Route optimization through integration of different activities at C-mark Eurofins

Author:
M.H.G. Tacken (Mathijs)
s2814307

Supervisors University of Twente:
1. Dr.ir. E.A. Lalla (Eduardo)
2. Dr.ir. J.M.J. Schutten (Marco)

Supervisor C-Mark Eurofins:
M. Eekelder (Mark)

University of Twente

August 18, 2024

BSc Thesis Industrial Engineering and Management

Faculty of Behavioural and Management Science

Preface

Dear reader,

I am pleased to present the thesis in front of you, titled *Route optimization through integration of different activities at C-Mark Eurofins.* This research has been conducted at C-Mark Eurofins in Deventer, the Netherlands, as the final assignment for my bachelor's degree Industrial Engineering and Management at the University of Twente.

I would like to take this opportunity, to express my gratitude to my company supervisor, Mark Eekelder, for allowing and enabling me to conduct this research at C-Mark Eurofins. Mark has provided me with the opportunity to gain numerous new insights, additional knowledge, and experiences for my future. In addition, I want to thank Peter Poll and Bart Nijland, for teaching and showing me the inner workings of C-Mark Eurofins' processes and services and for sharing with me all the necessary information, needed to be able to conduct the investigation and build the different models. Finally, I want to thank the rest of the employees for the enjoyable days at the office. Your company, interest and (moral) support has been highly appreciated.

Of course, I would also like to express my immense gratitude to my supervisor at the University of Twente, Eduardo Lalla, for his availability, help, and, above all, patience. Most of this research was completely new to me, and without his feedback and the expertise he has in this field of study and that he shared with me, I would not have been able to achieve such an end result. I am also grateful to my second supervisor, Marco Schutten, for taking the time to read, provide feedback and evaluate my thesis. Thank you for the hours spent on educating me!

Lastly, I want to thank my family and friends. They were a constant source of support, both during the ups and the downs of my journey to define, execute and document this research.

All in all, I have enjoyed working on this thesis and I am immensely proud of the end result. They often say that "the journey is the destination", which is also true for this thesis. Every time I thought the goal was not achievable, I worked and connected with others to discuss and find a new solution. And although I despaired a couple of times, I learned that every problem has a solution, as long as you do not stop looking for it.

I hope you enjoy reading my thesis as much as I enjoyed writing it!

Mathijs Tacken

Enschede, August 2024

Management Summary

This research focuses on the opportunities of route optimalisation, through the integration of different tasks at C-Mark Eurofins.

The company C-Mark is a part of Eurofins, a world leader in testing food, pharmaceutical and cosmetic products. C-Mark is specialised in clean water, like drinking water, swimming pools and other water sources. They carry out 1.) check valve controls and 2.) control measures, in order to prevent the legionella bacteria to spread. Currently, these tasks are planned individually, which potentially could result in inefficiencies in terms of time, resources, and costs.

Therefore, the goal of this research is to determine if and how the combination of check valve controls and control measures related activities could decrease costs at C-Mark. To reach this goal, the following research question was formulated:

How can the control measures and check valve controls be jointly planned to decrease costs at C-Mark (Eurofins)?

To answer this question, first a thorough context analysis has been conducted, in order to understand and describe the current planning practices. Also, the most important Key Performance Indicators (KPIs) were identified, being cost, route efficiency and resource utilization, so that the aim of the investigation could better be defined. The quality of the data was checked, and the data was cleansed, to provide the right input data for the optimization tool. Also, the problem itself was examined in this context analysis. Problem characteristics such as the "homogeneous fleet", the fact that mechanics start from their house, and the need for the use of "time windows", were identified. Finally, it is important that the salary of control measure mechanics is less than that of check valve mechanics. When a control measure mechanic must perform both tasks, they receive the same salary as a check valve mechanic.

Based on the problem characteristics, found in the context analysis, we identified the complexity of the scheduling problem at C-Mark as a "Multi-Depot Vehicle Routing Problem with Time Windows" (MDVRPTW). This is based on the problem characteristics, like the fact that there are multiple depots (the homes of mechanics) from which vehicles are starting, and that there are certain time windows that need to be considered, in which a set of services should take place. The MDVRPTW is an extension of the Vehicle Routing Problem (VRP), which is a commonly used approach to optimise routes.

Once the problem was identified as a MDVRPTW, different heuristics and meta heuristics have been examined. Heuristics are problem-solving approaches that can be applied to such routing problems. Heuristics provide more systemic ways to come to a near optimal solution, than " simply" exploring all potential combinations of routes. Given the number of potential solutions, the use of meta heuristics is necessary in order to be able to keep the calculation time of planning within reasonable boundaries. We chose the Iterated Local Search (ILS), because it is most commonly used heuristic in MDVRPTW problems.

The ILS method works by first searching for a local optimum and then perturbing the solution to prevent the algorithm from getting stuck in this local optimum. Doing this in a number of iterations makes it suitable for tackling the complex problem of C-Mark. During the testing phase of the ILS, it became clear that the number of locations (mechanics and customers), caused even the ILS to take too much run time. Therefore, we introduced the Reduced Local Search (RLS), which does not iterate for every possible move, but does so for a chosen number of swaps and moves.

The evaluation phase involved conducting experiments with different data instances for parameter tuning, method selection and generating results. Based on the results of the parameter tuning experiments for the RLS, the perturbation strength was set to 5, the iterations to 20, number of swaps to 20 and the number of moves to 100, to ensure the best results in a reasonable run time. The ILS and RLS were tested against each other, during the method selection experiments. It turned out that, even with the perturbation strength of 5 and iterations of 20, the ILS had a too long run time. Therefore, the RLS was chosen as the preferred method for generating the results. With the RLS the total cost, total distance, total duration and run time for 3 different instances and 3 different days are calculated. Where Dcv is the check valves scenario, Dcm the control measure scenario and Dc is the combined scenario. When adding the check valve scenario and the control measure scenario and subtracting the combined scenario the following results were found:

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)
D1cv + D1cm	2030.48	1851.78	4788.17
D1c	2134.63	1757.8	4712.99
D1cv + D1cm - D1c	-104.15	93.98	75.18

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)
D2cv + D2cm	1638.15	1088.34	3329.18
D2c	1741.23	1012.92	3268.84
D2cv + D2cm - D2c	-103.08	75.42	60.34

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)
D3cv + D3cm	1750.24	1031.21	3682.48
D3c	1884.34	887.76	3567.73
D3cv + D3cm - D3c	-134.1	143.45	114.75

The results indicate that the combination of tasks result in higher cost. The cost increase is mainly due to the increase in salary cost, from check valve mechanic to a mechanic that can perform both tasks. Specifically, this adds an additional 36.31 euro per day per mechanic. For the first scenario this means an extra cost of 145.24 per day for 4 mechanics. Despite the savings from the total distance, 41.09 euro, the costs are still negative.

With the decrease in duration, 97.88% of the mechanics are needed for the same work, which translates to a potential saving of 38.49 euros in salary costs per day. Instead of having one mechanic work less, it can be also argued that 2.12% more customers can be served. By doing this, C-Mark can help approximately 0.4 customers per day extra. Since we do not know the average revenue per customer, it is up to C-Mark to consider whether this increase makes combining customers cost-efficient.

Based on these findings, it is recommended to keep the tasks separated. The combination of tasks leads to higher costs due to the increased salary costs for control measure mechanics to a mechanics who handle both control measures and check valve controls. Even with one mechanic working less hours a day, the savings are not high enough to compensate for the higher salary costs. Keeping the tasks separated also prevents the need to search for new mechanics or retrain control measure technicians, as this could entail additional costs that are not yet included in the model. This approach also saves the extra salary costs that would have to be paid if the tasks were combined.

However, the time saved can be translated into approximately half an extra customer per day. It is up to C-Mark to consider whether the revenue from this additional customer is high enough to make the combination of tasks ultimately cost-efficient.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

This chapter introduces the company C-Mark Eurofins, where the research has been conducted. In Section 1.1 the company is introduced and described. Section 1.2 elaborates the current planning practice and Section 1.3 elaborates on the problem identification. In Section 1.4 the research question and its knowledge questions are formulated and explained

1.1 Company description

The company for which the research is done is called C-Mark and is part of Eurofins. Eurofins is a world leader in food, pharmaceutical and cosmetic product testing. Within this large international company, C-Mark specializes in ensuring clean water. Clean water in this respect encompasses drinking water, swimming pools, hotels baths and more. C-Mark has a clear mission: to contribute to health and safety by providing high quality consulting and laboratory services in the field of water.

To ensure this safety, C-Mark is specialized in sampling and analysis of all kinds of water. One of the main goals of this testing is to prevent the spread of legionella. Legionella is a bacterium that can develop and spread in pipelines under certain conditions, such as stagnant water and temperatures between 25 and 55 degrees Celsius. The legionella bacteria can cause pneumonia, which in 2-10% of cases results in death, especially under elderly and people with underlying health issues. To prevent legionella from developing and spreading in pipelines, C-Mark maintains and controls two specific tasks for their customers, namely 1. check valve controls and 2. control measures. These tasks are explained below:

- Check valve controls ensure that drinking water can only flow in one direction. These check valves must be applied in every source of the collective drinking water installation, to ensure that any infected or contaminated water cannot flow back into the main water pipe and spread in this way. These valve checks take 6 minutes per valve and need to be performed yearly. C-Mark Eurofins employs mechanics whose task it is to execute these controls.
- The control measures consist of two different tasks: being flushing taps and temperature controls.
    - Tap flushing involves opening taps that are not or rarely used. The reason is that these taps contain stagnant water, and with the right temperature, legionella can develop. Preventive flushing makes it much harder for legionella to develop and spread. Flushing should be done weekly, to prevent legionella most effectively, yet there are companies that choose to flush every two weeks.
    - Temperature checks in turn ensure that legionella cannot develop, or at least more difficult or slowly. These temperature checks are performed monthly. Tap flushing and temperature check take both individually take 2 minutes per tapping point to perform. For control measures, C-Mark Eurofins also employs mechanics that perform these tasks.

1.2 Current planning practice

As explained above, C-Mark Eurofins has two different types of activities. These activities are currently conducted completely separately. There are different planners responsible for planning the check valve mechanics and for planning the control measure mechanics. Also, these tasks are done by different mechanics.

The frequency on how often the check need to be performed depends on the task. The check valves need to be checked yearly, as this is required by the government, legal standard in NEN 1006. The control measures need to be performed weekly or bi-weekly (flushing) or monthly (temperature check).

The planner of the check valve control currently plans manually. To do this, he uses a big excel file with all the customer addresses, the number of check valves per location and the date of the last check. Next to this, he has an excel sheet which he uses to plan three weeks in advance for the different mechanics. He mails every customer in advance, so the customer can accept the date.

When planning, the planner must take some variables into consideration. Since a check must be done every year, the planner uses the last date the check valves were checked, to determine the ultimate date for the new check. The distance that has to be driven is minimized, by selecting the mechanic who lives closest to the customer's location. A constraint is the amount of time a mechanic has available during a workday. The travel time between customers and the service time at every customer cannot exceed the daily working hours of the mechanic. Finally, the last constraint is whether or not the customer accepts the visiting date. When the customer accepts the date, the plan is finalized.

The plan of the control measures is different. Since the control measures need to be performed weekly, bi-weekly, or monthly, currently the plan is simply the same for every week. Of course, the actual constraints should be tailored to the number of taps, the frequency of the control measures for that customer and the throughput time of each of the control measures. This then needs to be optimised based on the proximity of the customers and the location of each of the mechanics, within the available working hours of the mechanics. Again, currently planning is done manually, and optimisation is done by looking at which mechanic lives nearest to the customer.

1.3 Problem definition

1.3.1 Problem cluster

Figure 1 shows the problem cluster, where tasks are performed individually, and manual planning (core problem) is the cause of the higher costs (action problem).

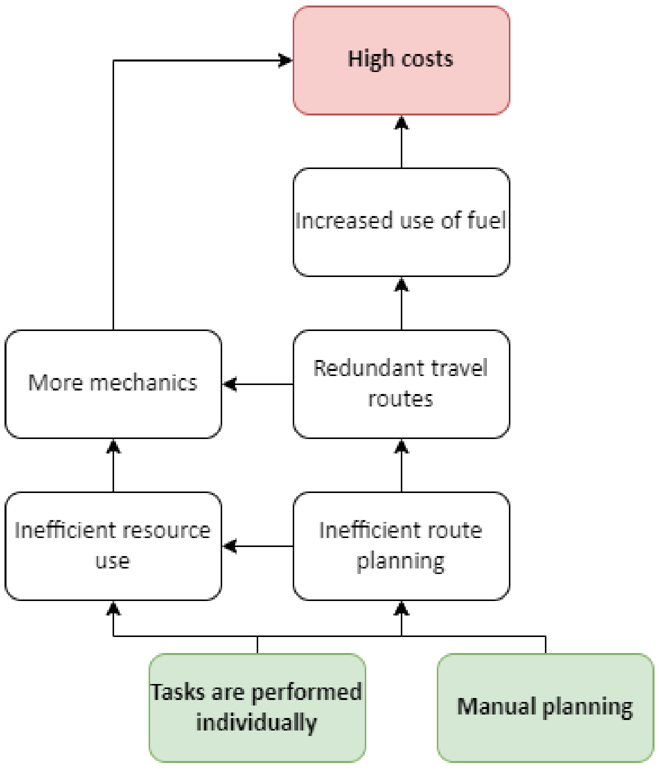


Figure 1 - Problem Cluster of C-Mark Eurofins

This problem cluster helps to see what aspects result in high costs. The problem cluster above shows the action problem its core problems. As seen, there are two problems that are considered to be the core problems, tasks are performed individually and manual planning. Both results in inefficient route planning and inefficient resource use. Inefficient resource use refers to inefficiency in time, employment, and resources, which all result in high costs. With the inefficient route planning redundant travel routes are used, this results in increased use of fuel, which also results in high costs.

An extra positive impact of this research would include the decrease of CO_2 emissions and therefore having a positive impact on the environment.

1.3.2 Core problem

This research focusses on the core problems “tasks are performed individually” and “manual planning.”. As these are the core problems of the problem stated by C-Mark, this research focusses on creating a tool that calculates costs and gives the near optimal routes for a certain day, with regards to the combination of tasks.

The before mentioned planning practice clearly shows that each mechanic only performs either the control measures or the check valve controls, which could create a suboptimal use of these mechanics. Since these tasks are done individually by different mechanics, this leads to inefficiencies in time, resources, costs, as it could be more efficient when a check valve mechanic also does the control measures at a certain location. For example, a check valve mechanic goes to the University of Twente to do a check valve control and the same day a control measure mechanic needs to be at the University of Twente. This is very inefficient as the first mechanic also could have done the control measures in one go. This would have saved resources and environmental impact as every mechanic is not limited to certain customers at a certain frequency and has the ability to also take other customers into account.

Next to this, manual planning also causes inefficient use of resources and results in redundant travel routes. This is because manual planning is unlikely to generate optimal routes for the mechanics. A planner could take some factors regarding starting location of mechanics into consideration but is likely to overlook better possibilities.

This gives our core problem, which is that the tasks are being done individually by mechanics *(tasks are performed individually)* and that the routes are planned manually (*manual planning*).

1.4 Research question

The research question can be derived from the action problem and core problem, with the context of the current situation at C-Mark. The goal of this research is to calculate if combining the tasks and optimizing routes can reduce costs. Therefore, the research question is formulated as follows:

How can the control measures and check valve controls be jointly planned to decrease costs at C-Mark (Eurofins)?

To answer the main research question, the following knowledge questions should be answered:

1. *How are the routes currently planned at C-Mark Eurofins?*
    - *What are important factors that influence the route planning?*
    - *What are the most relevant key performance indicators (KPIs) when planning?*
    - *What is the quality and structure of the data used in the route planning process?*

To start working on the theoretical framework and creating a tool, first it is necessary to gain a better insight into the current situation at C-Mark Eurofins. To do so, a context analysis is conducted. The goal of this analysis next to creating a better insight on how the routes are currently planned, is to analyse what considerations and constraints are involved, and what important decision-making factors there are. This analysis provides the context for understanding the problem to design and develop the proper approach in optimizing the route planning process for C-Mark Eurofins.

