the algorithm of the initial solution with the trucks sorted on the ratio capacity/set-up costs
- Random division of customers overall available trucks + phase 2 of the new algorithm
- Phase 1 of the algorithm of the initial solution + phase 2 of the new algorithm
- Phase 1 of the algorithm of the initial solution with the trucks sorted on the ratio capacity/set-up costs + phase 2 of the new algorithm

6.2.2 Algorithm parameters

The parameters of the model where experiments are conducted for are the x-% swap of some neighborhood structures, minimum and maximum length of the TL, the delta and base value of the violation costs and the sequence of the neighborhood structures.

6.2.2.1 Long-run experiment

After the best initial solution has been found, a long run experiment will be executed to find out after how many iterations the algorithm gets stuck in a local optimum. The purpose of this experiment is to decide after how many iterations the algorithm will be terminated. This is a trade-off decision between less computing time and a (slightly) better solution. The graph with the objective value per iteration will be analysed and the termination criterion, number of iterations without an improvement, will be set. This termination criterion will then be used for all other experiments.

6.2.2.2 The x-% swap

The neighborhood structures Cross-Exchange, Exchange inter-route and the GENI-insertion make use of the x-% swap in order to save some computational time. However, the GENI-insertion has to evaluate many more possibilities than the Cross-Exchange and the Exchange inter-route. Firstly, the GENI-insertion operator evaluates for every customer an insertion spot in any other route. However, this spot is not between two consecutive customers but could be between all customers in the route contrary to the Cross-Exchange and Exchange inter-route operator where the swap is executed between two consecutive customers. For this reason, a GENI-insertion operator that considers every single possibility will take extensively more computational time than a Cross-Exchange and the inter-route Exchange operators. For this reason, the x value of the GENI-insertion operator is divided by 4.

6.2.2.3 The minimum and maximum length of the Tabu List

As explained in Section 5.2.1, the new proposed algorithm has a dynamic Tabu List (TL) length that resets to the minimum length after an iteration with an improvement and moves a unit up until the maximum length of the TL has been reached after an iteration without an improvement. This is done to allow the algorithm to intensify as well as to diversify the search in the search area. In the experiment, there will be experimented with different sizes of the TL.

6.2.2.4 Parameters for the violation costs

In Section 4.2, the parameters of the function for the violation costs, α (load constraints), β (maximum duration constraint) and γ (time windows constraints) are introduced. These parameters are initiated at 1 and after every iteration these values will be update by either multiplying it with 1 + delta or dividing it by 1 + delta. In case, an iteration produced a solution that is infeasible, the corresponding parameter(s) to the constraint(s) that has been violated is multiplied by 1 + delta. In case, a constraint is not violated, the corresponding parameter is divided by 1 + delta. These parameters do have a lower and upper bound that are denoted as -10 and 10. In the experiments, the base value of the parameters and the delta value are tested in combination with each other to find the best combination of the proposed parameters values in the experiment.

6.2.2.5 Sequence of the neighborhood structures

In Section 5.2, every single neighborhood structure that is used in the algorithm has been explained. However, they were written down in an arbitrarily order. Consequently, the optimal order of these 6 neighborhood structures needs to be obtained in order to let this algorithm operate well. In the experiments, 2 sequences have been obtained from a paper in the literature and sequence has been made by the researcher based on the two proposed sequences in the literature.

6.2.3 Post-optimization method

The post-optimization method is discussed in Section 5.1.2 and has the objective to split routes with big vehicles into 2 routes with 2 smaller vehicles. In this way, the post-optimization method strives to improve the current best-found solution. In this experiment, the functionality and the quality of the post-optimization method will be tested. This will be done by running the model with and without the post-optimization method. In this way, the effect that the post-optimization method has on the best-found solution after running the algorithm could be obtained.

6.2.4 New proposed algorithm vs current algorithm of Districon

The new proposed algorithm including the newly obtained parameters from the experiments will be tested against the current algorithm of Districon to see which algorithm perform the best at the different instances. For this experiment, the 3 new created databases will be used. This means that in total there are 4 customers datasets and 6 vehicles datasets. Consequently, 24 instances could be used to test the new algorithm against the current algorithm. The parameters of the new algorithm have been set in the previous experiments and the parameters of the current algorithm of Districon are given in Chapter 4.

6.3 Experiments and Results

In this section, the experiments will be explained in more detail and the results will be presented and analysed. In Section 6.3.1 the values of the parameters before their corresponding experiments are run, will be presented. In Section 6.3.2, the experiments on the initial solution are conducted. The long-run experiment of the new proposed algorithm will be shown in Section 6.3.3. Furthermore, the experiments regarding the parameters of the new proposed algorithm are reported in Section 6.3.4. Moreover, the experiments alongside the post-optimization algorithm are outlined in Section 6.3.5. Finally, the experiment regarding the comparison of the current algorithm of Districon and the newly proposed algorithm is presented in Section 6.3.6.

The experiments will all be discussed in the same manner. Firstly the experiment will be explained in more detail than has been done in the Section 6.2. Afterwards, the results of the experiment will be presented. Finally, the results will be discussed. This will be the outline for this whole section.

6.3.1 Input data

In order to run the experiments before the parameters have been set based on the experiments, the parameters need to be assigned for the experiments that are executed in advance. Consequently, the following parameters have been set and are given in Table 6:

Parameter	**Value**
Delta	0.25
Min Tabu List length	10
Max Tabu List length	30
x-% swap	20%
Base value of violation costs	1
Sequence neighborhood structures	Relocate (inter-route), Relocate (intra-route), Exchange (inter-route), Exchange (intra-route), GENI-insertion and Cross-Exchange

Table 6: Input data of the experiments

The values of the delta and base value of the violation costs are adapted from the parameter settings of Districon's current algorithm. The minimum and maximum length of the Tabu List are taken from the paper of Molina, Salmeron and Eguia (2020). The x-% swap value has been selected after some small experiments where the duration of the iteration was evaluated. Finally, the sequence of the neighborhood structures is a combination of the proposed optimal sequences provided in the paper of Molina, Salmeron and Eguia (2020) and Paraskevopoulos et al. (2007).

6.3.2 Initial solution

In this section, firstly the experiment will be explained. Afterwards, the interpretation in general of the figures that provide the results of the experiments in this chapter will be explained. Finally, the results of the experiments will be discussed.

6.3.2.1 Experimental set-up

As explained in Section 6.2.1, 6 methods that produce an initial solution will be tested on the 6 different vehicle fleet datasets. Before the results of the experiments will be run the random initial solution needs some more explanation in detail. The random initial solution will randomly allocate the customers to one of the available vehicles no matter how many customers are already inserted in this route. Furthermore, the customer will be inserted at a random place in the route of the vehicle that is selected. In this way, the random initial solution does not consider any constraint of the heterogeneous vehicle routing problem with hard time windows. As mentioned before in the following 6 ways, the initial solution will be produced:

- Phase 1 of the algorithm of the initial solution (Phase 1 Hcap)
- Phase 1 of the algorithm of the initial solution with the trucks sorted on the ratio capacity/set-up costs (Phase 1 cap/setup)
- Random division of customers overall available trucks (Random)
- Phase 1 of the algorithm of the initial solution + phase 2 of the new algorithm (Phase 1 + 2 Hcap)
- Phase 1 of the algorithm of the initial solution with the trucks sorted on the ratio capacity/set-up costs + phase 2 of the new algorithm (Phase 1 + 2 cap_setup)
- Random division of customers overall available trucks + phase 2 of the new algorithm (Random + phase 2)

6.3.2.2 Results initial solution experiments

The results of the experiments have been presented in Table 7 and 8 where the above-mentioned labels in the brackets denote the title of the corresponding columns. Furthermore, Obj denotes the objective value and the worst, best and average objective values have been presented. For every vehicle data set (instance in Table 7 and 8), 5 runs have been executed except for Phase 1 Hcap and Phase 1 cap/setup since these two initial solutions methods do not contain any form of randomness. The average computing time is given by Avg time in seconds. Finally, the column Gap represents the difference of the best solution with the benchmark solution in percentages. In this case, Random is used as benchmark solution. Note that every number is in the table has been rounded to one decimal. These remarks apply to every similar figure that is presented in this chapter.

