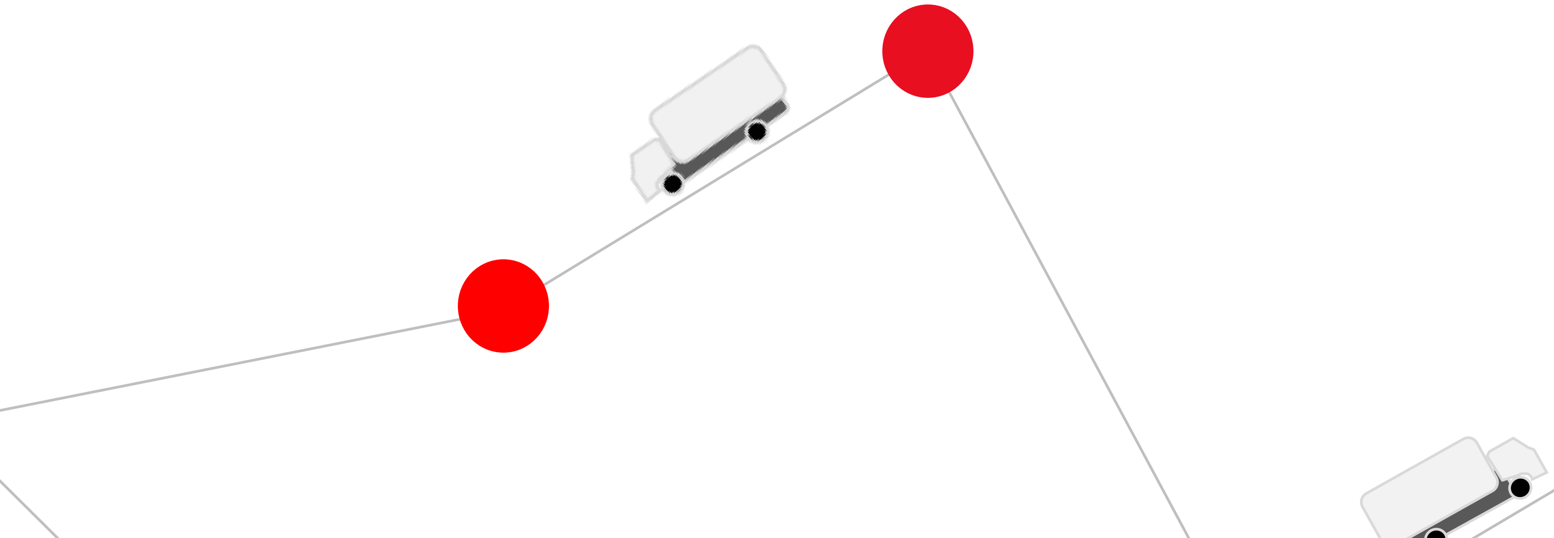
UNIVERSITY OF TWENTE.

FACULTY OF BEHAVIOURAL, MANAGEMENT AND SOCIAL SCIENCES

SHORTENING THE DELIVERY ROUTES OF KOSKAMP B.V.

Thomas de Boer



University of Twente

Dr. J. M. J. Schutten

Dr. E.A. Lalla

September 2024

Bachelor Thesis Industrial Engineering and Management

UNIVERSITY OF TWENTE.

Koskamp

Shortening the delivery routes of Koskamp B.V.

Author

Thomas de Boer
S2398265
t.deboer-6@student.utwente.nl

University of Twente

Industrial Engineering and Management
Drienerolaan 5
7500 AE, Enschede
(053) 489 9111

Faculty of Behavioral, Management and Social sciences (BMS)
Industrial Engineering and Management

Supervisors

Dr. J. M. J. Schutten (Marco)
Dr. E. A. Lalla (Eduardo)

Company

Koskamp B.V.

Supervisor

Nathan van Dijk

Publication date

September 2024

First edition
Number of Pages: 50
Number of Appendices: 3

This report was written as part of a research thesis in module 12 of the Industrial Engineering and Management educational program.

Management Summary

The goal of this research is to reduce travel distances in delivery routes, thereby lowering fuel costs and CO_2-emissions for Koskamp B.V. in Den Ham. Koskamp B.V.'s branch is the sale and purchase of automotive parts, serving companies as garages and car dealers across the Netherlands. The head office operates from Den Ham and together with 12 other additional locations serve as depot for delivering.