2. *What route optimization methods are possible for the problem stated by C-Mark Eurofins?*
   - *What type of optimization problem is stated by C-Mark Eurofins?*
   - *What are available methods for route optimization?*
   - *Which method is most suitable for the problem stated by C-Mark Eurofins?*

After the context analysis, a literature study is conducted to consider different methods and theories regarding route optimization. The objective is to describe and understand different methods, these methods are described in a theoretical framework. By doing the literature review insights on the best method is created and assessed for the suitability of the case at C-Mark Eurofins. The best method for the specific context at C-Mark Eurofins is selected.

3. *What should the solution approach look like?*
   - *What are the key components of a mathematical model for route optimization problems?*
   - *What kind of input data is needed?*
   - *What algorithm should be used?*
   - *What considerations regarding usability should be made?*

The objective of the solution design is to make a mathematical model and make a tool that optimizes the plan using a heuristic. This includes a mathematical representation of the variables, constraints, and objective function. Other factors emerge during the context analyses. The mathematical model gives a clearer picture of the problem. Next to the mathematical model, the tool is described and explained. What input data is needed and what is the expected output. Lastly the usability should be considered as someone that has not a lot of programming experience should be able to understand the code and use it.

4. *How to determine the best parameters and the best optimization method to generate the results?*

As the results should be reliable and trustworthy, it is important to know whether the solution provided form the tool is near optimal. That is why the solution should be evaluated carefully. If C-Mark is satisfied, recommendations and conclusions based on the tool can be made. If C-Mark is not satisfied or has additional remarks than it is necessary to go back to step 4, which means that the tool has to be revised, modified, and tested again.

5. *What recommendations and conclusions can be made from the evaluation of the tool?*
   - *How do the results compare to the current situation?*
   - *What are advantages and disadvantages of combing the tasks?*
   - *What limitations does the tool have?*

After the results of the tool are evaluated carefully, conclusions can be made. With these conclusions recommendations regarding the combination of tasks can be made. The recommendations should be clear and easily understood for C-Mark Eurofins.

With the research approach and sub research questions a flowchart is made which gives a clear overview of the research design. The flowchart can be found in the appendix.

2 Context analysis

The goal of this chapter is to highlight and determine relevant and important issues concerning the problem stated by C-Mark Eurofins. These issues are needed when creating the solution model in Section 5. In this chapter the following research questions are answered.

How are the routes currently planned at C-Mark Eurofins?

- What are important factors that influence the route planning?
- What are the most relevant key performance indicators (KPIs) when planning?
- What is the quality and structure of the data used in the route planning process?

The information concerning the context analyses has been gathered by interviewing and meeting with the planners, and by having conversations with other employees of C-Mark Eurofins. In Section 2.1 the current planning practices are explained. Section 2.2 explains the C-Marks KPIs, the next Section 2.3 the data is analysed, cleaned, and checked on quality. Section 2.4 explains the problem characteristics. Section 2.5 explain the constraints and Section 2.6 explain what the costs are based on. Lastly in Section 2.7 gives a conclusion based on the knowledge questions for this section.

2.1 Current planning

C-Mark has two different types of activities they need to perform, one being the control measures and one being the check valve controls. These activities are both planned by different planners and these tasks are done by individual mechanics.

How often the tasks need to be performed depends on the task. The check valves need to be checked yearly and the control measures need to be performed weekly or bi-weekly (flushing) or monthly (temperature check). As discussed before, the current route plan is made manually. The planner for the check valve control tries to take optimalisation into account, using his experience and taking important factors into account. The plan for the control measures is the same for each week. Both plannings practices are evaluated carefully.

2.1.1 Check valve control planning

The check valve control plan is made in an Excel spreadsheet where every mechanic is scheduled, based on the following factors: previous check, mechanic's residential city, and working hours. In Figure 2, the current planning method for the check valve control is visualised for better understanding.

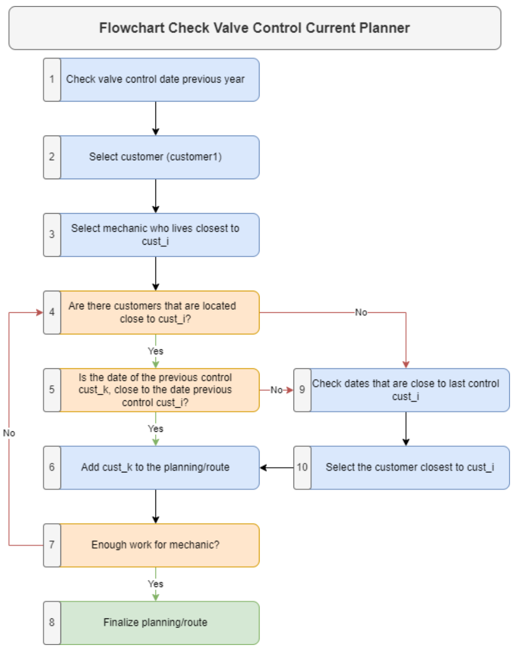

Flowchart Check Valve Control Current Planner
1 Check valve control date previous year
2 Select customer (customer1)
3 Select mechanic who lives closest to cust_i
4 Are there customers that are located close to cust_i?
No
Yes
5 Is the date of the previous control cust_k, close to the date previous control cust_i?
No
9 Check dates that are close to last control cust_i
Yes
No
6 Add cust_k to the planning/route
10 Select the customer closest to cust_i
7 Enough work for mechanic?
Yes
8 Finalize planning/route

Figure 2 - Flowchart check valve control current planner

The current planner uses an excel spreadsheet with all the customers who have the check valves controlled and schedules the mechanics 3 weeks in advance to ensure that the customers have enough time to accept the planned day. The planner first (1) picks the next customer, based on the last control. (2) The customer (cust_i) is selected, (3) the mechanic who lives the closest to the customer is selected. It then (4) checks whether any other customers are close to cust_i, if this is the case it (5) the planner checks whether their last check of cust_k is close to last check cust_i. This last check of cust_k must be within 1 to 2 weeks of cust_i, otherwise they are too far apart to place in the same route. If the last check is not within 1 to 2 weeks, (9) dates that are close to previous check cust_i must be checked and (10) the customer that is located the closest to cust_i is selected and added to the route.

If there is a customer who meets these two requirements, near location of cust_i and near previous check cust_i, they can be (6) added to the plan/route. If not, (9) dates that are close to previous check cust_i must be checked and (10) the customer that is located the closest to cust_i is selected and added to the route.

Next, it is important that (7) a mechanic has neither too few nor too many customers so that he can complete his workday within the given time. If the mechanic has too little work, the planner goes back to the step whether there are customers close to cust_i or cust_k and this continues until there is enough work. If the mechanic has enough work for a given day, (8) the schedule can be finalized, and confirmation emails can be sent to customers. By considering different important optimization strategies the planer does try to take optimality into account, however it cannot be said that the route made by the planner is optimal.

2.1.2 Control measures planning

The control measures plan is also based on the mechanic's residence. The mechanic performs the control measures closest to the mechanic's residence. The route the mechanic takes to perform the control measures is made by the mechanics themselves and is the same for each week. When an additional customer is added, the mechanics places this customer into a possible gap in the plan. Like with the check valve plan, this plan is made as good as possible by taking optimization strategies into account. This does not mean the plan is optimal.

2.2 Key Performance Indicators

Key Performance Indicators (KPIs) are important to indicate the efficiency and effectiveness of specific processes. With the KPIs, the solution model can be evaluated on efficiency and effectiveness. The following KPIs are used to assess route optimization:

2.2.1 Cost

The most important KPI in this research is, cost, as the main focus of this research and the goal of C-Mark is to decrease cost. The tool gives the cost of a certain day, which should be as low as possible. The tool can then be assessed on efficiency and effectiveness. To ensure cost reduction, two other important route optimizations KPIs need to be considered, namely route efficiency and resource utilization.

2.2.2 Route efficiency

Route efficiency is based on total distance and total driving time. To reduce costs route efficiency is important as in general limiting the distance and driving time results in lower fuel cost. This can be assessed by minimizing the route length taken by the mechanics. This is one of the main costs a mechanic makes during a working day.

2.2.3 Resource utilization

Like route efficiency, is resource utilization an important KPI for route optimization as it assesses the efficiency of the deployment of mechanics. The deployment of mechanics has also an impact on the main KPI, cost reduction since the mechanics need to be paid a salary.

2.3 Data

The data used in this research is provided by the planners. The datasets are large Excel sheets containing a lot of different values: one relates to the data concerning check valves and the other concerning the control measures data. In this section, the data is analysed, cleaned, and described, with the main goal of ending with the dataset which is used in the solution design.

2.3.1 Check valves data

The first excel file includes all the customers, their locations, number of check valves, time per location, previous check, and more. The excel spreadsheet includes a lot of data that is not necessary. Therefore, a lot of data is removed as it does not contain important information for the route optimization.

The data that is important are the Customer, Number of hours (service time), address (latitude and longitude) and the time window (if applicable). If there are time windows for certain customers, these are stated in the column with comments.

The number of customers per day depends on the service time of the customers and the number of mechanics that work that day. Where the service time for some customers might be only a few minutes, while for other customers it could take the whole day. Therefore, it is hard to say how many customers there are per day, however the average can be calculated by dividing the number of check valve customers by the number of working days, 4600/260 is 17.7 customers per day. Of all these customers, about 10% have a time window. These time windows can be at any time of the day between 8 and 5 o'clock.

The cleaned data gives all the relevant information regarding the customers location, service time and time windows. It is possible to start optimizing the routes when this data is imported in the solution model.

2.3.2 Control measures data

The data from the control measures contains, just like the check valve data, data that is not necessary for route optimization. That is why these are removed from the data. Also, not all addresses contain a certain service time. This service time can be calculated by multiplying the number of flushing tasks times the time that a flushing task takes. If there is missing data, this can be indicated to the planner so that it can be corrected. There are also a number of mechanics that are excluded from this research, due to the fact that the customers of these mechanics are fixed or are not managed by C-Mark. Therefore, these are removed from the data. Finally, the data contains customers who are no longer customers of C-Mark, so these are also removed from the data. This ensures that the data looks the same as the check valve data.

2.4 Problem characteristics

To determine which heuristic is best for the problem stated by C-Mark, all the problem characteristics need to be evaluated and described. In this section the problem characteristics: vehicles, mechanics, customers, and extra characteristics are described.

2.4.1 Vehicles

The vehicles used by the mechanics have identical characteristics, also known as a homogeneous fleet. Every mechanic has an Opel Combo as company vehicle. This Opel Combo is leased by the company, which costs 700 euro per month. The combined fuel consumption in the city and outside built-up areas is 4.2 l/100km (Opel, 2019). This value is used to calculate the fuel cost.

2.4.2 Mechanics

In total there are 18 different mechanics, 12 of these currently work as check valve mechanics and 6 work currently as control measure mechanics.

The residences of these mechanics are used as so-called depots. As there are 18 mechanics, there are 18 depots where routes start and finish. The number of mechanics per day depends, as not every mechanic works five days a week and sometimes mechanics are sick or are on vacation. The 18 mechanics can be deployed to perform both tasks. If a mechanic works, the mechanic works 9 hours for that day. The mechanics start driving from his home at 8:00 and should be back at his home at 5:00. The salary of a mechanic is the same for every mechanic, namely 2800-euro gross per month.

To use the mechanics characteristics, a data list of the mechanics and their residence is used in the solution model. These mechanics are referred to as there residential city.

2.4.3 Customers

C-Mark has many different customers that have check valve controls and control measures performed, amusement parks, holiday parks, hotels, restaurants and more. The locations of these customers are all around the Netherlands.

As the tool is using route optimization to optimize the current daily plan, the number of customers differs from day to day. To optimize the plan, the locations of the customer, the service time at the customer location and the time windows are needed.

2.4.4 Extra characteristics currently

In this thesis, the combination of tasks, the problem characteristics described above apply. However, to give recommendations on the combination of tasks the current practice should be compared with the new practice. To make this current practice, some additional characteristics should be described regarding the control measure mechanics and their cars.

The control measure mechanics currently get paid 2400 euro per month, which is 400 euro less than the salary of the mechanic after the combination of tasks. Additionally, the control measure mechanics have a different car which costs 400 euro to lease instead of 700 euro. These characteristics are important to simulate the current situation, as when combining the tasks the mechanics costs a total of 700 euro (400 salary and 300 lease) more than before. This means that in total a control measure mechanic cost 2800 euro per month and a check valve or combined mechanic costs 3500 euro per month.

The control measure mechanics drive a Peugeot 108 which has the same fuel consumption as the Opel, which means no extra characteristic on this vehicle should be described.

2.5 Constraints

It is necessary to identify constraints, to start working on the mathematical model and the tool. Constraints function as boundaries for problems and are important in making the mathematical model, and the tool. The following constraints have been found and are described.

2.5.1 Working hours

The mechanics have working days of 9 hours, starting at 8:00 and finishing at 17:00. This includes that mechanics should be able to return to their home (depot) within the 9 hours, however overtime is paid, which makes it more expensive when mechanics work longer than 9 hours. The number of customers that can be helped within 9 hours depends on the driving time from the depot to the customer, the duration of service at the customer location, and driving time from the customer back to the depot. If possible, additional customers can be served which adds the drive time from customer to customer and the time at a given customer location.

2.5.2 Depot

In this problem, the depot is the same as the mechanic's house address. Therefore, mechanics are required to start and end at their specific depot. It is not possible to end up at another, perhaps closer, depot, as this would mean that they end at another mechanic's home address.

2.5.3 Time windows

About 10 percent of the customers have desired time windows. This means that a customer has indicated the period during which they can or cannot be served. This is often between two different times and sometimes before or after a certain time. Both the check valve and the control measure plans are, if applicable, provided with remarks on time windows. As customer service is an important aspect of the service that is provided by C-Mark, time window constraints need to be considered in the solution model. The customers that do not have a time window are given the time window of the normal workday, which implies that the time when served is not important.

2.6 Costs

The costs are important as the main goal of the research is to find out if the combination of tasks with route optimization decreases costs. It is therefore necessary to understand which aspects these costs are based on. The costs consist of, fuel costs, salary costs, lease costs and possible overtime cost.

The costs per kilometre is 0.077658 euros. The more distance a mechanic travels, the higher these total driving costs are. It is therefore important to reduce the distance as much as possible to save time and costs.

Subsequently, cost of a mechanic plays a major role in the results during the comparison on the separated tasks and combined tasks total costs, as these costs are 145,23 euro per day for control measure mechanics and 181.54 euro per day when the tasks are combined (or only check valve mechanics). This can be reduced by deploying mechanics more efficient, and possibly hiring new mechanics in certain regions to reduce total route distances.

Lastly the overtime cost should be considered. The overtime cost is the salary cost per minute times 1.5, this means that the overtime for control measure mechanics is 0.389 euros and for check valve mechanics 0.454 euros.

2.7 Conclusion

The context analyses give a deeper understanding of the problem. The current manual route planning practice is explained, while the plan is based on logic factors and experience, it cannot guarantee optimality. Furthermore, the data is analysed and cleaned, to ensure that the data can be used as input for the solution model. The characteristics of the problem conclude that the fleet is homogenous, and the number of mechanics differ per day. Based on the characteristic's assumption concerning working hours, depots and time windows are made. Also, assumption to calculate the costs and service time are evaluated, so that they can be used in the solution model. Finally, the how the costs are made up is evaluated, where fuel costs and salary costs are important to consider.

Furthermore, the knowledge questions of the context analysis have been answered:

How are the routes currently planned at C-Mark Eurofins?