		Random			Phase 1 Hcap			Random + phase 2	
Instance		**Obj**			**Obj**			**Obj**	
	Best	**Average**	**Gap (%)**	**Best**	**Average**	**Gap (%)**	**Best**	**Average**	**Gap (%)**
1	29395.7	239318.8	0.0	2598.6	2598.6	-91.2	2.16E+07	3.07E+08	73428.0
2	2421862.1	3252302.4	0.0	2250.1	2250.1	-99.9	703.0	9.06E+08	-100.0
3	1533055.6	2279542.8	0.0	943.0	943.0	-99.9	707.6	718.4	-100.0
4	1593493.6	1948233.7	0.0	1133.5	1133.5	-99.9	630.6	650.7	-100.0
5	1036825.6	1754325.5	0.0	1469.2	1469.2	-99.9	657.5	664.0	-99.9
6	3.68E+12	8.61E+12	0.0	992.3	992.3	-100.0	767.3	775.8	-100.0
Avg	3.64E+12	1.43E+12	0.0	1564.5	1564.5	-98.5	3.60E+06	2.02E+08	12154.7

Table 7: Results experiments initial solution

		Phase 1 + 2 Hcap			Phase 1 cap/setup			Phase 1 + 2 cap_setup	
Instance		**Obj**			**Obj**		**Obj**		
	Best	**Average**	**Gap (%)**	**Best**	**Average**	**Gap (%)**	**Best**	**Average**	**Gap (%)**
1	1.70E+06	1.70E+06	5668.6	2598.6	2598.6	-91.2	1.70E+06	1.70E+06	5668.6
2	1.43E+06	1.43E+06	-41.1	2250.1	2250.1	-99.9	1.43E+06	1.43E+06	-41.1
3	943.0	943.0	-99.9	943.0	943.0	-99.9	943.0	943.0	-99.9
4	665.2	689.2	-100.0	906.6	906.6	-99.9	614.6	621.0	-100.0
5	662.2	671.1	-99.9	1026.9	1026.9	-99.9	1026.9	1026.9	-99.9
6	992.3	992.3	-100.0	992.3	992.3	-100.0	992.3	992.3	-100.0
Avg	521047.3	521052.8	871.3	1452.9	1452.9	-98.5	521099.7	521100.8	871.3

Table 8: Results experiments initial solution

From Table 7 and 8 can be concluded that the random division of customers over all vehicles including the phase 2 method of the newly proposed algorithm for producing the initial solution, shows the best results overall. However, the danger of using a random division of customers over all vehicles has been shown in the outcomes of dataset 2. The average objective values after executing 5 runs are outstanding high, implementing that one of the 5 runs resulted in an infeasible initial solution with a very high objective value. This method produced in 4 out of the 6 datasets of vehicles the best solution. Furthermore, it stands out that the phase 2 of the algorithm that produces the initial solution, extremely deteriorates the initial solution for vehicle set 1. However, in general, the phase 2 method of the algorithm that creates the initial solution improves the solution of phase 1. Especially, in case of the phase 1 that includes the random division of customers over the vehicles. All in all, the Random + phase 2 algorithm for producing the initial solution will be used in the experiments starting from Section 6.3.4.

6.3.3 Algorithm parameters

In this section, the experiments concerning the algorithm parameters will be explained in further detail and reported. Firstly, the termination criteria will be set. Subsequently, the value of the x-% swap will be set variable. Secondly, the length of the TL will be used as dependent variable. Afterwards, the parameters for calculating the violation costs will be tested. Finally, the sequence of the implemented neighborhood structures in the TS will be experimented with.

6.3.3.1 Long run experiment

The long-run experiment will be executed in order to define the termination criterion of the algorithm. This will be done by using the initial solution that is in Section 6.3.2.1 described as Phase 1 + 2 cap_setup and the second dataset of vehicle fleets. This combination of dataset and production method of initial solution has been chosen because it could be obtained in Table 7 that the objective value of instance 2 could be 703.00 (Random + phase 2) while using 8 vehicles. Since the newly proposed algorithm is not capable of removing or adding a trip to the current solution, an initial solution method which output uses 8 vehicles is selected for this experiment. The initial solution of Phase 1 + 2 cap_setup is 1427299.91, consequently, the algorithm must execute many successful iterations to find the best possible solution. For this reason, the convergence of the algorithm can be analysed in more detail. As a result, this combination of dataset and initial solution has been chosen.

The full convergence of the algorithm can be found in Appendix C-1. In Figure 18, the objective values for the first 150 iterations are shown. The running time of this experiment is about 19 minutes. After iteration 150, the algorithm still manages to improve itself, however, these margins are very small but do take a lot of time to process. The total improvement from iteration 150 till iteration 2850 is only 0.81 %. For this reason, that part of the graph is not considered for the decision with respect to the termination criterion. From Figure 18, it can be obtained that the algorithm improves the current best solution in most of the first 105 iterations. To be sure, the algorithm does not stop to soon while iterating in case a local optimum has been reached in the first 150 iterations, the termination criterion of iterations without an improvement has been set at 30.

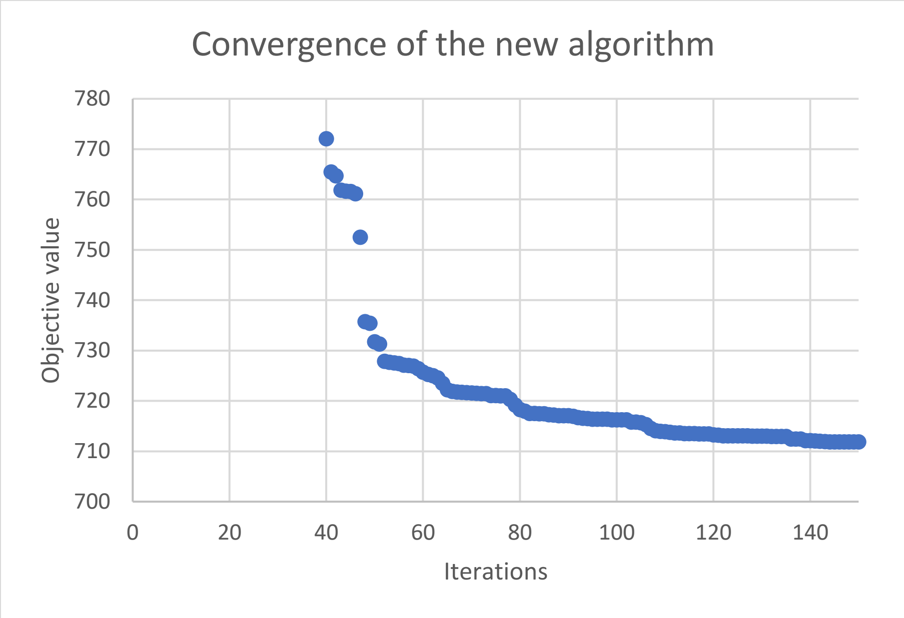


Figure 18: Convergence of the new algorithm including a zoom

6.3.3.2 The x-% swap

As mentioned in Section 6.2.3.1, the neighborhood structures Cross-Exchange, Exchange inter-route and the GENI-insertion make use of the x-% swap in order to save some computational time. Since the GENI-insertion considers more combinations of swaps, the x-% swap is divided by 4 for this operator. The values of x that will be experimented with are 5, 20 and 40. The results are presented in Table 9:

		5%-SWAP				20%-SWAP				40%-SWAP		
Instance		**Obj**				**Obj**				**Obj**		
	Best	**Average**	**Gap (%)**	**Avg time (s)**	**Best**	**Average**	**Gap (%)**	**Avg time (s)**	**Best**	**Average**	**Gap (%)**	**Avg time (s)**
1	2.07E+13	2.24E+14	0.0	21.2	2.30E+13	2.54E+14	11.2	40.7	2.40E+13	2.87E+13	15.8	66.9
2	704.6	713.3	0.0	53.3	706.6	711.9	0.3	96.7	700.5	705.5	-0.6	112.7
3	713.2	719.5	0.0	46.8	701.8	707.0	-1.6	92.3	705.8	709.8	-1.0	97.1
4	607.3	627.4	0.0	58.8	618.1	625.9	1.8	99.2	612.0	616.3	0.8	158.5
5	624.1	638.6	0.0	63.3	618.1	630.9	-1.0	106.1	621.4	634.4	-0.4	176.8
6	763.8	780.3	0.0	42.6	767.2	768.0	0.4	81.4	754.7	759.8	-1.2	113.9
Avg	682.6	695.8	0.0	53.0	682.4	688.8	0.0	95.1	678.9	685.2	-0.5	131.8

Table 9: Results experiments with the 5, 20 and 40% x-swap

The gap percentage has been calculated in relation to the results of the 5%-swap. The objective value of the outcomes of dataset 1 is so high and infeasible that they have not been considered in the graphs. The reason why these objective values are that high is that the initial solution is so bad that the algorithm cannot found a feasible solution. After each iteration, the violation cost parameters are updated. Since every iteration is infeasible the value of the parameters will go to their maximum and therefore assign extremely high costs to violations.