The company faces high fuel costs and CO_2-emissions due to the long delivery routes driven. For delivering automotive parts, Koskamp created 10 areas. Each area is a pre-compiled list of addresses. The addresses in an area are relatively close to one another, as the addresses that are put into one list are based on zip code. The routes are generated 6 times a day, as there are 6 time slots in one day where customers can have their products delivered. The goal is that Koskamp delivers the ordered items within 90 minutes from the start of a time slot. Because of the 10 fixed areas, 10 routes are generated per time slot and these routes can only be adjusted manually. The problem of high fuel costs and CO_2-emissions arises as Koskamp uses a routing generation method that utilizes this fixed allocation of addresses to areas. The goal of this research is to improve the delivery situation of Koskamp B.V. by optimizing the route planning to solve current issues in the delivery situation.

Each location has its own vehicles and is responsible for deliveries within allocated areas. Koskamp sells automotive parts varying from filters to car tires and more. One thing that sets Koskamp apart from its competitors is their fast delivery time. The company aims to deliver the products within 90 minutes from the start of the time slot, if the products are in stock. The vehicles are categorized in small, medium and large vehicles. Most of these are diesel vehicles, however, they also own a small set of electric vehicles. Koskamp only provides service to companies and not to individuals.

To solve the problem the company faces, we conduct a literature research. We search for optimization methods, that possibly fit in Koskamp's strategy. The most used optimization problems are the Travelling Salesman Problem (TSP) and the Vehicle Routing Problem (VRP). The TSP is a variant of the VRP, which has many other variants. Some examples of VRP variants are the Classical Vehicle Routing Problem (CVRP), the VRP with Time Windows (VRPTW) and the Vehicle Routing Problem with Pickup and Deliveries (PDVRP). Next, we search for methods that help solving the problem. We analyze exact, heuristic and metaheuristic methods and their variants. Exact methods provide optimal solutions but can be computationally intensive for large problems. Heuristic methods give proper solutions, but are not necessarily optimal. Metaheuristics find good solutions for complex problems through repetitive, intelligent search methods. We also search for available code online, to apply in the model.

After we review the literature search, we decide to use the CVRPTW with the Path Cheapest Arc method (PCA). The CVRPTW comes closest to the situation of Koskamp, as Koskamp's main goal is to deliver within a specific time, and sometimes aiming to deliver to specific customers within a certain time window during the time slot. The PCA method is very helpful due to the short calculation time. It is relatively simple but it still matches real life situation very good. PCA focuses on selecting the shortest path from the current node to the next, which provides efficient and practical solutions for real-world scenarios. We choose the PCA heuristic over an exact method, as it will generate routes quicker, which is necessary for Koskamp, because routes need to be generated within a short time before the vehicles leave.

We conduct experiments with the model. The input requires coordinates from the location where the customer is located, the number of products that that customer ordered for that time slot, the time

windows and vehicle capacity. In the input we can adjust the average service time per customer and the average speed of the vehicles. With the coordinates, the algorithm generates a distance matrix by utilizing the Haversine formula. After this, we adjust the distance matrix by a correction factor of 1.32 to better fit real-world distances. The time matrix is generated by taking into account the average speed of the vehicles. After we conduct the experiments, we conclude that the impact of the model varies a lot depending on the number of addresses. In cases where the number of addresses is below 50, the number of vehicles remain 10 or lower. In cases where the number of addresses exceed 50, the model uses more than 10 vehicles. At first sight, it might seem worse than the old situation, as the old situation utilizes 10 vehicles per time slot at any time slot. However, after we conduct experiments with real-world data, we see that the vehicles in the old situation exceed the time limit of 90 minutes a lot more often when the number of addresses is above 50. We find it challenging to give exact numbers on the level of improvement, as the improvement varies per experiment. We calculate the reduction in salary and depreciation costs for a random day. We base the depreciation costs on distance travelled by a vehicle. We conclude that the model reduces the salary costs by 14.1% and the depreciation costs by 11.3% for that day.

The study evaluates Koskamps's current routing generation to be inefficient. By adopting the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) and the Path Cheapest Arc (PCA) heuristic, Koskamp is able to achieve improvements, reducing the number of vehicles, the driven distance and the number of times the time limit gets exceeded. We recommend to Koskamp to start implementing the new model in phases. First, apply the model in practice to real-world time slots and analyze its performance. Based on the findings (change in travel distance, costs, etc.) adjust the model to improve it. When the model is refined once more to better match real-world conditions, a software could be developed to integrate the model into Koskamp's strategy. Besides adjusting the model based on practical experience, future research can focus on adapting it to handle returns and situations where the model does not fully align with real-world data. Koskamp might explore other advanced routing software, such as RoutePilot or RouteLogic, to see if they offer better solutions for their needs.