The current route plans are made manually, by considering factors like mechanics residential city, previous control data and working days. Additionally, the planner uses his experience to try and optimize the routes taken by the employees as good as possible, however due to manual planning the optimization cannot be guaranteed.

What are important factors that influence the route planning?

The important factors that influence the route planning are:

- Mechanics residence: the mechanics residence is an important factor that influences the route planning. By choosing a mechanic that lives close to certain customers, the total cost is decreased as the driving distance is minimized.
- Distance between customers: the distance between customers must also be minimized for the same reason as described above namely, to reduce the driving distance. By reducing this driving distance, the number of customers served per day can increase.
- Working hours: it is also important that the mechanics have neither too much nor too little work. It is therefore important that the working hours are considered when planning routes.
- Time windows: time windows exist because companies are only or not available at certain times. These are important for customer satisfaction.

The important factors are therefore mechanics residence, distance between customers, working hours and time windows. All these factors influence each other and influence the route planning.

What are the most relevant key performance indicators (KPIs) when planning?

The most relevant KPIs when planning is, to assess the effectiveness and efficiency of the process:

- Cost: the goal of the research is cost reduction. Therefore, this is the most important KPI to assess the effectiveness of the tool. Cost reduction is based on two other important rout optimization, namely route efficiency and resource utilization.
- Route efficiency: route efficiency is based on total distance and total driving time. If the routes are efficient the costs are most likely lower.
- Resource utilization: refers to the effectiveness of the deployment of mechanics and has an impact on the costs, therefore it is an important KPI to consider for this problem.

What is the quality and structure of the data used in the route planning process?

There are two different data file, which are evaluated separately:

- Check valve data: the check valve's data quality was good as there was no missing data and the data was clear. The data had to be cleaned by deleting values that were not interesting for the current problem. After deleting these values the important values as customer location, service time per customer and time windows were left.
- Control measure data: the control measure data needed more work, as this data contained data that was no longer applicable. In addition, the service time per customer was not directly described, but must be calculated by the number of taps that needed to be flushed times the time of one control measure. After cleaning the data looks the same as the check valve data.

3 Theoretical framework

The research question shows that the underlying problem is the problem of route optimisation, based on the assumption of combining multiple tasks over a group of people. Many route optimisation problems have been solved before. This knowledge is used to gain a better understanding of the problem.

In the theoretical framework below, the related theory from the literature concerning the core problem is discussed. In Section 3.1 the traveling salesman problem and vehicle routing problem are explained. Section 3.2 determines which methods for route optimization there are. In Section 3.3 the best heuristic for the problem is chosen and lastly Section 3.4 the chosen method is studied and discussed in more detail. In this section the following knowledge questions are answered:

What route optimization methods are possible for the problem stated by C-Mark Eurofins?

- *What type of optimization problem is stated by C-Mark Eurofins?*
- *What are available methods for route optimization?*
- *Which method is most suitable for the problem stated by C-Mark Eurofins?*

3.1 Traveling Salesman Problem and Vehicle Routing Problem

When looking at route optimisation, The Traveling Salesman Problem (TSP) is the most classical route optimization problem described and researched in operations research. However, the TSP approach often acts as a subproblem, within a more complex optimalisation issue. This more complex issue is also known as the Vehicle Routing Problem (VRP), it is important to first understand how a TSP works, before describing what a VRP is.

3.1.1 Traveling Salesman Problem

In the literature Hoffman (2001) describes the TSP as follows: if a traveling salesman wants to visit exactly *m* cities once, where the cost of traveling from city *i* to city j is c_{ij} and then return to the home city, what is the least costly route the traveling salesman can take? This implies that TSP tries to find the route where every city is visited before he returns to its starting point, with the aim to reduce costs, time, or distance. Figure 2 shows what a TSP route looks like.

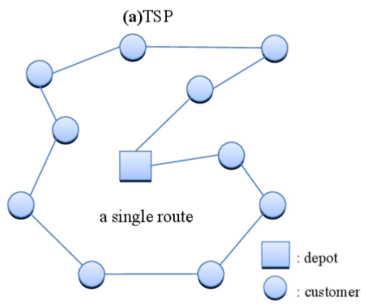


Figure 3 - Example of a TSP solution (Liu W. L., 2014)

The research question posed by C-Mark Eurofins can be seen as a lot of different TSPs, in which every individual mechanic should visit some customers before returning to their house (starting point). Currently there is a list of customers each mechanic must visit, which can be optimised, using TSP. However, during this research it is interesting to see how the overall routes can be optimized, as they influence each other. To do this a VRP must be resolved.

3.1.2 Vehicle Routing Problem

The first VRP was described by Dantzig and Ramser (1959), about “truck dispatching”. After that, a lot of progress on vehicle routing problems has been made. Effective truck routing in the current global economy can potentially leads to enormous optimisations, also in for example the airline travel industry.

The difference between a TSP and a VRP is that a VRP generalizes the TSP. This means that VRP the TSP extends. To do this, the VRP adds complexity, by extending the “relatively simple” Traveling Salesman Problem, by making use of multiple vehicles and by adding additional factors, such as capacity, time windows, and multi-depots (Li, 2019). Figure 3 shows a simple VRP example.

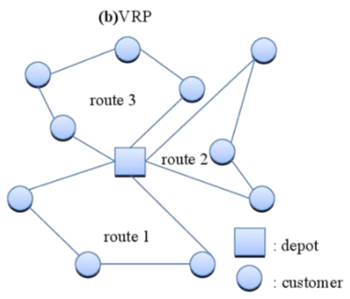


Figure 4 - Example of a VRP solution (Liu W. L., 2014)

The current Vehicle Routing Problems, and their related schedules, are very different from the first proposed VRPs, as they try to solve ever more complex real-life problems. Because of this, various VRP approaches, such as Capacitated VRP (CVRP), Multi-Depot VRP (MDVRP), VRP with Time Windows (VRPTW) and many more are defined and described (Braekers, 2016).

After some analysis, it turns out that C-Mark's problem is most closely related to the Multi-Depot Vehicle Routing Problem with Time Windows (MDVRPTW). The reason is, that in the case of the MDVRP, there are multiple warehouses, which function as a starting point for the routes. This resembles the situation of C-Mark, only in this case, instead of having multiple depots, each mechanic leaves from a different (home) address, which can be seen as depots, to serve customers. Next to this do some of the customers have time windows, which indicates that it is a MDVRPTW.

3.1.3 Multi-Depot Vehicle Routing Problem with Time Windows

The classical MDVRP generalizes the VRP by using more than one depot. Every depot manages its own fleet of vehicles, and with these vehicle all customers should be served. The MDVRP designs a route for each depot, to ensure that all customers are visited with minimum total costs or distance. Hereby the precondition is that every vehicle ends at the same depot as where it started, and depots do not share any vehicle (Aras, 2011).

Just like with VRP, there are many different variants within the specific type of MDVRPs. MDVRP can be capacitated which often indicates that a heterogeneous fleet is used. In some MDVRP’s, for example, the vehicles can have a limited capacity. A heterogeneous fleet of vehicles means that the vehicles each have different characteristics, like capacity / loading space or fuel use (Molina, 2020), thus adding another dimension of complexity to the model. The type of MDVRP that is related to the problem is the multi-depot vehicle routing problem with time windows (MDVRPTW). Time windows is an additional constraint in which a service or delivery have to be delivered.

At C-Mark, the mechanics provide a service and therefore are not restricted to capacity. The mechanics use the same kind cars, which means that fleet is homogeneous, meaning al vehicles have the same characteristics. Customers do in 10 percent of the cases have time windows which need to be considered in the solution model. Lastly the number of mechanics is fixed, however hiring new could be helpful optimizing the route.

Due to no restrictions on capacity, a homogeneous fleet and time windows, the problem stated by C-Mark is based on a MDVRPTW. Where the routes taken by the mechanics start at their home address (depot), providing the services to a set of customers, on control measures and/or check valve control, while time windows are considered and after a working day returning to their home address (depot).

3.2 Route optimization methods

With the description of the VRP type, different route optimization methods can be used. In research there are many different route optimization methods. Liu (2023) made a figure to classify different vehicle routing heuristics, based on constructive heuristics, improvement heuristics and metaheuristics. A heuristic is a method to solve complex real-life problems. Lui (2023) explains and evaluates these types of heuristics. Therefore, in this section different methods are evaluated to eventually help decide what type of heuristic is best for the problem stated by C-Mark Eurofins.

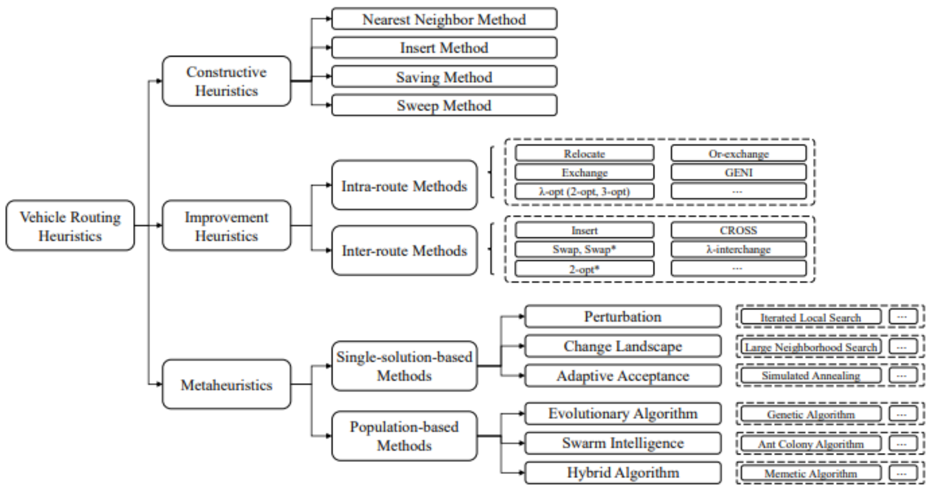


Figure 5 - A classification of vehicle routing heuristics (Liu F. L., 2023)

3.2.1 Constructive heuristics

Constructive heuristics are algorithms to make routing solutions, which means that constructive heuristics use a set of rules to compute a solution. The constructive heuristics generate solutions; however, these solutions are often not the optimal solution. This is because they are efficient in searching for a solution, however constructive heuristics do not search for even better solutions (Liu F. L., 2023). Since constructive heuristics are not good at finding a near-optimal solution, only the nearest neighbour method is explained as this is one of the most common used heuristic for creating an initial solution. Below the nearest neighbour method is described as it is used to generate the initial solution for this research.

Nearest Neighbour

With the nearest neighbour method, as the name suggests, the nearest neighbour (also known as the node or customer), is added to the existing route. This method is the easiest but does often not provide an optimal solution. Constraints indicate when the algorithm needs to stop adding the nearest neighbour to the route, for example if capacity is reached, and when to return to the starting location (Liu F. L., 2023).

3.2.2 Improvement Heuristics

With improvement heuristics, an existing route is improved by iteratively searching for solutions in the neighbourhood. This is different as the heuristic starts with an initial solution instead of finding a solution from scratch. The neighbourhood is the set of nearby solutions that can be found by making small changes. In this way, the methodology quickly determines a local optimum, being the best solution within a specific neighbourhood. Not surprisingly, this is also the biggest disadvantage of this approach. The local optimum, as found by the methodology, depends on the starting point of the analysis (the existing route).

Within improvement heuristics there are intra-route methods and inter-route operators. Intra-route operators focus on optimizing within one route by relocating and/or exchanging positions of customers, whereas inter-route methods focus on optimization across multiple routes.

As this research is based on a MDVRP it is important that both intra-route methods and inter-route methods are included. Inter-route methods result in big changes between different routes while the intra-route methods make minor adjustments in individual routes, to ensure that every route in itself in efficient. Next, various inter-route and intra-route methods for optimizing routes are described.

Insert

The insert selects one location and inserts it into an existing route. This means that a customer within route of mechanic *a* is placed in the route of mechanic *b.*

Swap

The swap, swaps two locations within two different routes. Here a customer within the route of mechanic *a* is swapped with a customer from mechanic *b'*s route.

2-opt*

With 2-opt*, two locations within a route are selected and linked to the opposite route. Here the two mechanics switch routes at a certain point.

CROSS and λ-interchange are further generalizations of the insert, swap and 2-opt*. The CROSS changes two nodes based on the insert, swap and/or 2-opt*. CROSS often has a limited number of λ, which indicates the maximum number of customers that are used for an insert, swap or 2-opt*. λ-interchange is than again a further generalization of CROSS, as it can change more nodes of two routes (Liu F. L., 2023).

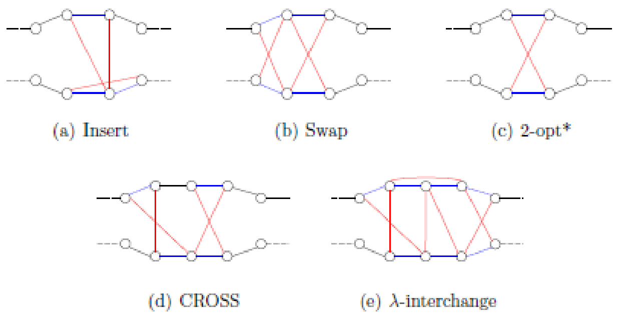


Figure 6 - Illustration of different inter-route improvement heuristics (Liu F. L., 2023)

3.2.3 Metaheuristics

Dökeroğlu (2019) describes metaheuristics as a higher level of heuristics, which is proposed for the solution of a wide range of optimization problems, as they can obtain a near optimal solution for large size problems in relatively small amounts of time. They can solve larger problems which make them important in the vehicle routing research. There are two ways to classify metaheuristics, namely single-solution-based and population-based methods.

Single-solution-based, also known as neighbourhood-based or local search-based, is based on one starting solution, whereas population-based methods are based on multiple starting solutions (Liu F. L., 2023). As the complexity of population-based methods is too high for the time spent on this research, only single-solution-based methods are evaluated in this research. Possible meta heuristics for the MDVRPTW are, Simulated Annealing (SA), Tabu Search (TS), Iterated Local Search (ILS) and Large Neighbourhood Search (LNS).

Simulated Annealing

The first kind of metaheuristic is called Simulated Annealing. SA is commonly used due to the ease of implementation (Robini, 2012). The idea with SA is that a new solution is created in addition to the starting solution. The new solution is accepted with a certain probability, even if it is worse than the previous solution. This is repeated until certain conditions are met (Liu F. L., 2023).

Tabu Search

Tabu search is a heuristic often used in vehicle routing problems. Tabu search is based on how problems are solved by humans, namely by trial-and-error and learning from previous experiences.

The heuristic tabu search explores the solution space, by checking different solutions while keeping track of moves that result in higher costs. By keeping track of these moves, the heuristic knows which moves to avoid. To optimize in this way, the heuristic ensures that it does not end up with a local optimum but explores multiple solutions (Escobar, 2014).

Iterated Local Search

ILS differs from SA and TS, because if it reaches a local optimum, this solution is disturbed to get out of this local optimum. After disturbing this optimum, the heuristics finds a new local optimum. This continues until certain conditions are met (Liu F. L., 2023).

Large Neighbourhood Search

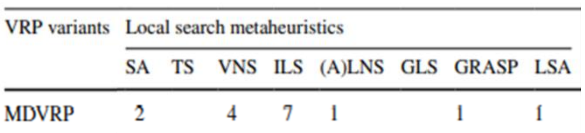
The idea of LNS is that when a large neighbourhood is investigated, it contains a good local optimum. To find this local optimum, the current solution is destroyed and recreated into a new solution. These destructions and re-creations are done by considering the relationships between deleted clients, which is usually measured by distance and other similarities (Liu F. L., 2023).

3.3 Selection of heuristic

Konstantakopoulos (2020) researches the correlation between the VRP problem and the method to solve this VRP problem. In this paper different literature studies, from the years 2010 to 2020, regarding the different VRP problems and their solution approaches are assessed. With this a table is made which shows what methods are used most to solve certain VRP problems.