From Table 9, it can be concluded that evaluating more possible swaps per iterations does not necessarily leads to a better objective value since the 5%-swap outperforms the 20%- and 40%-swap on instance 4. However, in general the higher the x%-swap the better the results. This relationship has been graphed and can be found in Appendix C-2. Nevertheless, the increase of considering more swaps per iteration comes at a cost, namely the computational time. The higher the x-% swap, the higher the computational time as could be seen in Appendix C-1. Since the computational time of the 40%-swap does not exceed the maximum running time of 15 minutes that has been set by Company A, this value of x has been chosen as parameter of the algorithm. From now on the 40-% swap will be used in the next experiments.

6.3.3.3 Min and maximum length of the Tabu List

In this section, the experiment about the length of the Tabu List (TL) is conducted. The dynamic TL enables the algorithm to intensify the search when an improvement solution has been found and to diversify the search when a local optimum has been found. The experiments will include a narrow, a medium and a large TL size. In the paper of Molina, Salmeron and Eguia (2020), the minimum length of the TL is 10 and the maximum length is 30. In this experiment, the intervals 5-10 and 5-50 will operate as the narrow and large TL sizes. The results of this experiment are presented in Table 10.

	TL 5-10				TL 10-30				TL 5-50			
Instance		Obj				Obj				Obj		
	Best	Average	Gap (%)	Avg time (s)	Best	Average	Gap (%)	Avg time (s)	Best	Average	Gap (%)	Avg time (s)
1	2.33E+13	2.60E+13	0.0	67.3	2.40E+13	2.87E+13	2.7	66.9	2.73E+13	1.20E+14	17.2	69.4
2	696.3	704.3	0.0	131.8	700.5	705.5	0.6	112.7	709.4	711.3	1.9	169.7
3	697.9	701.2	0.0	125.5	705.8	709.8	1.1	97.1	701.8	704.4	0.6	138.4
4	610.5	626.6	0.0	97.7	612.0	616.3	0.2	158.5	613.9	624.2	0.6	172.7
5	635.1	638.2	0.0	171.6	621.4	634.4	-2.2	176.8	617.7	629.0	-2.7	189.8
6	758.0	771.5	0.0	127.6	754.7	759.8	-0.4	113.9	762.7	764.1	0.6	119.2
Avg	679.6	688.4	0.0	130.8	678.9	685.2	-0.1	131.8	681.1	686.6	0.2	158.0

Table 10: Results experiments with various Tabu List lengths

From Table 10, it can be concluded that the best size of the TL is defined as 10-30, which denotes a minimum length of 10 and a maximum length of 30. On average this TL length outperforms the other two lengths with respect to the objective value. Furthermore, the average computing time is only one second longer than the lowest computational time and the average capacity utilization is only 0.01 lower than the highest obtained. For these reasons, the TL 10-30 is selected and will be still be used in the following experiments.

6.3.3.4 Violation costs parameters

The violation costs are calculated by multiplying every single unit of violation to the corresponding violation cost parameter. However, these violation cost parameters, denoted as α (load constraints), β (maximum duration constraint) and γ (time windows constraints), are dynamic parameters. After every iteration, the violations per constraint will be evaluated. In case, a constraint is violated the parameter of the corresponding constraint will be multiplied by 1 + delta otherwise it is divided by 1 + delta. The costs parameters are initiated at 1. Since the parameters delta and the base value together define the value of the violation cost parameter, the experiments will be conducted by sets of delta and base values. The delta values that will be used in the experiments are 0.1, 0.25 and 0.4. The base values used in the experiments are 0.5, 1 and 1.5. The results of the experiments are presented in Tables 11, 12 and 13.

		Base value 0.5	Delta 0.1			Base value 1	Delt a 0.1			Base value 1,5	Delta 0.1	
Instance		**Obj**				**Obj**				**Obj**		
	Best	**Average**	**Gap (%)**	**Avg time (s)**	**Best**	**Average**	**Gap (%)**	**Avg time (s)**	**Best**	**Average**	**Gap (%)**	**Avg time (s)**
1	7.64E+07	9.28E+07	-50.6	67.2	1.55E+08	2.72E+09	0.0	66.7	1.93E+08	2.30E+08	24.6	64.7
2	700.0	706.6	1.0	119.1	693.3	704.5	0.0	142.9	698.9	709.9	0.8	176.6
3	703.0	704.5	1.1	157.2	695.4	700.6	0.0	135.6	705.5	710.1	1.4	119.6
4	606.4	618.9	-1.3	168.6	614.7	619.8	0.0	168.3	615.1	617.5	0.1	137.5
5	618.6	621.2	-2.4	181.7	633.6	635.0	0.0	177.5	621.6	631.2	-1.9	175.9
6	761.3	770.4	0.3	124.7	758.9	761.4	0.0	122.9	757.7	764.5	-0.2	128.0
Avg	677.9	684.3	-0.3	150.3	679.2	684.3	0.0	149.4	679.8	686.7	0.1	147.5

Table 11: Results experiments with various violation costs parameters

		Base value 0.5	Delta 0.25			Base value 1	Delta 0.25			Base value 1,5	Delta 0.25	
Instance		**Obj**				**Obj**				**Obj**		
	Best	**Average**	**Gap (%)**	**Avg time (s)**	**Best**	**Average**	**Gap (%)**	**Avg time (s)**	**Best**	**Average**	**Gap (%)**	**Avg time (s)**
1	1.4E+13	1.4E+14	8922239.7	67.3	2.4E+13	2.9E+13	1.6E+07	66.9	3.4E+13	3.7E+13	2.2E+07	66.0
2	697.4	705.3	0.6	103.6	700.5	705.5	1.0	112.7	701.8	709.8	1.2	157.6
3	701.2	704.0	0.8	115.1	705.8	709.8	1.5	97.1	699.1	703.1	0.5	99.0
4	616.0	620.7	0.2	145.3	612.0	616.3	-0.4	158.5	617.6	623.8	0.5	143.4
5	619.2	627.7	-2.3	174.8	621.4	634.4	-1.9	176.8	619.3	620.7	-2.3	157.5
6	763.9	764.8	0.7	122.9	754.7	759.8	-0.6	113.9	757.1	762.5	-0.2	115.5
Avg	679.5	684.5	0.0	132.3	678.9	685.2	-0.1	131.8	679.0	684.0	-0.1	134.6

Table 12: Results experiments with various violation costs parameters

		Base value 0.5	Delta 0.4			Base value 1	Delta 0.4			Base value 1,5	Delta 0.4	
Instance		Obj				Obj				Obj		
	Best	Average	Gap (%)	Avg time (s)	Best	Average	Gap (%)	Avg time (s)	Best	Average	Gap (%)	Avg time (s)
1	4.8E+17	5.6E+17	3.1E+11	86.9	7.8E+17	8.6E+17	5.0E+11	67.3	1.2E+18	1.3E+18	7.5E+11	64.6
2	700.5	708.1	1.0	144.0	696.1	701.7	0.4	110.6	699.0	703.6	0.8	123.7
3	700.2	703.4	0.7	109.2	704.8	707.2	1.3	104.4	702.0	709.9	1.0	97.5
4	607.6	611.8	-1.1	167.1	600.5	616.6	-2.3	171.0	609.4	620.0	-0.9	138.1
5	618.8	623.7	-2.3	140.8	618.4	630.9	-2.4	210.6	651.1	660.1	2.8	184.1
6	762.6	768.4	0.5	137.3	756.8	762.5	-0.3	131.2	758.5	774.0	-0.1	104.3
Avg	677.9	683.1	-0.3	139.7	675.3	683.8	-0.6	145.6	684.0	693.5	0.7	129.6

Table 13: Results experiments with various violation costs parameters

The objective values of base value 1 and delta 0.1 have been used as the benchmark solution thus the gap percentage has been calculated with respect to these solutions. It can be concluded from Tables 11, 12 and 13, that the best values according to this experiment for the base value and the delta value are 1 and 0.4. These parameters outperform every other combination of base value and delta with respect to the objective value. The computational time is relatively high compared to the other combinations of base value and delta value, however, this differs only 10 to 15 seconds. Consequently, the base value of 1 and the delta value of 0.4 will be used in the experiments that still needs to be conducted.