Preface and Acknowledgements

Dear reader,

Thank you for taking the time to read this thesis. This thesis is the final project for my Bachelor's degree in Industrial Engineering and Management at the University of Twente. The title of this thesis is "Shortening the Delivery Routes of Koskamp B.V.". Koskamp B.V. is a company that specializes in the purchase, sales and delivering of automotive parts.

First, I want to thank Nathan van Dijk for allowing me to write this thesis for Koskamp B.V., conduct my research at the company and helping me with his guidance. I hope that my research contributes to the efficiency of the delivery situation.

Next, my gratitude goes towards my first supervisor, Marco Schutten, who provided valuable feedback throughout this research. This guidance helped me improve my academic writing and research skills, leading to the completion of this research. I would also like to thank my second supervisor, Eduardo Lalla, for providing feedback during the final stages of this research.

Lastly, I would like to express my appreciation to my family and friends for their support during the writing of this thesis.

Contents

List of Figures

List of Tables

List of Constraints

1 Introduction

This study focuses on optimizing the delivery operations at Koskamp B.V., a company specializing in the sale of automotive parts, headquartered in Den Ham, the Netherlands, with multiple locations all over The Netherlands. Koskamp faces a challenge of improving the efficiency of its delivery system, currently hindered by issues such as inefficient route planning, high fuel costs, and high CO_2-emissions.

In this chapter, we introduce the company and the problem. Section 1.1 provides a description of the company Koskamp B.V. Section 1.2 identifies the challenges in Koskamp's situation, including inefficient route planning and associated costs. Section 1.3 aims to outline the main goal of this research. Section 1.4 states the research questions guiding each phase of the study. Section 1.5 addresses the scope and limitations of the research.

1.1 Company description

The company is called Koskamp B.V. We refer to them as Koskamp. Koskamp's main business is the purchase and sale of automotive parts and other attributes related to cars. Their clients are companies that are in need of these parts. Their head building is in Den Ham, where main operations and deliveries are managed. They have 12 locations in all of the Netherlands and the 13th location is in the making. The other locations function as delivery depots only. Each location has its allocated areas where they deliver automotive parts. An area consists of certain addresses that are allocated to this specific area, based on where the addresses are located. Besides the deliveries, Koskamp also picks up returns when a customer wants to return a product. Their clients are limited to companies only, which in most cases are garages or car dealers. In this thesis the focus is on improving the delivery situation.

1.2 Problem of the company

Koskamp faces challenges with its current delivery system. For delivering the parts, the location in Den Ham made 10 delivery areas (relatively) near Den Ham. In this context, an "area" refers to a pre-compiled group of addresses created by Koskamp. The addresses in an area are relatively close to one another, as they are based on their zip code. At this location, routes are created to visit each of the 10 areas 6 time a day during specific time slots, provided there is at least one scheduled order in each area for each time slot. The time from leaving until the next time slot starts is 90 minutes. The first time slot is at 08:00 and the last one ends at 17:00. An address that is allocated to an area is allocated to that same area in every time slot and is therefore never allocated to other areas, unless done manually.

One thing that sets Koskamp apart from the competition is their fast delivery time. When a customer places an order, the customer can either choose from the six different time slots, or choose the option 'as fast as possible' and gets allocated to the first time slot after the order is placed. However, the system generates routes per area and an address is always allocated to the same area. This leads to inefficient planning, where some routes end up with too many addresses, that can not be handled within 90 minutes, while other routes only have one address. The only option that the planners currently have is to put addresses in another area manually, based on feeling that comes from experience. However, if this is ideal for the number of kilometers driven is not clear.

These issues result in long delivery routes for Koskamp, which on its turn results in high fuel costs and more CO_2-emission. This is the problem that Koskamp wants to address.