In the years 2010 through 2020, 29 publications of the MDVRP variant have been published. Due to the complexity of the problem stated by C-Mark, exact methods and heuristics are excluded as possibilities to solve the problem. On the other hand, are population search metaheuristics as described before excluded due to the high complexity of them. The same hold for hybrid methods, which refer to using more than one optimization technique.

As this research is based on a MDVRP where the customers in some cases time windows have. To choose the best method for the MDVRP the correlation between MDVRP and the algorithms is shown in Figure 7.



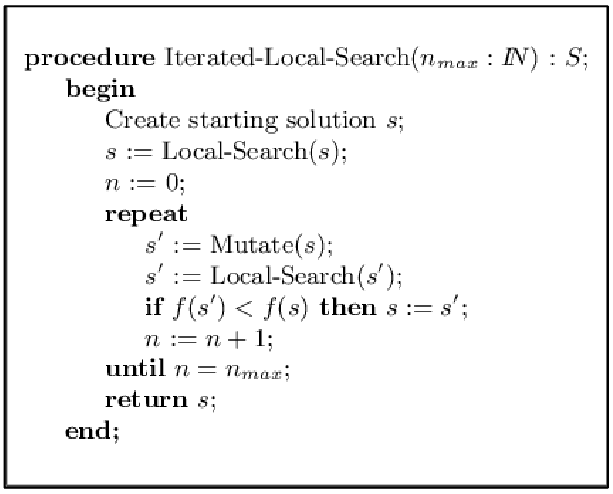
VRP variants	Local search metaheuristics							
	SA	TS	VNS	ILS	(A)LNS	GLS	GRASP	LSA
MDVRP	2		4	7	1		1	1

Figure 7 - Correlation between VRP variants and algorithms (Konstantakopoulos, 2020)

Figure 7 shows that iterated local search is used most to solve MDVRP, therefore the ILS is used to solve this MDVRPTW.

3.4 Iterated local search

As mentioned above iterated local search (ILS) is a metaheuristic. This is because it searches the solution space without getting immediately trapped in a local optimum. The name iterated local search suggests that the metaheuristic iteratively searches from local optimum to a new local optimum. Iterative means repeating certain steps. When using the ILS algorithm, every iteration a new local optimum is found, by a set of local search operators. The pseudo code of the ILS is formulated as seen in Figure 8.



```
procedure Iterated-Local-Search($n_{max}$ : $I\!N$) : $S$;
  begin
    Create starting solution $s$;
    $s$ := Local-Search($s$);
    $n$ := 0;
    repeat
      $s'$ := Mutate($s$);
      $s'$ := Local-Search($s'$);
      if $f(s') < f(s)$ then $s := s'$;
      $n := n + 1$;
    until $n = n_{max}$;
    return $s$;
  end;
```

Figure 8 - Iterated Local Search pseudo code (Merz, 2003)

The Iterated Local Search starts with a certain starting solution s. With this starting solution a local search is performed to find a local optimum s, by applying certain search operators. This solution is subsequently perturbed, which is used to escape from the local optima, which creates s'. Next these a local search is performed to find the local optimum of s'. If this new solution is better than the previous solution the new solution is remembered and set as previous solution. This continues until certain conditions are met, for example the number of iterations (Brandão, 2020).

3.5 Conclusion

The theoretical framework gives a better understanding of methods that can be used for route optimization. These route optimization methods are used for different kind of problems, which makes it important to choose the right route optimization method for the problem stated by C-Mark.

In the theoretical framework the following knowledge questions were answered and are conclude below:

What type of optimization problem is stated by C-Mark Eurofins?

The type of optimization problem stated by C-Mark is a Multi-Depot Vehicle Routing Problem with Time Windows. This is a problem that generalizes the known VRP. The Multi-Depots are known as the residence address of the mechanics and about ten percent of customers have time windows, which indicates that these time windows need to be met. Furthermore, there are no capacity constraints, and the fleet is homogeneous as every car has the same characteristics.

What are available methods for route optimization?

There are a lot of different route optimization methods, namely Constructive heuristics, Improvement Heuristics and Metaheuristics. The constructive heuristics provide a route, but this route does not necessarily be optimal, next to this the improvement heuristic are able to quickly find a local optimum. Lastly the metaheuristic is able to search the solution space which results in optimal results. This are all available optimization methods for VRP.

What route optimization methods are possible for the problem stated by C-Mark Eurofins?

As the problem of C-Mark is quite big not all optimization methods are possible in finding an optimal solution. The constructive and improvement heuristic are not used due to their limitation in finding an overall optimal solution. Therefore, metaheuristics are possible methods for the problem of C-Mark. The population-based methods are to complex and are therefore excluded, this gives that only single-solution-based methods are possible for this problem. The metaheuristics that are single-solution-based and possible for the problem are simulated annealing, tabu search, iterated local search and large neighbourhood search.

Which method is most suitable for the problem stated by C-Mark Eurofins?

The method that is most suitable for the problem is iterated local search, as this is the most used metaheuristic for a MDVRP. The ILS metaheuristic iterative searches for local optima. When a local optima is found it perturbed this local optima to find a new local optimum. Every iteration consists of three steps, the local search, which solution to perturbed, and the perturbation. This is done until certain stop criteria are met, and the best local optimum is the results of the algorithm. Therefore, the ILS method is used during the rest of this research.

4 Solution model

In this section the solution model is evaluated and explained, including the mathematical model and the algorithms used to get results. The first Section 4.1 gives the model description. Section 4.2 describes the mathematical model, with its parameters, decision variables, objective function, and constraints. Section 4.3 illustrates the problem by giving an example and visualisation of the problem. In the next Section 4.4, the assumption used in the tool are described. Section 4.5 explains the algorithms. Lastly, in Section 4.6 the following knowledge questions are answered:

How should the solution approach look like?

- *What are the key components of a mathematical model for route optimization problems?*
- *What kind of input data is needed?*
- *What algorithm should be used?*
- *What considerations regarding usability should be made?*

4.1 Model description

The problem stated by C-Mark Eurofins can be described as a Multi-Depot Vehicle Routing Problem with Time Windows (MDVRPTW) and can be defined as follows. Let $V = N \cup M$ be the set of nodes, where N is the set of customers and M is the set of mechanics (vehicle, depots). The depots refer to the residence of a mechanics as the mechanics start and finish their working days at their homes. Each customer i in N has a specific service time s_i and the customers have a time window in which the service should be finished. The start of a time window is indicated with a_i the end of a time window is indicated with b_i. The mechanics have a maximum working time T of 9 hours, for which they get paid a constant salary C. The driving time d_{ij} and the travel cost c_{ij}, between every nodes i and j is defined, where $d_{ij} = d_{ji}$, $c_{ij} = c_{ji}$ for $i, j \in V$. The MDVRPTW aims to minimize the total cost of routes taken by the mechanics, by making decisions and adhering the problem specific constraints.

4.1.1 Requirements

The MDVRPTW should adhere to several requirements. These requirements are stated below and form the basis for the constraints:

- Each customer should be visited once
- A mechanic must start and end at their depot location
- A working day for a mechanic 9 hours
- The time windows have to be adhered to

4.2 Mathematical formulation

The problem of C-Mark is modelled as a Mixed Integer Linear Programming. This section describes the mathematical model and its parameters, decision variables, objective functions, and constraints.

4.2.1 Parameters

The parameters stated below are divided in sets, time, and cost to give a better understanding of them. Parameters are the input of the solution model, which impact the decision-making of the solution model.

Sets:

- Total set of customers: $N = \{1,2,\dots,n\}$, where n is the total number of customers.
- Total set of mechanics: $M = \{1,2,\dots,m\}$, where m is the total number of mechanics (depots).
- Set of nodes: $V = N \cup M$

Time:

- Driving time between nodes i and j, where $d_{ij} = d_{ji}$ for $i, j \in V$
- s_i: Service time at customer i where $i \in N$
- a_i: Start of time window of customer i where $i \in N$
- b_i: End of time window of customer i where $i \in N$
- T: Maximum working time for each mechanic (9 hours)

Cost:

- c_{ij}: Travel cost between nodes i and j with mechanic k, where $c_{ij} = c_{ji}$ for $i, j \in V$
- C: Costs of mechanics working day

4.2.2 Decision variables

The decision variables represent the unknown information or choice in an optimization problem. For this problem, the following three decision variables were defined:

- $x_{ij}^k = \begin{cases} 1, if\ mechanic\ k\ travels\ from\ node\ i\ to\ node\ j \\ 0,\ otherwise \end{cases}$
- t_i^k: Arrival time at customer i for mechanic k
- y: number of used mechanics

The first decision variable is a binary variable, which indicates that the value can only be 1 or 0. The second decision variable is a continuous variable, which indicates that the value can be any real value, without breaching any constraints. The third decision variable has an integer value which depends on the number of mechanics working for a certain day.

4.2.3 Objective function

The goal of the objective function is to minimize the total costs of a certain day. This is done by multiplying the travel cost times the decision variable.

$$min \sum_{i,j \in V, j \neq i} \sum_{k \in M} c_{ij} * x_{ij}^k + C * y$$

4.2.4 Constraints

The constraints ensure that the solution is feasible within the given boundaries of the problem.

The first constraint ensures that every customer is visited exactly once.

$$\sum_{k \in M} \sum_{i \in V} x_{ij}^k = 1, \qquad \forall j \in N, \tag{1}$$

Constraint (2) and (3) ensure that every engineer starts and finishes at their own depot (home address). Where 0 refers to the home address of a mechanic k.

$$\sum_{i \in N} x_{0i}^k = 1, \qquad \forall k \in M, \tag{2}$$

$$\sum_{i \in N} x_{i0}^k = 1, \qquad \forall k \in M, \tag{3}$$

The following constraint ensures that when a mechanic visits a node it, that the mechanic also leaves this node, also known as flow conservation, as the number of mechanics visiting a node should be equal to the number of mechanics leaving the node. This is necessary to ensure that no mechanics vanish after visiting a customer.

$$\sum_{i \in V, j \neq i} x_{ij}^{k} = \sum_{i \in V, j \neq i} x_{ji}^{k}, \qquad \forall j \in N, \forall k \in M, \tag{4}$$

The next constraint is also known as the subtour elimination constraint. The subtour elimination constraint ensures that the routes are connected, and feasible and not have smaller loops which do not include all the assigned customers. The subtour elimination constraint works the following, if $x_{ij}^{k} = 1$, which indicates an engineer travels from i to j, then the start of service at customer j must happen at least after the service time plus travel time from i to j. If $x_{ij}^{k} = 0$, then T ensures that the equation still holds.

$$t_j^k + T * \left(1 - x_{ij}^k\right) \geq t_i^k + s_i + d_{ij}, \qquad \forall i \in V, \forall j \in N, \forall k \in M, \tag{5}$$

The next constraint ensures that engineers start at time 0.

$$t_0^k = 0, \qquad \forall k \in M, \tag{6}$$

Constraint (7) states that the start of a service at customer i must happen within its time window.

$$a_i \leq t_i^k \leq b_i, \qquad \forall i \in N, \tag{7}$$

The last constraint ensures that a mechanic does not work more than the maximum working time.

$$\sum_{i \in V} \sum_{j \in V, j \neq i} \left(t_i^k + s_i + d_{ij}\right) * x_{ij}^k \leq T, \qquad \forall k \in M, \tag{8}$$

4.3 Problem Illustration

Figure 9 presents an illustrative solution to the MDVRPTW described in the Section 4.2. This problem takes the above constraints into account. The illustration has 9 nodes, for which 7 customers labelled A to G and two depots labelled 1 and 2. A mechanic starts their working day from a depot at time 0. The service time is stated on top of every customers location, and one time window is added to customer F. Lastly the travel time between nodes is given in minutes next to the arrows.

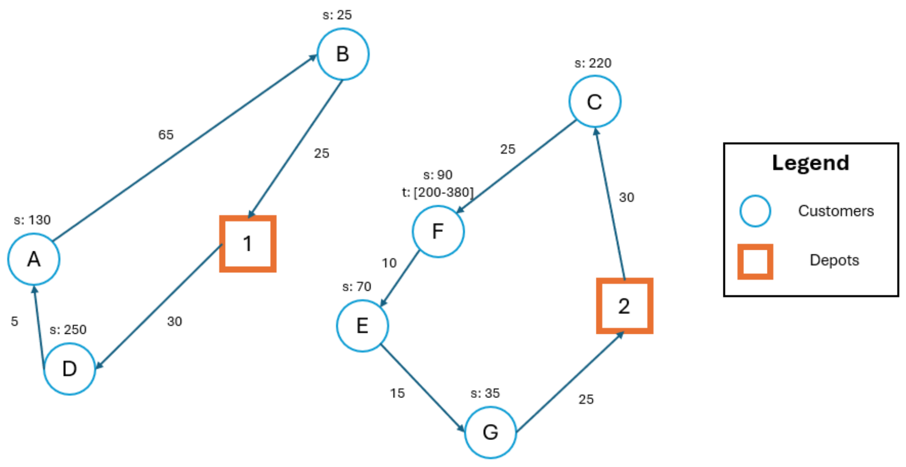


Figure 9 - Illustration of a MDVRPTW solution for C-Mark

In the solution shown above, mechanic 1 departs from the depot at 8:00 AM, start time is 0. Despite the fact that node E and F are the closest from depot 1, the time window need to be considered and therefore mechanic 1 takes a different path. For this path, the travel times and the service times need to be considered, to ensure not exceeding the maximum working day hours. As the total time of the first route does not exceed the 540 minutes, namely 530 minutes, this route is feasible.

Customer F has a time window which means that between 200 minutes after the start of day and before 380 minutes the customer should be served. In the illustration the start of service time at customer F is 275 (30 + 220 + 25) and the end of service is 365 (275 + 90), as this is between the time window the time window constraint is met. The total duration of the second route is 520 which is smaller than 540 and therefore is also a feasible route. The cost of this route is then the travel cost plus salary cost times the number of mechanics (depots).

4.4 Assumptions

As some variables depend on different unknown factors, assumptions need to be made to ensure that the tool works. Therefore, some assumptions are true during this research. These assumptions are based on factors and research, including averages and technical car specifications, and they are evaluated and described in this section.

4.4.1 Fuel consumption

The fuel consumption of a car depends on several things, such as how fast does a mechanic drive or how fast does the mechanic accelerate. Since this is different for everyone, it is important to assume the average fuel used per kilometre. This assumption is based on the technical specifications of the Opel Combo, which states that the average fuel consumption in urban areas is 4.5 l/100km and 4.0 l/100km in rural areas. Opel states that average fuel consumption of an Opel Combo is 4.2 l/100km (Opel, 2019). This means every kilometre 0.042 litre diesel is used.

4.4.2 Diesel price

The price for diesel depends on the gas station, therefore an assumption on the diesel price needs to be made. On the ANWB site an average diesel price, which was updated on May 10, 2024, of 1.849 euros is given (ANWB, 2024). Therefore, it is assumed that C-Mark pays 1.849 euro per litre diesel.

Now that the diesel price 1.849 euro, and fuel consumption 0.042 l/km is known, the price per kilometre can be calculated:

$$1.849 * 0.042 = 0.077658 \text{ €}/km$$

4.4.3 Driving time

The driving time for a mechanic is important for the working hours constraint, as the driving time directly influences the number of customers that can be helped during a day. It is assumed that the average speed of a mechanic is 75 km/h. With this assumption the time between customers can be calculated by dividing the distances between customers by 75km/h.