6.3.3.5 Sequence of neighbourhood structures

The last algorithm parameter that has to be set is the sequence of the neighborhood structures. The sequence of the neighborhood structures decides in which order the neighborhood structures will operate. The following sequences will be experimented with:

- Relocate (inter-route), Exchange (inter-route), GENI-insertion, Relocate (intra-route), Exchange (intra-route) and lastly the CROSS-Exchange
- GENI-insertion, CROSS-Exchange (only sets of 2-2), Relocate (intra-route), Relocate (inter-route), Exchange (intra-route) and finally Exchange (inter-route)
- Relocate (inter-route), Relocate (intra-route), Exchange (inter-route), Exchange (intra-route), GENI insertion and ultimately the CROSS-Exchange

The first sequence is originated from the paper of Molina, Salmeron and Eguia (2020) that used this order of neighborhood structures. On the other hand, the paper of Paraskevopoulos et al. (2007) has based their sequence on the cardinality of the neigborhood structures. This resulted in the second sequence. The third sequence has been constructed by combining parts of the sequences of both sequences found in the literature. The results of the experiments constructed can be found in Table 14.

		Sequence 1				Sequence 2			Sequence 3		
Instance		Obj				Obj			Obj		
	Best	Average	Gap (%)	Avg time (s)	Best	Average	Gap (%)	Avg time (s)	Average	Gap (%)	Avg time (s)
1	9.6E+17	9.7E+17	23.5	65.4	7.9E+17	1.1E+19	1.9	69.7	8.6E+17	0.0	67.3
2	704.8	708.8	1.2	229.9	697.8	807.0	0.2	321.3	701.7	0.0	110.6
3	703.8	708.0	-0.1	159.7	702.4	705.7	-0.3	307.4	707.2	0.0	104.4
4	628.0	638.6	4.6	216.0	623.1	629.5	3.8	458.0	616.6	0.0	171.0
5	624.6	641.1	1.0	203.0	620.3	626.9	0.3	438.8	630.9	0.0	210.6
6	764.5	770.7	1.0	161.5	763.9	770.2	0.9	274.3	762.5	0.0	131.2
Avg	685.2	693.4	1.5	194.0	681.5	707.9	1.0	360.0	683.8	0.0	145.6

Table 14: Results experiments with 3 different sequences of neighborhood structures

Sequence 3 has been used as benchmark solution thus the gap for sequence 1 and 2 have been calculated with respect to the best-found solution of sequence 3. From Table 14, it can be clearly concluded that sequence 3 outperforms sequence 1 and 2 in every single KPI apart from the number of vehicles used. For this reason, sequence 3 has been used in the following experiments.

6.3.4 Evaluation of the post-optimization method

In this section, the usefulness of the post-optimization method will be evaluated. This will simply be done by running the model, including the above set parameters, with and without the post-optimization method. The outcomes of this experiment are presented in Table 15:

		With post-optimization						Without post-optimization				
Instance		Obj						Obj				
	Best	Average	Gap (%)	Avg time (s)	Cap utlization	Vehicle used	Best	Average	Gap (%)	Avg time (s)	Cap utlization	Vehicle used
1	9.5E+17	1.1E+18	0.0	69.3	0.7	8.0	9.8E+17	5.6E+18	3.7	67.7	0.7	8.0
2	701.1	705.9	0.0	128.5	1.0	8.0	704.0	708.8	0.4	189.4	1.0	8.0
3	702.0	703.4	0.0	439.3	1.0	10.0	705.1	710.8	0.4	115.1	1.0	10.0
4	614.6	616.4	0.0	137.1	1.0	11.0	639.4	644.4	4.0	110.2	1.0	9.0
5	621.2	626.1	0.0	146.4	0.9	11.0	658.2	663.5	6.0	137.1	1.0	8.0
6	763.0	764.1	0.0	150.8	1.0	10.0	764.5	775.4	0.2	138.5	1.0	11.0
avg	680.4	683.2	0.0	200.4	1.0	10.0	694.2	700.6	2.2	138.1	1.0	9.2

Table 15: Results experiments while running the model with and without the post optimization method

In Table 15, the outcomes of the model including the post-optimization has been used as a benchmark. From Table 15, it can be obtained that none of the solution of the model without post-optimization outperform the solutions of the model including the post-optimization method. The use of the post-optimization method is especially visible for the solution outcomes of instance 4 and 5. The fleet of vehicles used in the final solution for both models have been presented in Table 16.

Instance	With post-optimization	Without post-optimization	Available fleet
4	A1, B1, C3, D3, E2, F1	A2, B2, C1, D3, E1, F0	A2, B2, C3, D3, E2, F3
5	A1, B3, D3, E4	A3, B3, D1, E1	A3, B3, D3, E4

Table 16: Description of used fleet sizes in experiment regarding the post-optimization method

From Table 16, it can be concluded that the post-optimization method perfectly does what it has to do, namely, evaluate whether it is more cost-efficient to use 2 smaller vehicles than 1 big vehicle. The vehicle fleet used by the model including the post-optimization method utilizes the smaller vehicles instead of the larger vehicles. In Table 15, it can then be noted that this difference in vehicle fleets, decreases the objective value by on average 5 %. Consequently, it can be concluded that the post-optimization is certainly useful and will be still be used in the next experiment.

6.3.5 Newly proposed TS algorithm vs current TS algorithm

In this final experiment, the TS algorithm that will be used in the new model will be selected. The newly proposed TS algorithm will be compared against the current TS algorithm of Districon. As mentioned before, the TS algorithm will not only be used in the improvement phase of the model. It will also be used in phase 2 of the creation of the initial solution and after the post-optimization method is finished. The initial solution of phase 1 will for both TS algorithms in the experiment be random apart from the initial solution of vehicle dataset 1. The algorithm parameters of the current TS algorithm of Districon can be found at Section 4.2 and the stopping criterion will be 10 minutes of iterating per run. The input parameters of the newly proposed algorithm are the parameters that performed the best in the experiments conducted in the previous sections. For this final experiment, new datasets are created in order to test both the algorithms in different situations. In total 3 new datasets will be used together with the adjusted PoC dataset in combination with the 6 different vehicle fleets, resulting in 24 different instances. The new datasets are similar to the PoC dataset, however some details are adapted. Firstly, the second dataset has 200 customers and smaller distances between the customers. Secondly, the third dataset has 250 customers and larger distances between the customers. Finally, the fourth and last dataset contains of 50 customers that are demanding larger demand. In order to compensate for the larger dataset, 2 extra trucks per truck type per vehicle fleet have been added. The results of the final experiment can be found in Tables 17 and 18. Instance 1 denotes the adjusted PoC dataset with the first vehicle fleet of Table 5 and e.g., instance 8 denotes the second dataset in combination with the second vehicle fleet of Table 5.

Current TS algorithm							
Instance			**Obj**	**Gap (%)**			
	Initial solution	**Best**	**Average**		**Avg time (s)**	**Truck utilization**	**Vehicles used**
1	5262.306	882.7021	884.0696	13.03672	745.4603	0.747504	8
2	1965448	703.4259	726.7771	1.899631	815.358	0.970044	11
3	1234644	703.146	739.6795	0.877689	807.7872	0.97876	9
4	1323401	590.4362	650.0069	-2.12246	810.6792	0.970795	13
5	437578.9	613.3752	634.3724	-2.84916	1099.185	0.927419	12
6	738599	818.6733	848.5348	7.586559	795.6481	0.930018	14
7	368137.7	1084.843	1085.755	-0.19402	714.678	0.990145	10
8	6800935	1111.039	1117.077	-100	780.7264	0.988615	17
9	5143852	1122.752	1128.485	-1.11422	709.7255	0.964247	17
10	4761124	975.1133	991.5301	-0.88278	734.6992	0.969866	19
11	4371197	1008.907	1018.508	1.001074	696.9985	0.958496	16
12	5776475	1224.577	1227.621	0.073667	716.5373	0.966476	17
13	38324.55	881.9742	1119.434	-100	703.7438	0.781644	8
14	697288.9	748.4997	771.5734	3.389769	818.7554	0.941557	15
15	1054944	754.5053	777.6574	4.041795	727.4231	0.90635	13
16	1170341	623.1362	630.5848	-0.28323	837.2788	0.952753	13
17	990627.6	683.331	736.3928	4.950022	716.2755	0.865274	13

18	818359.4	801.4442	833.5231	3.351852	820.0796	0.975261	11
19	1081253	1293.214	1337.094	76.83774	691.127	0.567101	12
20	4745891	1344.207	1358.423	84.70483	693.711	0.78575	20
21	2688974	1086.177	1152.25	48.14883	695.1633	0.707384	18
22	5200349	889.8235	963.7565	38.46964	694.6548	0.780758	17
23	5234058	923.6241	999.8735	40.71299	692.9708	0.932003	15
24	3598097	1183.055	1184.934	49.64174	693.7367	0.812504	18
Avg	2510215	918.8326	954.913	7.136612	758.8501	0.890447	14
Avg (ex 8 and 13)	2427541	911.7713	940.0637	16.8763	760.3605	0.89093	14.13636