4.4.4 Mechanic cost per day

The salary costs are based on a gross income of 2800 euro per month plus the lease cost, which are 700 euro, and the average working hours per month. On average does a mechanic work for 154,24 hours per month, with the vacation days taken into account. So, the average income of a mechanic per day can be calculated:

$$\frac{3500}{154{,}24} * 8 \approx 181.54 \text{ €}/day$$

4.4.5 Control measure mechanic cost per day

To calculate how the current planning practise compares to the combined planning practise it is important to know how much a control measure mechanic costs. To calculate the same formula as above is used, the control measure mechanic gets 2400 euro salary per month and the car costs 400 euro per month.

$$\frac{2800}{154{,}24} * 8 \approx 145{,}23\ €/day$$

4.5 Solution Methodology

The metaheuristic iterated local search should be able to find an optimal and feasible solution in an appropriate running time, for current plans and plans in the future. As this problem is too big to mathematically give the best results it is important to write an algorithm that is able to find this feasible and near optimal solution within an acceptable time. To do this the used algorithms for this problem are explained in the next section.

4.5.1 Iterated local search

Algorithm 1 shows the extended ILS code used in the tool, where first a local search is performed on the initial solution to get a local optimum. This is then saved as the best route (best_route) and best cost (best_cost).

Then, a for loop loops over the chosen number of iterations. The best route is then perturbated, using the perturbation definition. Using the local search definition, the perturbated solution (perturbed_route) gives a local optimum. If the new cost (new_cost) is smaller than the best cost, then the new route (new_route) becomes the best route and the new cost becomes the best cost. This continues until the number of iterations is achieved. The best route and best cost are returned.

Algorithm 1 Iterated Local Search

```
Input: Route, iterations
Output: best_route, best_cost
1:  Start:
2:  local_search on initial solution gives current_route and current_cost
3:  best_route is current_route
4:  best_cost is current_cost
5:  for a defined number of iterations:
6:      perturbation(best_route) gives perturbed_route
7:      local_search(perturbed_route) gives new_route and new_cost
8:      if new_cost smaller then best_cost
9:          best_route is new_route
10:         best_cost is new_cost
11:     end if
12: end for
13: return best_route, best_cost
```

Algorithm 1 - Iterated Local Search

4.5.2 Input

In Section 2 the input data is described. This is for the depots the location, longitude (lat) and latitude (lon), and for the customers the location, longitude and latitude, and the service time per customer, and possible time windows. With these locations a distance (distance_matrix) and time matrix (time_matrix) are made, which indicates a matrix with the distance and travel time from every location to ever other location. With the distance matrix, fuel consumption and the fuel price a cost matrix (cost_matrix) is made.

4.5.3 Distance and time matrix

It is important to create a distance and time matrix, to calculate the total distance and time covered within a route. C-Mark has a lot of different customers, often more than 50 customers a day, where more than 10 mechanics have to work. With more than 60 locations, this translates to more than 3600 distances between the locations. While there are methods to obtain precise distances and travel times using distance matrix APIs from applications like OpenStreetMap and Google Maps, these services have request limitations, typically capped at 3500 requests or fewer. This limitation poses a challenge for the problem addressed by C-Mark, as the number of distances between locations exceeds these API request limits. That is why another method is used to calculate the distances between locations.

One of the most used ways to calculate the distances between coordinates (longitude and latitude), is the Haversine formula. This formula takes the curvature of the Earth into considerations to determine good distances between locations. The Haversine formula is as follows:

$$D = 6378\{\arccos\,(\sin LAT_A * sin\ LAT_B + \cos\ LAT_A * cos\ LAT_B\ * \cos\ |LONG_A - LONG_B|)\}$$

With this formula the distance between location A and location B is calculated using the radius (6378 km) of the earth. However, this method calculates the distance between two locations is a straight line, but in reality, routes are not completely straight form location to location. Therefore, it is important to compensate for this problem. To do this the API of OpenRouteService is used to give the distances matrix for 59 locations (3481 requests). For the same 59 locations a distance matrix was created using the Haversine method. The ratio between the API matrix and the haversine matrix per distance was calculated and the average was 1.329, which is close to the calculated ratio by (Ballou, 2002), which found ratio of 1.32 for routes in Germany.

Therefore the 1.329 ratio found by comparing the API matrix with the Haversine matrix is used to get a distance matrix for larger instances. To compare and analyse if this number is good for estimating the distances between locations, a distance matrix for 27 locations using the API and using the Haversine method times 1.329 were created and evaluated.

When comparing these matrices, the MAE (Mean Absolute Error) and MAPE (Mean Absolute Percentage Error) are calculated. The MAE measures the average errors of predictions. Where the MAPE gives the average percentage of the error.

	Value
MEA	10.35
MAPE	8.56%

Table 1 - MEA and MAPE for distance matrix

The MEA as a percentage of the mean real values is 7.92%, given that both values are smaller than 10 percent this is a good way of calculating the distance between two locations. Therefore, the value 1.329 is used to calculate the distance between two locations.

The average of the real distances divided by the real times was 1.245 kilometre per minute, which means that the average speed is 74.7 kilometres per hour. That is why 74.7 kilometres per hour is used to calculate the time matrix.

4.5.4 Initialization using Nearest Neighbour method

The Nearest Neighbour method is used to create an initial solution. This is done by adding the nearest neighbour to a given route. This initial solution is not yet feasible yet, as it does not take the maximum working hours into account. Some of the routes take way more minutes than is acceptable during a working day. Next to this are the time windows not yet considered, this results in high cost. However, due to the high penalty cost the time window constraint is met during the ILS phase.

It is very hard to create an initial solution within the time constraint. The easiest way to deal with this is, to correct it during the optimization process, by punishing the overtime with a rate of 0.2 euro per minute. This way, the overtime weighs so heavily, that an alternative route leads to a cheaper outcome.

Algorithm 2 shows the way how this Nearest Neighbour method creates the initial route. The input values are the number of depots (NrDepots), which indicate the number of mechanics, the number of customers (NrCustomers), time matrix and the data from the customers (customers_data).

First a list is created for every depot and the first depot is added to this list. A copy of the time matrix (CopyTimeMatrix) is created so that when a customer is served, all distances to this customer are set to infinity. This ensures that the customer is never accidentally used more than once. By using a copy of the time matrix, the initial time matrix remains unchanged, preventing potential issues in other steps. Then the time of each route is set to zero.

The "for" loop ensures that every customer is added to a route. To do this, the algorithm first evaluates which route takes the least amount of time, this route is selected. Then the nearest next customer is Selected and added to the route. The total time of the route is then updated, by first removing the trip from the depot to the last customer, as this trip does not take place. This is replaced by adding the time from the last location to the new location, plus the service time of the new location, plus the time from the new location back to the depot . The distance from each location to the just added customer is set to infinite, to make sure it is not selected for a route again. This is done until every customer is placed into a route.

Lastly the initial depot is added to the last location so that the route starts and ends at the same depot and the algorithm returns the route of the initial solution.

Algorithm 2 Nearest Neighbour

```
Input: NrDepots, NrCustomers, time_matrix, customers_data
Output: initial_route
1:  Start:
2:  Initialize empty list for each route
3:  Add starting depot to each route
4:  Create a copy of the time matrix
5:  Set time of each route to zero
6:  for every customer in the range of customers:
7:      Find route that takes the least amount of time
8:      Select last location of vehicle
9:      Find nearest unserved customer
10:     Add nearest unserved customer to route
11:     Update total time of a route
12:     Set distance to customer in copy time matrix to infinite
13: end for
14: Add the depot to the route
```

```
15: return initial_route
```

Algorithm 2 - Nearest Neighbour

4.5.5 Local search

The local search algorithm is used to find a local optimum for the given starting solutions. It evaluates every possible move for each customer to every potential location and determining which move results in the lowest cost. Once the optimal move is identified, the solution is updated and the process repeats, evaluating all possible moves again. This continues until no better moves are found, indicating that the local optimum is found. To do this the algorithm 3 is made.

The algorithm first sets the Route as current route and cost of Route as current cost. Then improved is set to true and a while loop is introduced. This while loop sets improved to false and runs until no better solution is found. Two for loop then loop over all the possible moves of the given current route and the best solution is returned. This new solution then enters the loop where again all possible moves are assessed. When no better cost can be found the best route is set as current route and best cost is set as current route and is used for the perturbation step.

```
Algorithm 3 def local_search()
Input: Route, cost_matrix, time_matrix, customers_data
1:  Start:
2:  current_route is Route
3:  current_cost is cost of Route
4:  improved is True
5:  while improved:
6:      improved is False
7:      best_route is current_route
8:      best_cost is current_cost
9:      Initialize list for all customers in routes
10:     for each customer i in customers:
11:         for each customer j in customers:
12:             Make copy of route
13:             Remove customer i from first route
14:             Insert customer j into the second route
15:             Calculate cost of new route
16:             if new_cost is smaller than best_cost:
17:                 best_cost is new_cost
18:                 best_route is new_route
19:                 improved is true
20:             end if
21:         end for
22:     end for
23:     if improved:
24:         current_route is best_route
25:         current_cost is best_cost
26:     end if
27: end while
28: return current_route, current_cost
```

Algorithm 3 - def local_search()

As looping over all possible moves might take a lot of time the random local search optimization strategy is introduced and is evaluated in the next section.

4.5.6 Reduced Local Search

The reduced local search definition is introduced as another way of finding a good result with a lower running time, for a given starting route. To do this two different methods are introduced, namely swaps, which can be found in Algorithm 4 and indicates swapping two customers between routes or in the same route, and moves, which can be found in Algorithm 6 and indicates moving a customer to a different route or a different position in its own route.

Swapping means that randomly customers are swapped between two routes or swapping customers randomly in the same route, to determine whether the outcome becomes better. For a given number of samples (number_samples_swaps) solutions are created by performing a swap. Half of the cases inter route swaps are done and half of the cases intra route swaps are done. Swapping between different routes is done by randomly selecting two different routes, checking if they have at least one customer within the route, randomly selecting two different customers within the routes and swapping these customers. The new given solution is stored in the list "neighbours". At the end there are a given number of solutions in this list. Intra route swapping is done by selecting a random route, checking whether there are at least two customers in that route and swapping random customers in the given route.

For each of these solutions in the "neighbours" list, the cost of that solution is calculated. If the cost (neighbour_cost) is lower than the best neighbour cost (best_neighbour_cost), then the solution with the best cost is the new best neighbour cost. If this best neighbour cost lower than the cost of the starting route (current_route), the best neighbour cost is the new current route and the number of no improvements (no_improvement_swaps) is reset to zero. If the best neighbour cost is not lower than starting route one is added to number of no improvements.

When no better solution is found 5 times in a row, the while loop ends, and a specific number of moves are done.

Algorithm 4 def reduced_local_search() (swaps)

```
Input: Route, cost_matrix, time_matrix, customers_data, NrDepots
1:  Start:
2:  current_route is Route
3:  current_cost is cost of Route
4:  while no_improvement_swaps smaller than total_no_improvement_swaps:
5:      Initialize neighbours list
6:      Set num_routes to NrDepots
7:      for each sample in number_samples_swaps:
8:          Make copy of current_route as neighbour
9:          if random smaller than 0.5
10:             Select two random routes
11:             if length of route1 and route2 are greater than two:
12:                 Randomly select a customer i in route1
13:                 Randomly select a customer j in route2
14:                 Swap the selected customers between route1 and route2
15:             end if
16:         else:
17:             Select random route
18:             if length of route is greater than three:
```

```
19:                Randomly select a customer i in route
20:                Randomly select a customer j in route
21:                Swap the selected customers in route
22:             end if
23:          end if
24:          Add the modified neighbour to the neighbours list
25:       end for
26:       for each neighbour in neighbours:
27:          Calculate cost of the neighbour
28:          if neighbour_cost is lower than best_neighbour_cost:
29:             best_neighbour is neighbour
30:             best_neighbour_cost is neighbour_cost
31:          end if
32:       end for
33:       if best_neighbour_cost is smaller than current_cost:
34:          current_route is best_neighbour
35:          current_cost is best_neighbour_cost
36:          Reset no_improvement_swaps to zero
37:       else:
38:          Add one to no_improvement_swaps
39:       end if
40:    end while
```

Algorithm 4 - def reduced_local_search() (swaps)

The second method to optimize the routes is the use of moves. The algorithm for the swaps is almost the same as for the moves only the content from line 9 to 23 are different. In Algorithm 5 the move part is shown. Again, in half of the cases inter route moving is done and half of the cases intra route moving is done. For the move part, first two random routes are selected. It is important that the route that moves a customer has at least one customer within the route. Then a random customer from the first route is selected and a random position in second route is selected. After this the customer from first route is removed and inserted into second route. During the intra route swapping a random route is selected, if the length is greater than three a random customer is selected and placed in another position of the route.

After swapping and moving customers to find a local optimum, the current_route and the current_cost are returned and used for the next step, the perturbation.

Algorithm 5 def reduced_local_search() (moves)

```
1:  Start:
2:  if random smaller than 0.5
3:     Select two random routes
4:     if length of route1 and route2 are greater than two:
5:        Randomly select a customer i in route1
6:        Randomly select a position j in route2
7:        Remove the customer form route1
8:        Insert the customer into route2
9:     end if
10: else:
11:    Select random route
12:    if length of route is greater than three:
13:       Randomly select a customer i in route
```

```
14:         Randomly select a position j in route
15:         Remove the customer form route
16:         Insert the customer into position j
17:     end if
18: end if
19: return current_route, current_cost
```

Algorithm 5 - def reduced_local_search() (moves)

4.5.7 Perturbation

The perturbation algorithm is necessary to get out of a local optimum. This is necessary to find new local optima. The definition ensures that the local optima is disrupted so that a new local optimum can be found with the local search definition. The algorithm used to perturbate the route can be found in Algorithm 6.

The route is perturbated by randomly swapping a certain number of customers and afterwards moving a random number of customers. This way of swapping and moving is the same for the local_search definition. The goal is to get a random solution, so a new local optimum can be found.

```
Algorithm 6 def perturbation()
Input: Route, NrDepots
Output: perturbed_route
1:  Start:
2:  Create a copy of the Route
3:  Set num_changes to the number of changes to perform
4:  for the num_changes:
5:      select two different routes
6:      if length of route1 and route2 are greater than two:
7:          Randomly select a customer i in route1
8:          Randomly select a customer j in route2
9:          Swap the selected customers between route1 and route2
10:     end if
11: end for
12: for the num_changes:
13:     select two different routes
14:     if length of route1 and route2 are greater than two:
15:         Randomly select a customer i in route1
16:         Randomly select a position j in route2
17:         Remove the customer form route1
18:         Insert the customer into route2
19:     end if
20: end for
21: return perturbed_route
```

Algorithm 6 - def perturbation()

4.6 Conclusion

Within the solution model the mathematical model is made and the algorithms for the iterated local search and the reduced local search are explained. In this section the following knowledge questions were answered and are conclude below:

What should the solution approach look like?

Due to the fact that this problem is too large to solve mathematically it is important to create a code which is able to find an optimal route with its optimal cost. The mathematical model therefore gives a better understanding of the problem, which is important when making the Python code.

What are the key components of a mathematical model for route optimization problems?

The mathematical model consists of the parameters, decision variables, the objective function, and constraints. The decision variable tells us if a mechanic k travels from node i to node j, next to this does it tell what the arrival time of mechanic k at customer i is. The objective function calculates the cost of the route, and the constraints ensure that the solution is feasible within the given boundaries of the problem.

What kind of input data is needed?

The input data that is need are location, longitude, and latitude, for both the depots and customers and additional input date for the customers is the service time per customer and possible time windows. With the longitude and latitude this locations distance, time and cost matrix are made, from every location to ever other location, using the Haversine method. All these inputs are needed to for calculating the output.

What algorithm should be used?

Within the tool 7 different algorithms are used and explained, the first is the ILS algorithm, the second to create the initial solution using the nearest neighbour method. The third algorithm is a cost function to calculate the cost for the routes. Then there are the algorithms for the local search and the reduced local search, using swaps and moves, for finding a local optimum. Lastly, you have the algorithm that takes care of going out of the local optimum.

What considerations regarding usability should be made?

For usability, it is important that the code is well explained so that even people with no understanding of programming understand how to optimization process, using iterated local search, works. In addition, it must be easy to use, which is achieved because any data set with the right input data can be used to find an optimal result.