Table 17: Results current TS algorithm applied to the datasets

			Newly proposed TS algorithm				
Instance			**Obj**				
	Initial solution	**Best**	**Average**	**Gap (%)**	**Avg time (s)**	**Truck utilization**	**Vehicles used**
1	9205.75	780.8985	782.1928	0	189.1604	0.85429	7
2	3026371	690.3125	699.37	0	216.7832	0.996351	7
3	945053.8	697.0282	699.507	0	142.5512	0.996717	9
4	221759.6	603.2397	612.9557	0	187.8946	0.974837	12
5	694465.5	631.3638	635.2611	0	215.8654	0.918921	11
6	1393228	760.9438	763.2878	0	166.7357	0.99686	10
7	1449120	1086.952	1087.838	0	704.5308	0.990145	10
8	5612637	2.18E+20	2.42E+20	0	87.69401	1.051315	22
9	5540976	1135.402	1.54E+20	0	209.5839	0.933572	18
10	4436870	983.7981	9.44E+19	0	488.5729	0.987372	17
11	3423125	998.907	8.1E+19	0	307.4573	0.969146	16
12	4792791	1223.676	6.26E+19	0	443.8768	0.976224	18
13	38379.31	1.43E+18	1.54E+18	0	264.2846	0.481011	13
14	1398056	723.9592	735.82	0	1259.517	0.977452	9
15	1823813	725.1945	733.6187	0	1004.675	0.981901	10
16	743318.8	624.9061	633.7352	0	1150.823	0.959794	13
17	341102.2	651.1013	1.08E+19	0	751.8256	0.977897	10
18	1077522	775.4522	779.7785	0	1157.1	0.985541	9
19	533591.2	731.3	733.0596	0	8.680444	0.972173	7
20	1797771	727.7596	732.0244	0	15.72037	0.989575	9
21	4077143	733.1658	733.6496	0	13.20516	0.98915	10
22	5828953	642.6127	4.82E+50	0	17.35742	0.936966	13
23	2279162	656.3887	5.71E+55	0	12.71341	0.968789	11
24	3246999	790.5919	809.3593	0	11.53728	0.989119	10
Average	2280476	9.15E+18	2.38E+54	0	376.1727	0.952297	11.70833
Avg (ex 8 and 13)	2230927	789.7706	2.6E+54	0	394.3712	0.969218	11.18182

Table 18: Results newly proposed TS algorithm applied to the datasets

In Table 17 and 18, the new proposed TS algorithm has been used as benchmark solution. Accordingly, it can be concluded that the current algorithm of Districon provides on average solutions that include 7.1% more costs. On 16 out of 24 instances, the new algorithm provides better results than the current algorithm. Furthermore, it stands out that the running time of the new algorithm is relatively low with respect to the 15 minutes and the running time of the current algorithm. The average running time is 376 seconds, while the average running time of the current algorithm is 758 seconds. However, for the instances of dataset 3 (12 to 16), the average running time is above the 15 minutes. Moreover, the average used vehicles of the new proposed algorithm is significantly lower than the average used vehicles of the current algorithm, namely 11.7 against 14. Consequently, the average truck utilization of the new proposed TS algorithm is also higher: 0.95 vs 0.89. One remarkable result is that the newly proposed TS algorithm was not able to handle the initial solution provided for instances 8 and 13. In case, the initial solution creation method of Districon was used, feasible results appeared. From Table 18, it can be noticed that such situations occur more often since the average objective value is for some instances extremely high. Fortunately, the running times for almost all of these instances are well within the 15 minutes, consequently, there is sufficient time left to run the model again and obtain a feasible outcome. In case, the instances 8 and 13 are excluded from the results, the newly proposed algorithm provides results that are on average 16.9% better than the current algorithm with respect to the objective value. All in all, it can be concluded that the new proposed TS algorithm outperforms the current algorithm on average considering the KPIs.

6.4 Conclusion

In this section, a summary will be provided over this chapter regarding the conducted experiments. Firstly, in Section 6.1, the experimental design including the datasets and KPIs has been explained. Afterwards, in Section 6.2, the 8 experiments that are conducted are explained. Section 6.3 provides the results of the experiments that have been executed.

From the experiments that are executed in this chapter, it can be concluded that the best method to produce an initial solution is the random division of customers over the vehicles including the second phase of the new proposed algorithm for the initial solution. The termination criterion has been set at 30 iterations without an improvement. Moreover, the minimum and maximum length of the TL has been set at 10-30. Furthermore, the base value regarding the violation costs has been set at 1 and the delta parameter has been set at 0.4. In addition, the sequence of the neighborhood structures is as follows:

- Relocate (inter-route), Relocate (intra-route), Exchange (inter-route), Exchange (intra-route), GENI insertion and ultimately the CROSS-Exchange.

The post-optimization method showed its usefulness in the experiments in Section 6.3.5 since none of the experimental results of running the model without the post-optimization method outperforms the results of running the model including the post-optimization method. On average, this difference was 2.2%, implying that the post-optimization method functions well. Above all, the new proposed algorithm that uses all the above-mentioned parameters outperforms on average over all the instances the current algorithm of Districon. In 16 out of 24 instances, the new algorithm proposed solution including a lower objective value. In general, the new algorithm outperforms the current algorithm with 7.1% with respect to the objective value. Furthermore, the KPIs running time and truck utilization indicates that the new algorithm outperforms the current model. The new algorithm needs on average 382 seconds less computing time and the average truck utilization is 6% larger. All in all, it could be concluded that the new proposed algorithm provides on average better results than the current algorithm used by Districon.

7. Conclusions, recommendations and future research

This final chapter will conclude the research that has been conducted and has been reported in the previous chapters. Section 7.1 will discuss the limitations of the new model and will critically reflect on the newly proposed algorithm. In Section 7.2, the main research question formulated in Section 1.3.1 will be answered and motivated. Afterwards, in Section 7.3 the recommendations will be presented. Finally, the last Section 7.4 discusses further research that can be conducted by Districon.

7.1 Discussion

In this section, the limitations of the new model will be discussed. Districon should take them into account when this model will be used in practice in future projects.

Firstly, in the experiments, only one dataset of customers was used to set the parameters of the model Consequently, every parameter that has been set based on an experiment, is related to this specific dataset. In future projects, the datasets will be different, either larger or smaller and the customers will have different characteristics. For example, the x-% swap parameter that has been set at 40 % after running some experiments, implements that 40% of every possible swap will be evaluated at every iteration. However, if the dataset, e.g., includes 250 customers instead of a bit more than 100, running the model will take way longer as shown in Table 18. For this reason, the optimal parameters of the model could be different for other instances with different characteristics.

Secondly, the best method to produce the initial solution according to the experiments is to firstly divide the customers random over the available vehicles. Afterwards, the second phase of the initial solution creation method will be applied and with help of the algorithm the vehicles used will be minimized with respect to the costs. However, the danger of this procedure is that sometimes the new proposed algorithm is struggling with the initial solution since the initial solution of phase 1 is completely random and could be too hard to handle for the new algorithm. This will result in an infeasible outcome of the model while it is possible to produce a feasible solution.

Finally, the algorithm is quite restricted in moves after the two-stage method for creating an initial solution has been executed because the algorithm has not the ability to remove or add a trip while iterating. Since the initial solution is also created with help of the algorithm (phase 2), the trips are already quite well optimized and the number of vehicles used is (almost) minimized. Consequently, the algorithm has limited space for exploring his search in the improvement phase. On the other hand, the algorithm has much space for exploring his search in the second phase of the creation of the initial solution since all customers are divided over all the possible vehicles. This results in a large amount of possible swaps and thus a relatively big exploring space.

7.2 Conclusions

In this section, the main research question, formulated in the first chapter, will be answered. The main research question is formulated as:

How can the current model of Districon be adapted in order to fulfil all the needs of Company A?

This answer to this question has been derived by answering the sub research questions, denoted in Section 1.3.1. First of all, a literature study was conducted in Chapter 2 in order to obtain more information about the vehicle routing problem and its solving methods. Furthermore, 4 papers that solves a Heterogeneous Vehicle Routing Problem with Hard Time Windows (HVRPHTW) with help of a Tabu Search (TS) algorithm have been reviewed and the algorithm from the paper of Molina, Salmeron and Eguia (2020) has been selected to implement into the new model. This algorithm has been chosen since it fits the case of Company A well in terms of maximizing the truck capacity utilization.

Furthermore, it outperforms the other algorithms on well-known instances and it is a recent paper that indicates that the researcher has the ability to use the newest findings in research.
In Chapter 3, the requirements of Company A were determined after some interviews with employees of Districon. Company A would like to have a tailor-made Routing Planning Software (RPS) that it able to solve heterogeneous vehicle routing problems including hard time windows within 15 minutes. Chapter 4 has been used to analyse the current model of Districon and the algorithms that are included. This was necessarily to implement the newly proposed algorithm correctly. The current model of Districon produces the initial solution by sorting the customers by their cartesian coordinates and insert them into routes until the limit of the capacity of a truck has been reached. This initial solution will be improved by a TS that diversifies the searching procedure by assigning penalty costs to moves that has been made in the past. Furthermore, the algorithm allows the search in the infeasible area, these violations are penalized by dynamic parameters.
The newly proposed algorithm that uses a Variable Neighborhood Tabu Search (VNTS) is explained together with the construction of the initial solution and the post-optimization method in Chapter 5. The new proposed algorithm consists of 3 phases: creation of the initial solution, improvement phase and the post optimization phase. The initial solution has been constructed by first dividing the customers random over the available vehicles and afterwards with help of the VNTS the vehicles used in the solution will be minimized. This completes the initial solution and this initial solution will be improved by the VNTS in the improvement phase. This VNTS uses 6 different neighborhood structures: Relocate (inter- and intra-route), Exchange (inter- and intra-route), Cross-Exchange and the GENI-insertion. These neighborhood structures are all explained in Section 5.2.2. The algorithm terminates its search when 30 consecutive iterations without an improvement have been made. Subsequently, the best-found solution will be used as input solution for the post-optimization method. The post-optimization method attempts to split a route into two routes to evaluate if it is more cost-efficient to use two small vehicles instead of 1 big one. Finally, the VNTS will be applied once again to the output of the post-optimization method since new routes could be constructed which implements a new search area for the VNTS. In this way, the algorithm is able to tackle heterogeneous vehicle routing problems and make use of various different vehicles within a vehicle fleet. This completes the newly proposed algorithm and in Chapter 6, experiments were conducted in order to improve and evaluate the performance of the new proposed algorithm.
In Chapter 6, the parameters are explained in detail as well as the corresponding experiments. From the experiments the parameters of the model have been set. The algorithm terminates after 30 iterations without an improvement, the minimum and maximum length of the Tabu List (TL) are 10 and 30, for the neighborhood structures GENI-insertion, Cross-Exchange and Exchange inter-route 40% of the swaps will be evaluated per iteration, the base value of the parameter that assigns the violation costs will be set to 1 and the delta parameter to 0.4 and lastly the following sequence of neighborhood structures will be used:

- Relocate (inter-route), Relocate (intra-route), Exchange (inter-route), Exchange (intra-route), GENI insertion and ultimately the CROSS-Exchange.

The answer to the main research question is that by using the new proposed algorithm the requirements of Company A are all fulfilled. For the experiments, 4 different datasets concerning the customer data and 6 different datasets concerning the vehicle fleets have been used. This has resulted in 24 instances where the current algorithm could be tested against the new proposed algorithm. This new algorithm provides on average within 15 minutes a solution to the HVRPHTW by using the above-mentioned parameters and VNTS. The new algorithm outperforms the current algorithm on 16 out of 24 instances. On average, this resulted in a 7.1% lower objective value while needing 382 seconds less computational time. Furthermore, the average truck capacity utilization is 6% larger. However, in some cases the new proposed algorithm is struggling with the provided initial solution and is not able to find a feasible solution. Since the computational time is, in most cases, well within the 15 minutes this causes not always trouble. All in all, this new proposed algorithm should be used in the adapted version of the model, which takes into account heterogeneous vehicles including different characteristics as set-up costs, travelling time and costs.

7.3 Recommendations

After conducting this research and running many experiments, some recommendations for Districon have been formulated.

Firstly, it is recommended to Districon that they should test this new model on even more various datasets in order to define the parameters values. Consequently, the algorithm parameters fit for datasets in general and are not just aligned for one dataset.

Furthermore, to enlarge the search area of the algorithm in the improvement phase, Districon should enable the algorithm to create and remove trips while iterating such that more potential moves are created. This could help the algorithm escaping from a local optimum.

Finally, further research should be conducted since the experiments also indicates that this algorithm has a high potential with respect to the current algorithm. Unfortunately, the research time was too limited to fully optimize the model causing that conducting further research will be useful.

7.4 Further research

In this research, a new algorithm has been implemented in the model in Python, however, the algorithm can still be improved. In this section, some suggestions regarding further research will be discussed.

Firstly, phase 2 of the new algorithm that constructs the initial solution works in general very well. However, in every experiment that included vehicle dataset 1, the solution of phase 1 was extremely deteriorated by phase 2 of the algorithm. This was also the case for instances 8 and 13 in Table 18. In case the old initial solution creation method of Districon was used, this problem did not occur. Therefore, the first suggestion for further research is to investigate why this problem occurs and how it could be solved in order to let the algorithm of the initial solution work for every case or a new initial solution creation method could be constructed.

Moreover, the goal of the proposed initial solution was to find the smallest fleet of vehicles with the highest truck capacity utilization. In case it is cheaper to use 2 smaller trucks instead of 1 big vehicle, the post-optimization method would compensate for this. However, based on the experiments the best creation of an initial solution has been made by the random + phase 2 method (see Section 6.3.2). For this reason, the solution that will function as the starting solution of the post-optimization method will not be per definition be the smallest fleet of vehicles. As a result, the post-optimization method could become more or less useless if the starting solution of the post-optimization method already contains many small vehicles. Therefore, the second suggestion for further research will be to investigate a new post-optimization method.

Finally, the last suggestion for further research to Districon is the extension of the new model to a multi-depot VRPTW with heterogeneous vehicles. This suggestion has been made since Company A declared that they are serving customers in multiple regions that all have their own depot. For this reason, it might be interesting from a business point of view to extend the model such that it can solve VRPTW with heterogeneous vehicles that involve multi-depots. Besides, this extension could be used for future projects as well, since many companies that consult Districon have multiple depots.

8. References

Bernal, J., Escobar, J. W., & Linfati, R. (2017). A granular tabu search algorithm for a real case study of a vehicle routing problem with a heterogeneous fleet and time windows. Journal of Industrial Engineering and Management, 10(4), 646-662.

M. Boujlil and S. Lissane Elhaq, "The vehicle routing problem with Time Window and Stochastic Demands(VRPTW-SD): Review," 2020 IEEE 13th International Colloquium of Logistics and Supply Chain Management (LOGISTIQUA), 2020, pp. 1-6, doi: 10.1109/LOGISTIQUA49782.2020.9353927.

Burke, E. K., Burke, E. K., Kendall, G., & Kendall, G. (2014). Search methodologies: introductory tutorials in optimization and decision support techniques. Springer.

Caceres-Cruz, J., Arias, P., Guimarans, D., Riera, D., & Juan, A. A. (2014). Rich vehicle routing problem: Survey. ACM Computing Surveys (CSUR), 47(2), 1-28.

Chen, J. (2021, February 28). Heuristics. Investopedia. https://www.investopedia.com/terms/h/heuristics.asp

Cooper, D. R., & Schindler, P. S. (2013). Business Research Methods, 12th Edition (12th ed.). McGraw-Hill Education.

Cordeau, J. F., Gendreau, M., & Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks: An International Journal, 30(2), 105-119.

Cordeau, J. F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational research society, 52(8), 928-936.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. Management science, 6(1), 80-91.

Dixit, A., Mishra, A., & Shukla, A. (2019). Vehicle routing problem with time windows using meta-heuristic algorithms: a survey. In Harmony search and nature inspired optimization algorithms (pp. 539-546). Springer, Singapore.

Glover, F. (2015, April 27). Metaheuristics - Scholarpedia. Scholarpedia. http://www.scholarpedia.org/article/Metaheuristics#Definition

Glover, F. (1990). Tabu search: A tutorial. Interfaces, 20(4), 74-94.

Goel, Rajeev & Bansal, Sandhya. (2019). Hybrid algorithms for rich vehicle routing problems: a survey. 10.1016/B978-0-12-815715-2.00011-7.

Hao, P., Wang, Z., Wu, G., Boriboonsomsin, K., & Barth, M. (2017, October). Intra-platoon vehicle sequence optimization for eco-cooperative adaptive cruise control. In 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC) (pp. 1-6). IEEE.

Jiang, J., Ng, K. M., Poh, K. L., & Teo, K. M. (2014). Vehicle routing problem with a heterogeneous fleet and time windows. Expert Systems with Applications, 41(8), 3748-3760.

Jourdan, L., Basseur, M., & Talbi, E. G. (2009). Hybridizing exact methods and metaheuristics: A taxonomy. European Journal of Operational Research, 199(3), 620-629.

Khachay, Michael & Ogorodnikov, Yuri. (2019). Polynomial-Time Approximation Scheme for the Capacitated Vehicle Routing Problem with Time Windows. Proceedings of the Steklov Institute of Mathematics. 307. 51-63. 10.1134/S0081543819070058.