5 Evaluation

In this chapter the two developed solution approaches, iterated local search (ILS) and reduced local search (RLS), are evaluated. In Section 5.1 the experimental design is described. After the experimental design, the data instances used for the parameter tuning and experiments is described in Section 5.2. Section 5.3 analyses the results from the parameter tuning, with these results the best method is analysed and selected in Section 5.4. Lastly, the scenarios for C-Mark are described and evaluated in Section 5.5. The following knowledge question is answered in this section:

How to determine the best parameters and the best optimization method to generate the results?

5.1 Experimental design

To assess and evaluate the tool several experiments are conducted.

- The first experiments are related to the parameter settings for both the ILS and the RLS. For both the ILS and the RLS, the number of iterations and the perturbation strength should be determined. Additionally, for the RLS, the number of moves and the number of swaps need to be determined.
- The second experiment is the numerical experiments for the different methods. This is done to assess the best optimization method for different scenarios.
- Lastly, the experiments for the final results are conducted. Based on these results conclusions and recommendations are made.

All these experiments are run on a Windows computer with 16GB RAM, Intel i7 core and 2.60 GHz core. Where the ILS and the RLS algorithms are written in Python 3.12, using the PyCharm IDE.

5.2 Data instances

To validate and do experiments with the code it is important that the data used is consistent throughout all the experiments.

5.2.1 Parameter tuning instances

Three different data instances, DS (small), DM (medium), and DL (large), are used for the parameter tuning, of the ILS and the RLS. The data instances for parameter tuning are created by random sampling, which means where a random selection of customers is chosen from the data sets provided by C-Mark. The number of customers and number of depots are chosen to ensure that every customer can be serviced, without a mechanic having to work a too long working day. The number of customers and number of depots per instance can be seen in Table 2.

This method for selecting data is used to create an environment, where both the ILS and RLS can be tested on performance, by introducing different parameter values. The data instances are used to determine the perturbation strength, the number of iterations, the number of moves and number of swaps. The best parameters for the ILS and the RLS are chosen based on the results, like total cost and running time, of the tools.

ID	Number of Customers	Number of Depots
DS	20	5
DM	40	8
DL	60	11

Table 2 - Data instances for parameter tuning

5.2.2 Method selection instances

Before generating the final results, the right method for the problem should be chosen based of experiment with six data instances. These data instances are generated by again selecting random customers from the data sets provided by C-Mark. The data instances can be seen in Table 3.

ID	Number of Customers	Number of Depots
D1	10	3
D2	20	5
D3	30	6
D4	40	8
D5	50	10
D6	60	11

Table 3 - Data instances for method selection

It is important to use different numbers of customers and numbers of depots, to determine the robustness of the tool and to give a clear overview on what tool performs better. The best performing tool, ILS or RLS, is used to generate the last results.

5.2.3 Real scenarios instances

For generating the results where the recommendations and conclusion are based on, real-life data is used from three different days. The real-life data gives insight if combining the tasks or keeping them separated is cost effective. The real-life data are 3 different days when the check valves and control measures for a given number of customers and a given number of mechanics have been performed. The data that is used for optimizing the routes are Thursday 31-01-2024, Wednesday 07-02-2024 and Monday 11-03-2024 and can be seen in Table 4.

ID	Number of Customers	Number of Depots
D1cv	22	7
D1cm	39	4
D1c	61	11
D2cv	16	6
D2cm	38	3
D2c	54	9
D3cv	15	6
D3cm	31	4
D3c	46	10

Table 4 - Data instances for results

The D1cv is the data of the check valves, D1cm is the data of the control measures and D1c is the data when the check valves and the control measures data are combined. It is important to have several scenarios to ensure that the results match multiple scenarios, as only having only one results does not guarantee that the conclusions are right.

5.3 Parameter Tuning

In this section the parameters for the ILS and the RLS are chosen and evaluated, based on the total cost and the run time. Parameter tuning is important as it improves performance of the tool and gives more insights into different aspects of the tool. First the perturbation strength, and the number of iterations is evaluated. Thereafter the number of swaps and the number of moves for the RLS are chosen.

5.3.1 Parameters ILS

To improve and analyse the performance of the tool, parameter tuning is an important step. Tuning the parameters is done by iteratively adjusting the parameters. This ensures that all the different possible options of different parameters are covered, in order to choose the best parameters. Due to the many different options between the perturbation strength, number of iterations, the data instances and that every option is executed five times to ensure good results, the running times get very high.

When executing the ILS all the different moves that are possible are evaluated and the best move (based on the greatest cost reduction) is adopted as the new solution. An acceptable running time for the code was set at 20 minutes, but this limit was quickly reached due to a high number of iterations and strong perturbations, even with low amounts of data. Because of this it would take way too long to go through all the different options of iterations and perturbations strengths.

Since the RLS does not go through all options but does a certain number of swaps and moves until it has not found a better solution 5 times in a row, therefore the RLS is faster than the ILS. As the RLS is based on the ILS it is decided to do the parameter tuning experiments on the RLS and apply the findings to the ILS. This allows good comparison could be made during the method selection in Section 5.4.

5.3.2 Parameters RLS

For the RLS parameter tuning different experiments are performed. First the perturbations strength and number of iterations are determined, afterwards the number of swaps and the number of moves are determined. The parameters that yield the best improvement and minimum cost are selected, these parameters are used for method selection and for generating the results.

Perturbation strength

The first experiment is to assess the number of iterations combined with the perturbation's strength. There are sixteen different possibilities with iterations being [5, 10, 20, 50] and the perturbation strength [5, 10, 20, 30]. The perturbation strength is the percentage, this percentage is multiplied by the number of customers to give the number of moves and swaps. The number this gives is the number of moves and swaps that are used to perturb the local optimum.

The experiment is run for each of the data instances determined in Section 5.2. As there is randomness involved, in both the local search and the perturbation, each of the experiments is performed 5 times, and the average improvement compared with the initial solution and average run time are calculated to compare the different options. It is important to notice that the initial solution is sometimes very high, which also makes the improvement very high. This is due to the fact that by generating the initial solution the time window constrains are not considered. Therefore, often the punishment cost of 1000 euro is added to the total cost of the initial solution. This is not a problem as it is still possible to compare the different results with each other. The experiment are as follows:

- *Experiment 5,5: In this experiment the number of iterations is 5 and the perturbation strength is 5.*
- *Experiment 5,10: In this experiment the number of iterations is 5 and the perturbation strength is 10.*
- *Experiment 5,20: In this experiment the number of iterations is 5 and the perturbation strength is 20.*
- *Experiment 5,20: In this experiment the number of iterations is 5 and the perturbation strength is 30.*

The results of this experiment can be found in Table 5, where the top of each column is shown as number of iterations, perturbation strength. The improvement en run time are the average as every experiment is performed 5 times.

ID	Initial Solution (€)	5,5		5,10		5,20		5,30	
		Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)
DS	1053.94	**5.70**	19.47	5.66	21.96	5.66	24.90	5.60	27.17
DM	2816.43	**44.06**	112.83	43.78	103.06	43.43	128.74	42.21	140.93
DL	4783.37	**53.18**	186.57	53.02	245.16	52.90	304.53	52.17	323.67
Avg	2884.58	**34.31**	106.29	34.15	123.39	34.00	152.72	33.33	163.92

Table 5 - Experiment 5 iterations

From the first experiments it is clear that a perturbation strength of 5 is better than any of the other perturbations strengths, when doing 5 iterations. In all three of the data instances this perturbation strength gives the best improvement compared with the initial solution, which can also be seen in the average improvement.

In the second experiment the number of iterations is 10. The results of this experiment can be found in Table 6. The experiments are formulated as follows:

- *Experiment 10,5: In this experiment the number of iterations is 10 and the perturbation strength is 5.*
- *Experiment 10,20: In this experiment the number of iterations is 10 and the perturbation strength is 10.*
- *Experiment 10,20: In this experiment the number of iterations is 10 and the perturbation strength is 20.*
- *Experiment 10,30: In this experiment the number of iterations is 10 and the perturbation strength is 30.*

ID	Initial Solution (€)	10,5		10,10		10,20		10,30	
		Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)
DS	1053.94	5.55	32.51	**5.79**	42.73	5.50	47.27	5.58	47.27
DM	2816.43	**44.01**	172.63	43.96	210.35	43.81	237.97	43.40	249.95
DL	4783.37	**53.27**	406.45	53.25	463.85	53.06	597.24	52.70	632,79
Avg	2884.58	34.28	203.86	**34.33**	238.98	34.12	294.16	33.89	148.61

Table 6 - Experiment 10 iterations

For the data instance DS, the perturbation strength of 10 has the best improvement. However, for the last two data instance again a perturbation strength of 5 is best. On average with the number of iterations being 10 the perturbation strength of 10 is best.

The third experiment does a total of 20 iterations. The results of the third experiment can be found in Table 7. The experiments are formulated as follows:

- *Experiment 20,5: In this experiment the number of iterations is 20 and the perturbation strength is 5.*
- *Experiment 20,20: In this experiment the number of iterations is 20 and the perturbation strength is 10.*
- *Experiment 20,20: In this experiment the number of iterations is 20 and the perturbation strength is 20.*
- *Experiment 20,30: In this experiment the number of iterations is 20 and the perturbation strength is 30.*

ID	Initial Solution (€)	20,5		20,10		20,20		20,30	
		Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)
DS	1053.94	**5.80**	68.66	5.80	76.79	5.73	90.47	5.76	96.68
DM	2816.43	44.31	319.95	**44.46**	413.67	43.69	452.74	43.95	497.97
DL	4783.37	**53.60**	775.73	53.42	872.07	52.93	1053.68	53.37	1139.40
Avg	2884.58	**34.57**	388.11	34.56	454.18	34.12	532.30	34.36	578.02

Table 7 - Experiment 20 iterations

For this experiment the best improvement is again between a perturbation strength of 5 and 10. The data instance DS has for both a perturbation strength of 5 and 10 an average improvement of 5.80, however the average run time when the perturbation strength of 5 is lower than the perturbation strength of 10. Therefore, the best average improvement for data instance DS is a perturbation strength of 5. In total best average improvement is with a perturbation strength of 5.

The fourth and last experiment for the number of iterations and the perturbation strength, is 50 iterations. The results can be found in Table 8. The following experiments are conducted:

- *Experiment 50,5: In this experiment the number of iterations is 50 and the perturbation strength is 5.*
- *Experiment 50,20: In this experiment the number of iterations is 50 and the perturbation strength is 10.*
- *Experiment 50,20: In this experiment the number of iterations is 50 and the perturbation strength is 20.*
- *Experiment 50,30: In this experiment the number of iterations is 50 and the perturbation strength is 30.*

ID	Initial Solution (€)	50,5		50,10		50,20		50,30	
		Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)	Improvement (%)	Time (s)
DS	1053.94	5.89	155.83	**5.92**	183.29	5.89	217.41	5.86	247.40
DM	2816.43	**44.50**	780.75	44.38	951.39	44.17	1127.244	44.27	1254.17
DL	4783.37	**53.86**	1757.09	53.59	2138.77	53.63	2885.03	53.10	2795.41
Avg	2884.58	**34.75**	897.89	34.63	1091.15	34.56	1409.89	34.41	1432.33

Table 8 - Experiment 50 iterations

Again, the best improvement is for a perturbation strength of 5 and 10 for different data instances. Something else that stands out is the run time for data instances DM and DL, as the run time is very high. Where even the average run time for the large data instance DL is all above the 10 minutes runtime restriction.

When evaluating all results for all different experiments it is clear that a perturbation strength of 5 is the best. In the experiments a perturbation strength of 5, has the most improvement for most or even all data instances. Therefore, the perturbation strength is 5.

Number of iterations

Table 9 is created to better evaluate the optimal number of iterations, where the minimum, average and maximum cost, and average running time for the numbers of iterations with a perturbation strength of 5 is displayed.

ID	Initial Solution (€)	5,5				10,5				20,5				50,5			
		Cost (€)			Time (s)	Cost (€)			Time (s)	Cost (€)			Time (s)	Cost (€)			Time (s)
		Min	Avg	Max		Min	Avg	Max		Min	Avg	Max		Min	Avg	Max	
DS	1053.94	992.03	993.91	996.01	19.47	991.55	995.46	999.39	32.15	**991.53**	992.84	997.13	68.66	991.57	991.88	992.49	155.83
DM	2816.43	1564.86	1575.42	1591.74	112.83	**1560.07**	1576.95	1598.27	172.63	1562.33	1568.41	1577.70	319.95	1560.61	1563.06	1568.48	780.75
DL	4783.37	2214.57	2239.50	2264.43	186.57	2219.15	2235.20	2258.53	406.45	2210.93	2219.44	2235.10	775.73	**2199.69**	2206.98	2229.35	1757.09
Avg	2884.58	1590.49	1602.94	1617.39	106.29	1590.26	1602.54	1618.73	203.74	1588.26	1593.56	1603.31	388.11	1583.96	1587.31	1596.77	897.89

Table 9 - Experiment number of iterations

The bold numbers show which combination has found the best minimum costs per data instance. It can be seen that 5 iterations in no case found the lowest cost. Therefore, the number of iterations is not 5.

The 50 iterations give the best average results, however the running time for the larges data instance is too large and therefore is not suitable to use for the further experiments. The 20 iterations generally have a better average compared to the 10 iteration, next to this the minimum found cost is better in two out of three cases for 20 iterations. Lastly, the run time of the 20 iterations is also acceptable. Hence the number of iterations is 20 for choosing the number of swaps and number of moves, the method selection and obtaining the results.

Number of swaps vs number of moves

With the number of iterations being 20 and perturbations strength 5 an experiment can be performed on the number of swaps and the number of moves. There are four different possibilities when the set of number of swaps is [10, 20], and for the number of moves is [50, 100]. Again, like before are all experiment run 5 time due to the randomness involved in both the local search and the perturbation phase. The results can be found in Table 10. With this the experiments are as follows:

- *Experiment 10,50: In this experiment the number of iterations is 10 and the perturbation strength is 50.*
- *Experiment 10,100: In this experiment the number of iterations is 10 and the perturbation strength is 100.*
- *Experiment 20,50: In this experiment the number of iterations is 20 and the perturbation strength is 50.*
- *Experiment 20,100: In this experiment the number of iterations is 20 and the perturbation strength is 100.*

ID	Initial Solution (€)	10,50				10,100				20,50				20,100			
		Cost (€)			Time (s)	Cost (€)			Time (s)	Cost (€)			Time (s)	Cost (€)			Time (s)
		Min	Avg	Max		Min	Avg	Max		Min	Avg	Max		Min	Avg	Max	
DS	1053.94	991.55	994.44	999.90	41.51	991.60	991.88	992.77	62.01	991.54	993.36	996.03	43.72	991.35	992.85	997.13	69.87
DM	2816.43	1570.27	1581.30	1600.60	177.26	1560.34	1576.19	1605.44	328.93	1568.33	1577.46	1583.24	197.34	1561.46	1564.19	1566.80	331.64
DL	4783.37	2211.53	2231.54	2245.51	388.11	2211.56	2223.92	2242.62	708.66	2218.42	2230.37	2264.13	419.08	2211.83	2226.10	2246.29	768.43
Avg	2884.58	1591.12	1602.43	1615.34	202.30	1587.83	1597.33	1613.61	366.54	1592.76	1600.40	1614.47	220.05	1588.21	**1594.38**	1603.41	389.98

Table 10 - Experiment number of swaps and number of moves

When looking at the results from the previous experiment, it can be seen that on average the 20 swaps are better than the 10 swaps. As there is randomness involved it is important that on average the results are good and consistence, and that the results are not dependent on chance. With the swaps being 20 the number of moves need to be determined. When looking at experiment 20,50 and 20,100 not only the average of 100 moves is better but also the minimum and maximum cost is better than the 50 moves. Therefore, the number of swaps is 20 and the number of moves is 100.