Kramer, A., Lalla-Ruiz, E., Iori, M., & Voß, S. (2019). Novel formulations and modeling enhancements for the dynamic berth allocation problem. European journal of operational research, 278(1), 170-185.

Kramer, R., Subramanian, A., Vidal, T., & Lucídio dos Anjos, F. C. (2015). A matheuristic approach for the pollution-routing problem. European Journal of Operational Research, 243(2), 523-539.

Laguna, M. (2000). Global optimization and meta-heuristics.

Lahyani, R., Khemakhem, M., & Semet, F. (2015). Rich vehicle routing problems: From a taxonomy to a definition. European Journal of Operational Research, 241(1), 1-14.

Lalla-Ruiz, E., & Voß, S. (2020). A popmusic approach for the multi-depot cumulative capacitated vehicle routing problem. Optimization Letters, 14(3), 671-691.

Lalla-Ruiz, E., & Voß, S. (2020). A popmusic approach for the multi-depot cumulative capacitated vehicle routing problem. Optimization Letters, 14(3), 671-691.

Laporte, G., Gendreau, M., Potvin, J. Y., & Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. International transactions in operational research, 7(4-5), 285-300.

Lenstra, J. K., & Kan, A. R. (1981). Complexity of vehicle routing and scheduling problems. Networks, 11(2), 221-227.

Liu, F. H., & Shen, S. Y. (1999). The fleet size and mix vehicle routing problem with time windows. Journal of the Operational Research society, 50(7), 721-732.

Molina, J. C., Salmeron, J. L., & Eguia, I. (2020). An ACS-based memetic algorithm for the heterogeneous vehicle routing problem with time windows. Expert Systems with Applications, 157, 113379.

Noon, C. E., & Bean, J. C. (1993). An efficient transformation of the generalized traveling salesman problem. INFOR: Information Systems and Operational Research, 31(1), 39-44.

Paraskevopoulos, D. C., Repoussis, P. P., Tarantilis, C. D., Ioannou, G., & Prastacos, G. P. (2008). A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows. Journal of Heuristics, 14(5), 425-455.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. European Journal of Operational Research, 225(1), 1-11.

Salhi, S., Imran, A., & Wassan, N. A. (2014). The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation. Computers & Operations Research, 52, 315-325.

Semet, F., Toth, P., & Vigo, D. (2014). Chapter 2: Classical exact algorithms for the capacitated vehicle routing problem. In Vehicle Routing: Problems, Methods, and Applications, Second Edition (pp. 37-57). Society for Industrial and Applied Mathematics.

Vidal, Thibaut & Crainic, Teodor Gabriel & Gendreau, Michel & Prins, Christian. (2012). Heuristics for Multi-Attribute Vehicle Routing Problems : A Survey and Synthesis. European Journal of Operational Research. 231. 10.1016/j.ejor.2013.02.053.

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

Appendices

Appendix A: Taxonomies

This appendix includes the taxonomies used in the literature review in Chapter 2. In Appendix A-1, the taxonomy of rich vehicle routing problems can be found that is used to explain multiple variants of vehicle routing problems in the Section 2.1. The taxonomy in Appendix A-2 has been used in to provide an overview of the solution methods available for solving a vehicle routing problem. In Section 2.2, some of these methods are elaborated on.

Appendix A-1: Taxonomy of Rich Vehicle Routing problems

1 Scenario characteristics

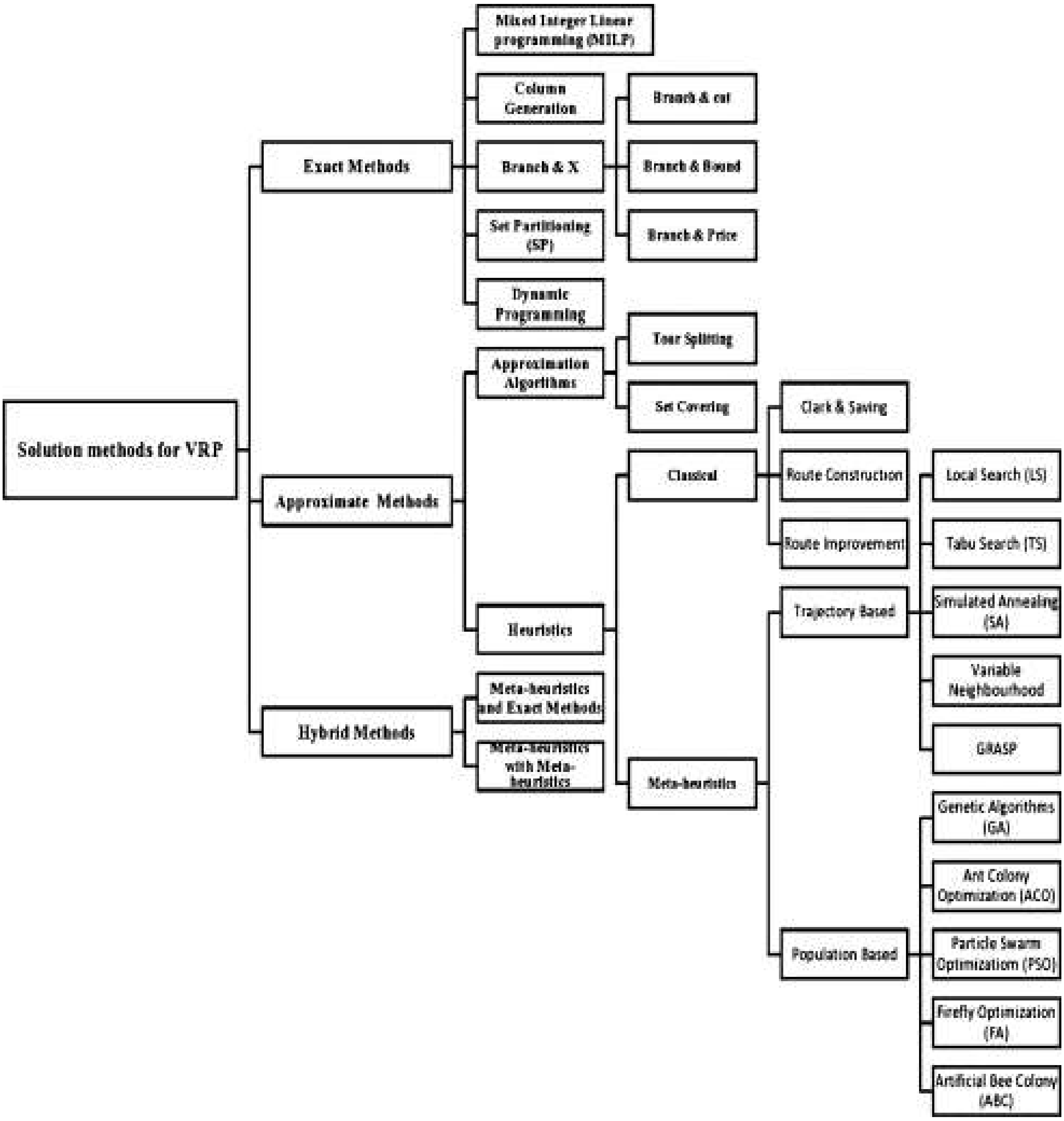
- 1.1 Input data
  - 1.1.1 Static
  - 1.1.2 Dynamic
  - 1.1.3 Deterministic
  - 1.1.4 Stochastic
- 1.2 Decision management components
  - 1.2.1 Routing
  - 1.2.2 Inventory and routing
  - 1.2.3 Location and routing
  - 1.2.4 Routing and driver scheduling
  - 1.2.5 Production and distribution planning
- 1.3 Number of depots
  - 1.3.1 Single
  - 1.3.2 Multiple
- 1.4 Operation type
  - 1.4.1 Pickup or delivery
  - 1.4.2 Pickup and delivery
  - 1.4.3 Backhauls
  - 1.4.4 Dial-a-ride
- 1.5 Load splitting constraints
  - 1.5.1 Splitting allowed
  - 1.5.2 Splitting not allowed
- 1.6 Planning period
  - 1.6.1 Single period
  - 1.6.2 Multi-period
- 1.7 Multiple use of vehicles
  - 1.7.1 Single trip
  - 1.7.2 Multi-trip