With all the results from the previous experiment the number of iterations is set to 20 as it gives really good results in a reasonable run time. The perturbation strength is set to 5, due to the fact that it has the best improvement compared with a perturbation strength of 10, 20 and 30. Lastly the number of swaps is set to 20 and the number of moves is set to 100.

This means that for the ILS the number of iterations is also 20 and the perturbation strength is 5. As the ILS tries every possible move, the number of swaps and number of moves are not applicable for the ILS.

5.4 Method selection

With the given parameters the next experiment can be conducted. This experiment has the goal of choosing the best method to generate the results needed to give recommendations to the company. For this, the ILS and RLS is run 5 times for every data instance to generate the minimum, average and maximum cost and run time. The results of the two experiments can be found in Table 11.

ID	Initial	Iterated Local Search						Reduced Local Search					
	Solution	Cost (€)			Time (s)			Cost (€)			Time (s)		
	(€)	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
D1	707.18	592.75	592.75	592.75	2.06	2.13	2.23	590.5813	591.66	591.93	16.22	16.65	16.96
D2	2223.28	977.84	978.11	978.42	73.08	76.17	82.09	973.45	973.81	975.1457	79.41	85.85	90.10
D3	3130.82	1371.65	1431.31	1529.37	300.11	327.69	352.80	1368.50	1372.15	1379.52	180.22	190.74	218.77
D4	3742.41	1557.06	1564.36	1577.90	1190.92	1301.64	1368.94	1554.51	1558.90	1566.83	427.98	453.34	496.29
D5	2305.02	1951.82	1955.53	1960.38	2855.07	2994.978	3154.12	1951.55	1957.39	1965.94	503.76	556.45	614.33
D6	4713.47	2165.06	2173.418	2184.58	7203.36	7600.058	8044.12	2175.00	2183.43	2198.52	901.27	942.86	987.97
Avg	2803.70	1436.03	1449.25	1470.57	1937.43	2050.44	2167.38	1435.60	1439.56	1446.31	351.48	374.32	404.07

Table 11 - Experiment method selection

As mentioned earlier, the initial solutions can be very high due to the time windows. By chance, a data instance with more mechanics and depots might have a lower initial cost. This can be observed in data instance D5, which has a lower cost than data instances D3 and D4.

With the results given in Table 10 the best method for the MDVRPTW for C-Mark can be chosen. What is immediately noticeable, is the fact that data is missing for the iterated local search experiment from data instance D4. This has to do with the fact that the runtime became much higher than 20 minutes. As a result, it would take too long to perform this experiment, and with the high running time the tool would not be a good method for C-Mark.

When we compare the data for the first three instances, it can be seen that the reduced local search gives better results than the ILS. In addition, the run time for the ILS is so high that it cannot be used to obtain results. For these reasons, the RLS is used for the final experiment, obtaining the results.

5.5 Results

The last experiments are conducted to get the results. Before this can be done first the three different data instances are explained carefully in Section 5.5.1. In Section 5.5.2. the results are shown and discussed.

5.5.1 Scenarios

The experiment consists of finding the minimum cost, total distance, total duration and run time of three scenarios, namely check valves, control measures and combined scenario, and compare them.

Check valves

The first scenario is the to calculate the minimum cost for the check valve mechanics of a certain day. A check valve mechanics gets paid 181.54 euro per day and the overtime cost is 0.454 per minute, with these two values and the cost matrix, calculated by taking the price of diesel into account, the minimum cost of this scenario can be calculated.

Control measures

The second scenario involves the control measures. For the scenario, the control measures mechanics get paid 145.23 per day, and the overtime cost is 0.389. With the cost of the check valves and the control measures scenario, these cost can be added together, which can be seen as the simulated current practice. These added together cost can be compared with the combined scenario to draw conclusions.

Combined

In the last scenario, both tasks are combined, meaning that mechanics can both control check valves and control measures. Again, the minimum cost is calculated. Since the mechanics in this scenario have to be able to do both tasks, they get paid the same amount as a check valve mechanic, namely 181.54 euros. This means that previous control measure mechanics get paid 36.31 euro extra, which is an important aspect for the results.

With the scenarios explained above the tool is able to calculate the total cost, total distance, total duration and run time in the following section.

5.5.2 Result experiments

With all the parameters evaluated, the chosen method selected, and the scenarios explained in more detail, the experiments for the final results can be executed. For this is important to change the salary and overtime cost depending on the data instance. For all Dcv and Dc data instances these are set to 181.54 and 0.454 and for all Dcm data instances these are set to 145.23 and 0.389. The results of the last experiments can be seen in Table 12 and are described as follows:

- *Experiment day 1: In this experiment the total cost, total distance, total duration and run time of day 1 is calculated.*
- *Experiment day 2: In this experiment the total cost, total distance, total duration and run time of day 2 is calculated.*
- *Experiment day 3: In this experiment the total cost, total distance, total duration and run time of day 3 is calculated.*

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)	Run Time (s)
D1cv	1406.84	1301.70	3501.36	95.77
D1cm	623.64	550.08	1286.81	328.41
D1c	2134.63	1757.80	4712.99	877.55
D2cv	1176.30	751.42	2419.14	29.89
D2cm	461.85	336.92	910.04	220.48
D2c	1741.23	1012.92	3268.84	852.72
D3cv	1137.39	620.03	2128.04	27.48
D3cm	612.85	411.18	1554.44	225.59
D3c	1884.34	887.76	3567.73	540.24

Table 13 - Results of real-life data

The values should be compared to each other. To do this it is important to add the check valve scenario total costs, total distance and total duration to the control measure scenario total costs, total distance, and total durations. Then subtract the total cost, total distance, and total duration of combined scenario to see the impact of combining the task and give recommendations based on them. The outcome of this is presented in Tabel 13, 14 and 15.

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)
D1cv + D1cm	2030.48	1851.78	4788.17
D1c	2134.63	1757.8	4712.99
D1cv + D1cm - D1c	-104.15	93.98	75.18

Table 12 - Differences combined vs separated day 1

What is interesting to see is that the total distance and total duration of the combined scenario is indeed less than when check valves and control measures are separated, which is good following the

KPIs efficiency and recourse utilization. The routes are more efficient as the total distance decreases and the recourses (mechanics) utilized more efficient as the total duration is less. However, the most important KPI, cost, is worse for combined versus separated. This is due to the fact that four mechanics need to be paid more as they change from a control measure mechanic, salary 145.25 euro and 0.389 euro overtime cost per minute, to a mechanic that has to do both tasks, salary 181.54 euro and 0.454 euro overtime cost per minute. Paying these four mechanics 36.31 euro extra adds an additional cost of 145.24 euro (4 * 36.31). A small amount, namely 41.09 euro, is made up for by combining the tasks and reducing the total distance, which indicates a decrease in fuel cost, and possible previous overtime costs. However, the total cost is still higher when the tasks are combined. To visualise the scenarios for day one Figure 10, 11, 12 are made. These visualizations are helpful for the company to see what routes are driven by the different mechanics.

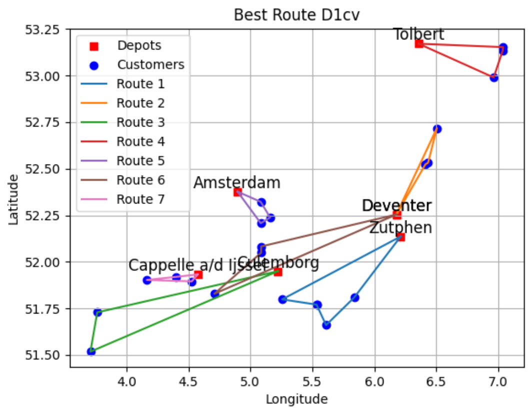


Figure 12 - Best route D1 check valves

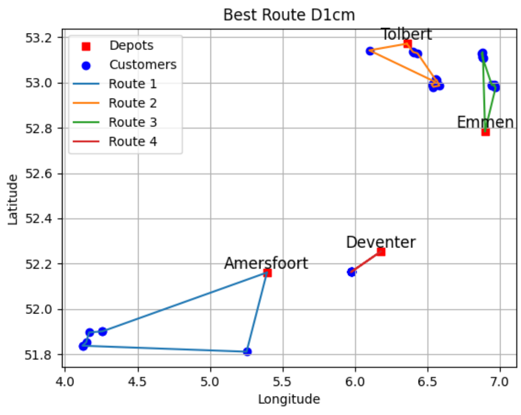


Figure 11 - Best route D1 control measures

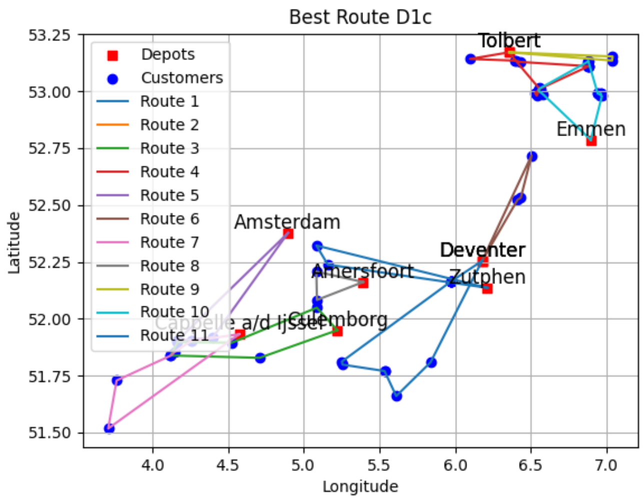


Figure 10 - Best route D1 combined

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)
D2cv + D2cm	1638.15	1088.34	3329.18
D2c	1741.23	1012.92	3268.84
D2cv + D2cm - D2c	-103.08	75.42	60.34

Table 15 - Differences combined vs separated day 2

ID	Total Cost (€)	Total Distance (km)	Total Duration (min)
D3cv + D3cm	1750.24	1031.21	3682.48
D3c	1884.34	887.76	3567.73
D3cv + D3cm - D3c	-134.1	143.45	114.75

Table 14 - Differences combined vs separated day 3

The same is true for the other two experiments, the total distance and total durations is reduced by combing the tasks, however the cost is more due to paying the mechanics more salary. For the second experiment the extra cost of paying three mechanics more salary is 108.93 euro, and the third experiment the extra cost of paying four mechanics is 145.24 euro. Which means that again only a small amount is made up for by combing the tasks, but the cost for changing the mechanics is too high. The routes of the last two experiment can be found in the appendix. So, the conclusion is that it is not cost efficient to combine tasks.

However, the analysis above is built on the premise that salary cost is fixed this is true if the amount of mechanics were fixed. However, efficiency gains due to a decrease in total duration can result in less mechanics or more customers:

The total duration / throughput time of the three optimised plans, including travel and the control measures related activities is 11549.56 units for Dc and 11799.83 units for Dcv + Dcm. This means an efficiency gain of approximately 2.12%. This leaves us two options 1) reduce the number of mechanics 2) increase the number of customers.

- An efficiency gain of 2.12% translates to a need for 97.88% of the current mechanics capacity, in order to perform the same amount of work. The impact on costs can be determined as follows. On average, 10 mechanics work during these 3 days. This means you need a total of 9.788 mechanics to do the same work, indicating that one mechanic does not need to work a full day. This would save 2.12% * 181.54 euro per day * 10 mechanics, being about 38.49 euro per day. The combined savings from the distance and efficiency is still not enough to positively impact the overall cost.
- A third component of "cost" would be to look at the potential for additional income, which could be generated in the time gained through the efficiency gain. So not to focus on reducing the number of mechanics, but on increasing the number of customers you can serve. With the reduction in total work time, you can serve about 2.12% more customers. For the company C-Mark, this means you can serve about 0.39 additional customers per day. Assuming the income outweighs the associated costs of the mechanics, the gap of the additional costs should be further reduced. I did not dive into this analysis.

5.6 Conclusion

In this chapter the evaluation of the tool is discussed, and with the evaluation the final results of the research were produced. In this chapter the following knowledge question is answered:

How to determine the best parameters and the best optimization method to generate the results??

The tool can be evaluated by executing different experiments with different data instances. These data instances are used to evaluate the parameters, method selection and generate results. The parameters are important to assess different aspects of the tool, like robustness. The ILS takes too long to assess the parameters through experiments as it checks every possible move, therefore the found parameters form the RLS are used for the ILS. Due to the randomness in both methods every experiment is run 5 times. The best perturbation strength is 5, and the number of iterations is 20. This gives a good balance between a good result and a reasonable running time. In addition, the number of swaps is 20 and the number of moves is 100, as this generates the best average cost. The ILS has the same number of iterations and perturbations strength as the RLS and does not have the parameters number of swaps of moves as it tries every possible move.

As the ILS has a to high run time the RLS is chosen as the method which is used for producing the final results. The final results show that by paying the mechanics more money the cost of the combined task is higher than the separated tasks, on the other hand is the amount of kilometres and the duration the routes reduced when combining the tasks.

6 Conclusion

In this chapter the results of the problem faced by C-Mark are described. In Section 6.1 conclusion based on the tool are made. Based on the conclusion recommendations are made in Section 6.2. Section 6.3 elaborates on the contribution to practice and theory and the last Section 6.4 gives the limitations and addresses possible future research.

6.1 Conclusions

This research has investigated the route optimization problem faced by C-Mark, finding that the company relies on manual planning and has no dedicated route optimization system. A thorough analysis has been done into the entire problem context, the type of vehicle routing problem and the best meta heuristic for the problem. The analysis showed that both travel costs and salary costs have a significant impact when calculating the total cost of the routes.

The primary objective was to research if and how the combination of check valves and control measures would be helpful in decreasing the costs and to answer the main research questions: How can the control measures and check valve controls be jointly planned to decrease costs at C-Mark (Eurofins)?

A thorough literature study formed the basis for developing the tool. A mathematical model was created for C-Mark's Multi-Depot Vehicle Routing Problem with Time Windows (MDVRPTW) and two different methods, Iterated Local Search (ILS) and Reduced Local Search (RLS), were developed. These methods were created for the specific features of C-Mark's VRP problem, such as the time windows, service times, maximum working hours and the mechanics starting and ending their day at their own home. The data and assumptions about the data were provided by C-Mark.

To evaluate the ILS and RLS, several data instances were created for parameter tuning, method selection, and obtaining the final results. The same parameters used for the RLS were applied to the ILS, such as a perturbation strength of 5 and 20 iterations. This approach was taken because the ILS had a long run time, a direct result of including all possible moves. Additionally, the number of swaps and moves were set to 20 and 100, respectively. However, these last two parameters do not apply to the ILS, as it includes all possible moves. Due to the high run time of the ILS, the RLS was chosen as the best method, and the results were obtained accordingly.

The research revealed that, although combination of tasks results in a reduction in the total distance and duration of the routes, the total costs increase. This is because mechanics who perform both tasks receive a higher salary then control measure mechanics, namely an additional 36.31 euros per day per mechanic. For four mechanics, this equals an additional salary cost of 145.24 euros per day. As a result, the total cost is 104.15 euros higher when the tasks are combined, meaning that the distance savings only amount to 41.09 euros.

With the decrease in duration, 97.88% of the mechanics are needed for the same work, which translates to a potential saving of 38.49 euros in salary costs per day. Instead of having one mechanic work less, you can also argue that you can serve 2.12% more customers. By doing this, C-Mark can help approximately 0.4 customers per day extra. Since I do not know the average revenue per customer, it is up to C-Mark to consider whether this increase makes combining customers cost-efficient.

6.2 Recommendations

Based on the findings of this research, the following recommendation is made regarding the combination of check valve control and control measures: The results show that combining the tasks entails additional costs. Therefore, it is recommended to not combine the check valve controls and control measures and to keep them separated. This is because the combination of tasks does not save costs initially and is even more expensive than when the tasks are separated. This is due to the higher salary costs that must be paid. Additionally, the potential cost savings from employing a mechanic for less hours a day does not save a sufficient amount of money to make the combination of tasks cost-effective. Keeping the tasks separated also prevents the need to search for new mechanics or retrain control measure technicians, as this may entail additional costs that are not yet included in the model. This also saves the extra salary costs that would have to be paid if the tasks were combined.

However, the time saved can be translated into approximately 0.4 customers extra per day. It is up to C-Mark to consider whether the revenue from this additional customer is high enough to make the combination of tasks ultimately cost-efficient.

6.3 Contribution to theory and practice

In the following two sections, the contributions of this research are explained for the contribution to theory and the contribution to practice.

6.3.1 Contribution to theory

In the research field of vehicle routing problems, different solution methods have been introduced in papers. However, most of these solution methods are fit-for-purpose for the specific problem explained per paper. About MDVRPTW, there is limited information for problem C-Mark encounters. Next to this, the use of service times and the limits of working hours, is little reflected in the theory. This implies that my research gives new guidance on how to solve a vehicle routing problem with these specific characteristics, thus the main contribution are the heuristics and assessment of a new application of the problem.

6.3.2 Contribution to practice

This research offers practical insights for C-Mark, on what the influence would be of combining different activities, on the total cost, total distance, and total duration of the routes. The solution design is specially developed for C-Mark and provides insights into how route optimization could help work more cost-efficiently. The various analyses in this research also provide insights that could adjust decision-making by C-Mark in the future. Even with the tasks not being combined is the tool is able to provide a good results which could be used by C-Mark in the future. Finally, the research could provide other companies with insight into combining tasks.

6.4 Limitations and future research

Although this study provides much relevant information for C-Mark Eurofins, there are some limitations that should be acknowledged. An important limitation is that it cannot be said with certainty that the results of the RLS (Reduced Local Search) algorithm are optimal. This is due to the randomness during the local search phase of the algorithm.

The RLS algorithm performs a random swap or move and then chooses the best swap as the next solution. This creates the possibility that the algorithm does not find a better swap or move too early and then returns a sub-optimal solution. This means that there could potentially be better solutions that are not found by the RLS algorithm. Nevertheless, it can be said that the RLS algorithm is able to "approach" the optimal result within a reasonable run time. The near optimal results still provide valuable insights and improvements over current planning methods.

The second limitation is that the service times are assumed, using the number of check valves or taps, by multiplying the number of check valves by 6 minutes and the number of taps by 2 minutes.

However, in reality, not every mechanic works at the same speed and sometimes the distance between check valves or taps is big, which causes a higher service time.

The last limitation is that the distance matrix is calculated using the Haversine method, instead of real-life distances and travel times. This is due to restrictions on getting more than 3500 distances and travel times, between locations, from the OpenStreetMap API. It could be interesting for C-Mark to find out what type of applications or paid subscriptions there are, for getting an API with no maximum on distances and travel times.

Future research could explore whether serving an additional 0.48 customers per day could potentially offset the higher salary costs. This could be possible due to the extra revenue that is earned by serving this additional 0.48 customers. If this is the case the combination of tasks could be beneficial.

Although the literature review indicated that the best metaheuristic for an MDVRPTW was the Iterated Local Search (ILS), future research could also explore other metaheuristics. The chosen metaheuristic for this research turned out to have a too long run time, due to the size of the dataset with customers and depots. Testing alternative metaheuristics can lead to optimised solutions within a reasonable running time.

7 References

ANWB. (2024). *Brandstofprijzen Europa | Bekijk het actuele overzicht | ANWB.* Retrieved from https://www.anwb.nl/vakantie/reisvoorbereiding/brandstofprijzen-europa

Aras, N. A. (2011). *Selective multi-depot vehicle routing problem with pricing. Transportation Research. Part C, Emerging Technologies, 19(5), 866–884.* Retrieved from https://doi.org/10.1016/j.trc.2010.08.003

Ballou, R. R. (2002). *Selected country circuity factors for road travel distance estimation. Transportation Research Part A 36, pp. 843–848.*

Braekers, K. R. (2016). *The vehicle routing problem: State of the art classification and review. Computers & Industrial Engineering, 99, 300–313.* Retrieved from https://doi.org/10.1016/j.cie.2015.12.007

Brandão, J. (2020). *A memory-based iterated local search algorithm for the multi-depot open vehicle routing problem. European Journal Of Operational Research, 284(2), 559–571.* Retrieved from https://doi.org/10.1016/j.ejor.2020.01.008

Dökeroğlu, T. S. (2019). *A survey on new generation metaheuristic algorithms. Computers & Industrial Engineering, 137, 106040.* Retrieved from https://doi.org/10.1016/j.cie.2019.106040

Escobar, J. W. (2014). *A hybrid Granular Tabu Search algorithm for the Multi-Depot Vehicle Routing Problem. Journal of Heuristics, 20(5), 483–509.* Retrieved from https://doi.org/10.1007/s10732-014-9247-0

Hoffman, K. P. (2001). *Traveling salesman problem.* Retrieved from https://seor.vse.gmu.edu/~khoffman/TSP_Hoffman_Padberg_Rinaldi.pdf

Konstantakopoulos, G. D. (2020). *Vehicle routing problem and related algorithms for logistics distribution: a literature review and classification. Operational Research, 22(3), 2033–2062.* Retrieved from https://doi.org/10.1007/s12351

Li, Y. S. (2019). *An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives. Journal of Cleaner Production, 227, 1161–1172.* Retrieved from https://doi.org/10.1016/j.jclepro.2019.03.1

Liu, F. L. (2023). *Heuristics for Vehicle Routing Problem: A Survey and Recent Advances.* Retrieved from https://arxiv.org/pdf/2303.04147#:~:text=They%20build%20a%20solution%20quickly,%2C%20and%204)%20sweep%20method

Liu, W. L. (2014). *Minimizing the Carbon Footprint for the Time-Dependent Heterogeneous-Fleet Vehicle Routing Problem with Alternative Paths. Sustainability, 6(7), 4658–4684.* Retrieved from https://doi.org/10.3390/su6074658

Merz, P. (2003). *An iterated local search approach for minimum Sum-of-Squares clustering.* Retrieved from https://doi.org/10.1007/978-3-540-45231-7_27

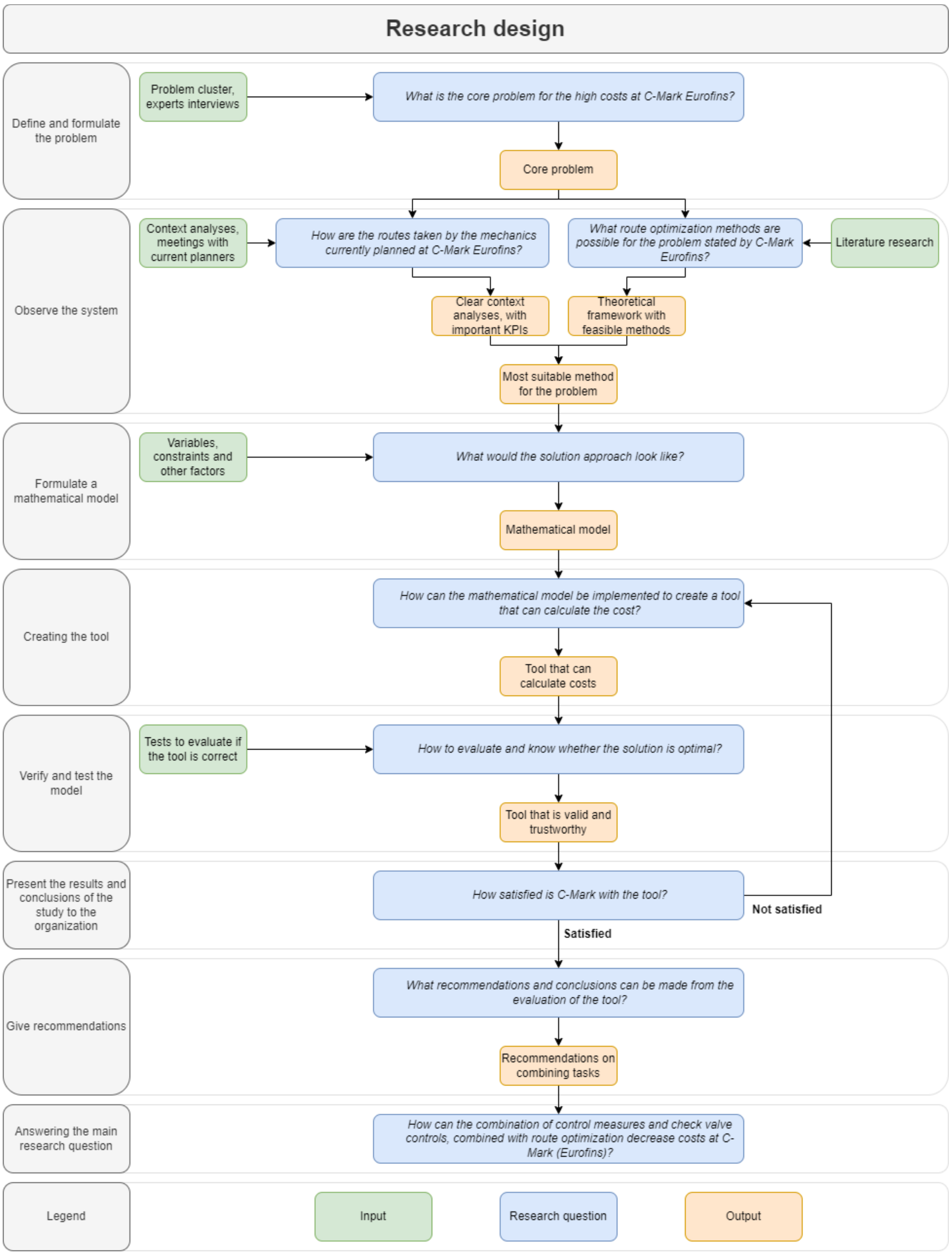
Molina, C. A. (2020). *The heterogeneous vehicle routing problem with time windows and a limited number of resources. Engineering Applications of Artificial Intelligence, 94, 103745.* Retrieved from https://doi.org/10.1016/j.engappai.2020.103745

Opel. (2019). *Opel Combo. In Modeljaar 2020.* Retrieved from https://www.opel.nl/media/showroom/pdfs/nl/combo_specs.pdf

Robini, M. C. (2012). *From simulated annealing to stochastic continuation: a new trend in combinatorial optimization. Journal of Global Optimization, 56(1), 185–215.* doi:https://doi.org/10.1007/s10898-012-9860-0

8 Appendix

8.1 Research design



Research design
Define and formulate the problem
Problem cluster, experts interviews
What is the core problem for the high costs at C-Mark Eurofins?
Core problem
Observe the system
Context analyses, meetings with current planners
How are the routes taken by the mechanics currently planned at C-Mark Eurofins?
What route optimization methods are possible for the problem stated by C-Mark Eurofins?
Literature research
Clear context analyses, with important KPIs
Theoretical framework with feasible methods
Most suitable method for the problem
Formulate a mathematical model
Variables, constraints and other factors
What would the solution approach look like?
Mathematical model
Creating the tool
How can the mathematical model be implemented to create a tool that can calculate the cost?
Tool that can calculate costs
Verify and test the model
Tests to evaluate if the tool is correct
How to evaluate and know whether the solution is optimal?
Tool that is valid and trustworthy
Present the results and conclusions of the study to the organization
How satisfied is C-Mark with the tool?
Not satisfied
Satisfied
Give recommendations
What recommendations and conclusions can be made from the evaluation of the tool?
Recommendations on combining tasks
Answering the main research question
How can the combination of control measures and check valve controls, combined with route optimization decrease costs at C-Mark (Eurofins)?
Legend
Input
Research question
Output

Figure 13 - Research design

8.2 Route figures results

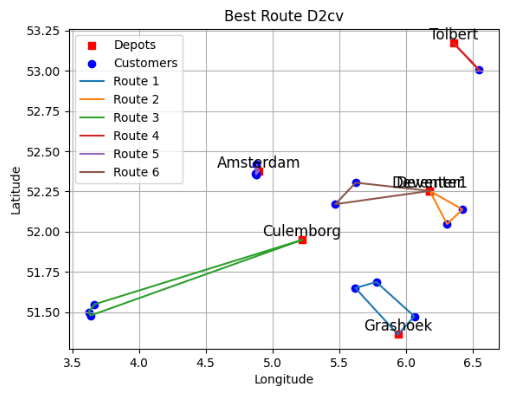


Figure 15 - Beste route D2 check valves

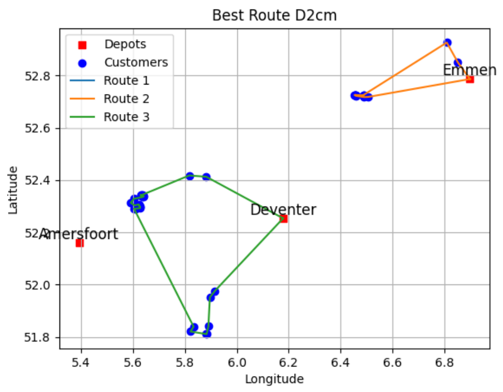


Figure 14 - Beste route D2 control measures

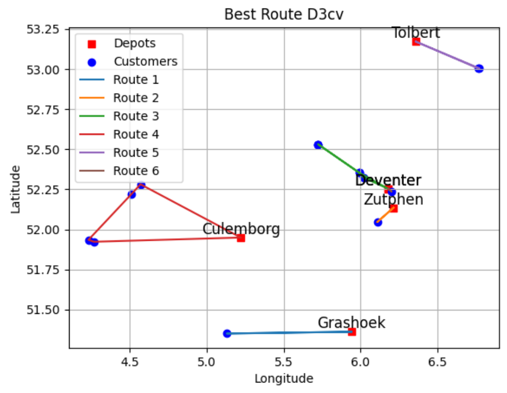


Figure 17 - Beste route D3 check valves

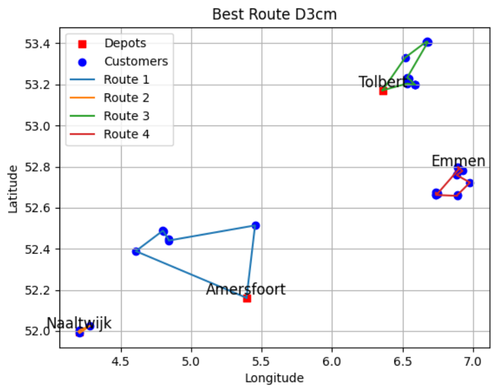


Figure 16 - Beste route D3 control measures

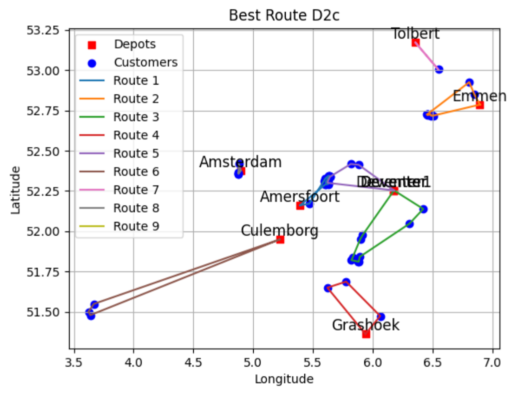


Figure 18 - Beste route D2 combined

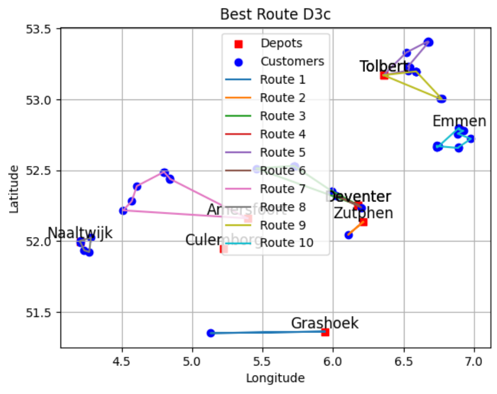


Figure 19 - Beste route D3 combined