2 Problem physical characteristics

- 2.1 Vehicles
  - 2.1.1 Type
    - 2.1.1.1 Homogeneous
    - 2.1.1.2 Heterogeneous
  - 2.1.2 Number
    - 2.1.2.1 Fixed
    - 2.1.2.2 Unlimited
  - 2.1.3 Structure
    - 2.1.3.1 Compartmentalized
    - 2.1.3.2 Not compartmentalized
  - 2.1.4 Capacity constraints
  - 2.1.5 Loading Policy
    - 2.1.5.1 Chronological order
    - 2.1.5.2 No policy
  - 2.1.6 Drivers regulations
- 2.2 Time constraints
  - 2.2.1 Restriction on customer
  - 2.2.2 Restriction on road access
  - 2.2.3 Restriction on depot
  - 2.2.4 Service time
  - 2.2.5 Waiting time
- 2.3 Time window structure
  - 2.3.1 Single time window
  - 2.3.2 Multiple time windows
- 2.4 Incompatibility constraints
- 2.5 Specific constraints
- 2.6 Objective function
  - 2.6.1 Single objective
  - 2.6.2 Multiple objectives

Source: Lahyani, Khemakhem and Semet (2015)

Appendix A-2: Taxonomy of vehicle routing solution techniques



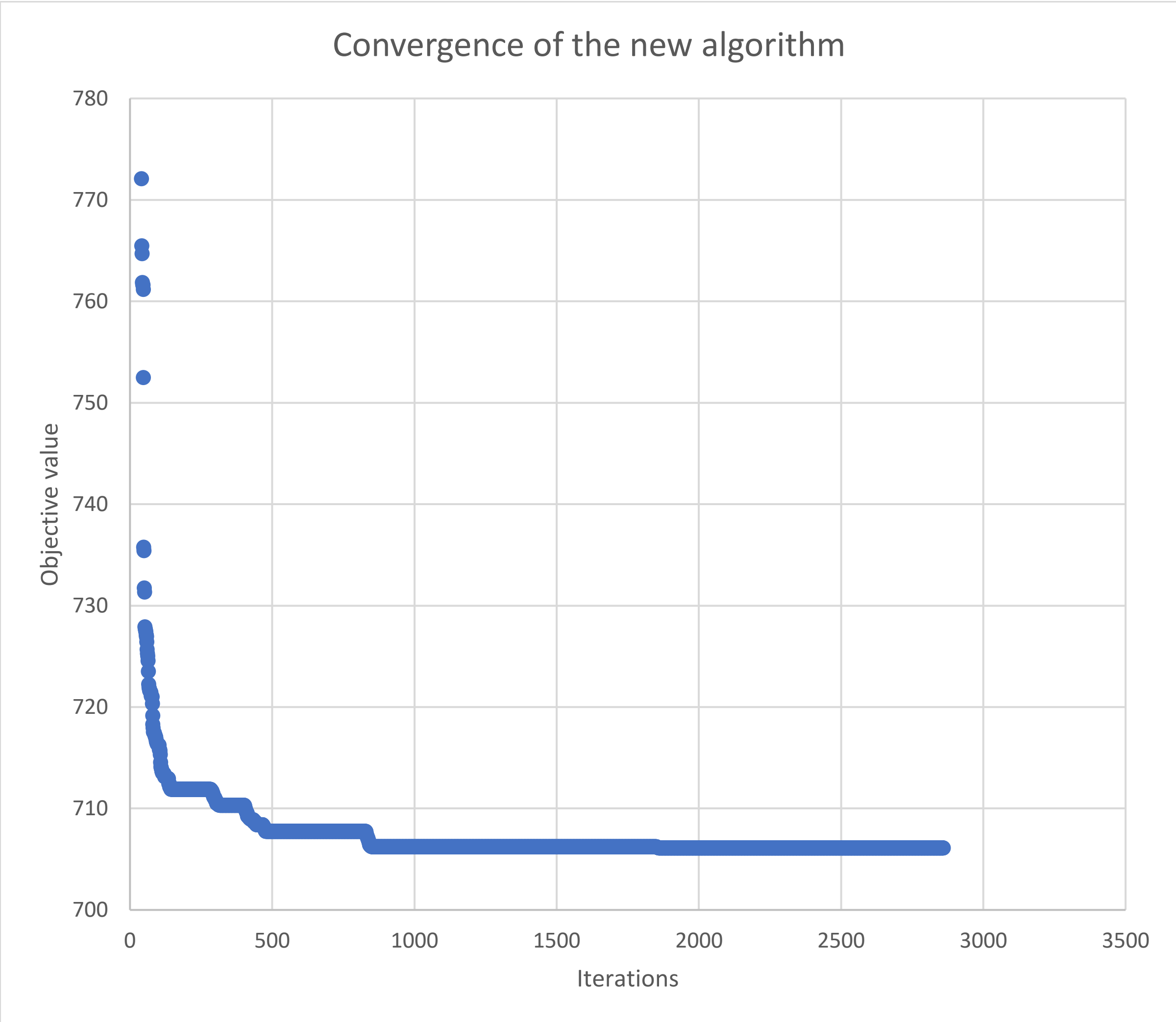
Source: Goel et al. (2019)

Appendix B: Data analyse

Appendix C: Experiments

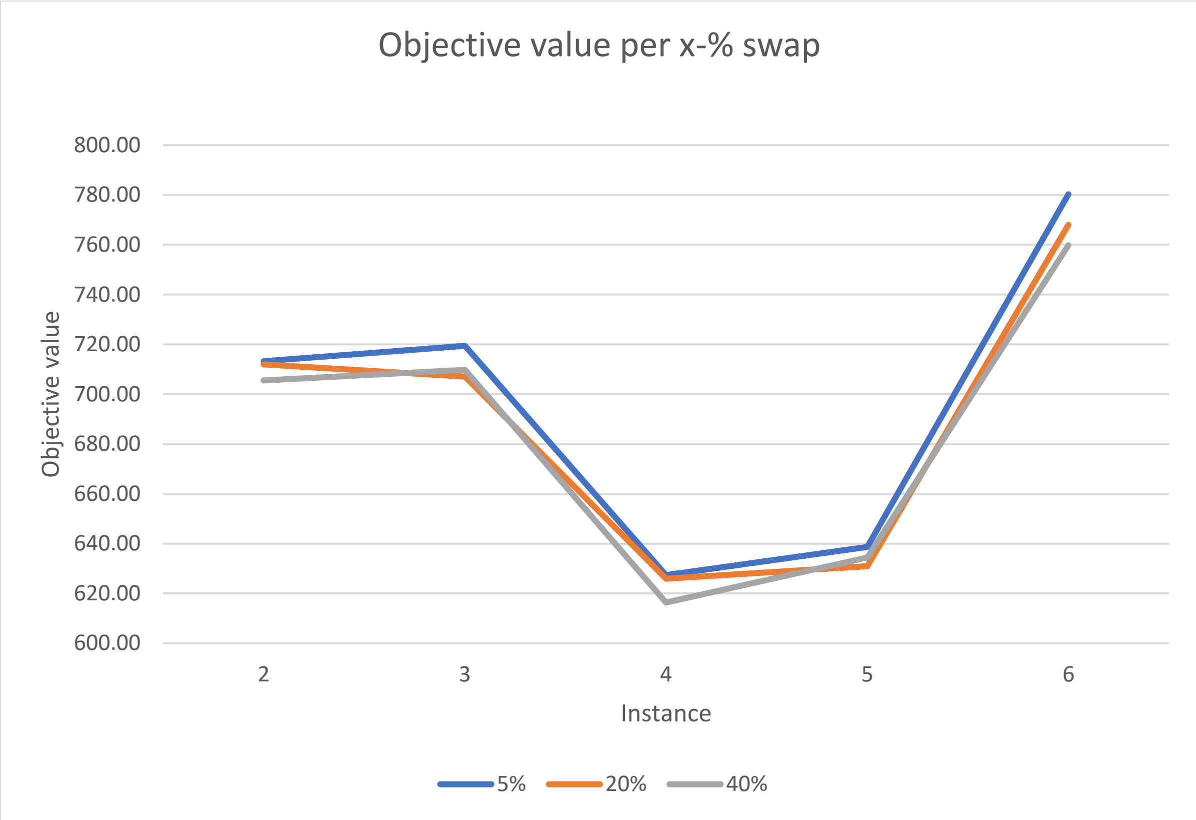
In this appendix, graphs that were used to analyse the experiments are presented. In Appendix C-1, the full convergence of the new algorithm where the objective value has been plotted against the iteration number. Appendix C-2 includes the graphs that were used to analyse the experiments regarding the x-% swap.

Appendix C-1: Full convergence of the new algorithm



Appendix C-2: Figures X-swap experiments

In this figure, the average objective value of each x-% swap of the experiments has been plotted per instance of vehicle fleets. As mentioned before, instance 1 is not considered for the decision-making process since the data is unusable.



In the next figure, the computational time per x-% swap included in this experiment has been presented for every instance.