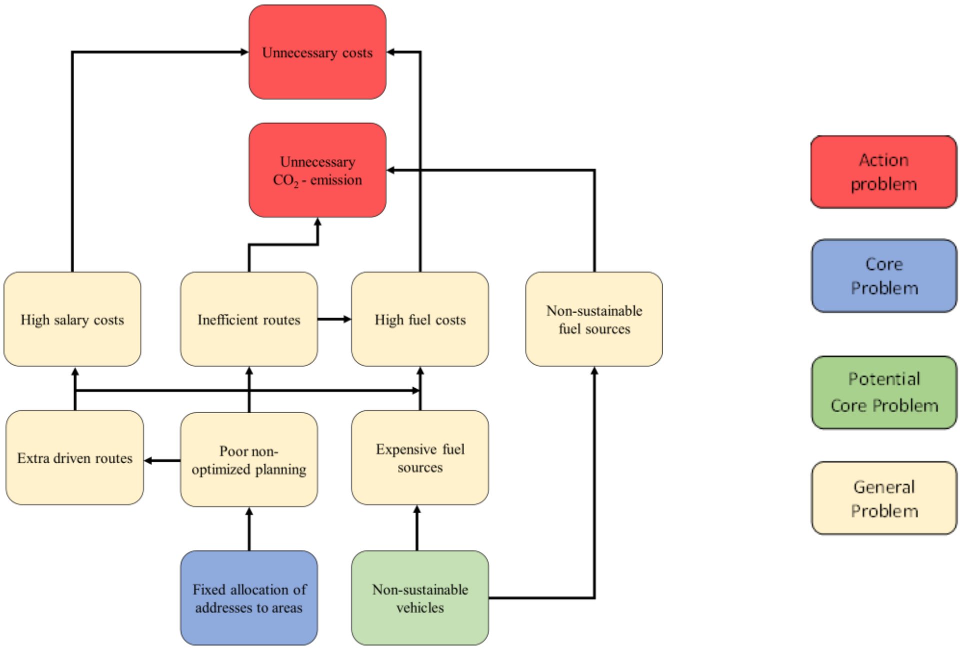


Figure 1-1: Problem Cluster

We develop a problem cluster to get to the core problem (see Figure 1-1). A problem cluster identifies relationships between multiple problems that a company faces. The core problem is the underlying issue that eventually leads to the action problem. The action problem is the main issue that the company wants to resolve. There are four subproblems leading to the problem the company faces.

Salary costs

The first sub-problem is the cost of salaries. The salary costs are unnecessarily high as some routes could be combined with others, reducing the number of routes needed. When an unnecessary route is generated, they need more drivers, even though the addresses from that extra route might fit into another route.

Inefficient routes

The second sub problem is the inefficiency of routes. Addresses are allocated to that same area every day. This fixed allocation is not efficient, as an address might fit better in another area on a specific day. There is a need for flexibility in how the routes are generated. Customers should be allocated each day to the area that is most efficient in terms of travel distance. This poor planning occurs, because the current system does not fully use the potential for optimizing routes.

Fuel costs

The third subproblem is the high cost of fuel, which also arise from the inefficient routes. The fuel costs are also high because of diesel that is relatively expensive in comparison to electric vehicles (Meijs, 2019).

Sustainable fuel sources

The fourth sub-problem comes from the same underlying cause as the third one. The company has limited electrical vehicles, which means that they mostly rely on diesel-based vehicles. These vehicles are not very sustainable in comparison to electrical vehicles, which is a potential core problem. Because of the use of non-sustainable fuel sources, there will eventually be more CO_2-emissions in comparison to electrical vehicles. This will lead to high CO_2-emissions.

There is a clear issue that requires a solution. In the cluster, the two red boxes display the action problem. There is a need to lower the costs and CO_2-emisions. This issue stems from two potential core problems leading to this. The two potential core problems are: a fixed allocation of addresses by the current planning system and non-sustainable vehicles. We choose to focus on the fixed allocation issue, as Koskamp addresses that this research will not lead to radical change of non-electric vehicles to electric vehicles.

1.3 Aim of research

The aim of this research is to create an advice for a more efficient delivery system for Koskamp B.V. by optimizing the route planning. The current system leads to unnecessarily high distances driven, high fuel costs, high CO_2-emissions and high salary costs. This research aims to identify opportunities for optimizing routes to reduce the total distance traveled, thereby lowering costs and lowering CO_2-emissions. By reaching a more flexible delivery system, the company aims to enhance the efficiency and get closer to a sustainable way of working. The target is to achieve a reduction of 5-10% in the average daily distance traveled, fuel costs, and CO_2-emissions.

1.4 Research questions

For the problem approach, we formulate certain research questions. We divide the questions across chapters to eventually solve the problem.

Chapter 2 describes the first step to get a good view of the current situation. The primary research question for this chapter is: "What are the key aspects of Koskamp's current operational setup?" We come up with sub-questions for this primary research question.

- What are the specifications of the vehicles they are currently using?
- What type of customers does Koskamp have?
- How are the routes currently planned?
- How does the ordering system work?
- How do the drivers currently experience the routes, in terms of stress and schedule tightness?

Chapter 3 reviews the outcome of a literature search that aims for finding appropriate literature for the research. The literature helps determine what to solve and how to solve it. The primary research question for this chapter is: “How can routing problems be solved, and what available methods and tools can be utilized?”. The following sub-questions are:

- What are the type of optimization problems and key methodologies used in routing problems?
- What programming code is already available for routing problems?

Chapter 4 outlines the choice of an appropriate model. We need the literature from Chapter 3 to do this. We need to define variables, the objective function, constraints and other aspects related to optimization. The aim is to choose an appropriate model with an algorithm that is most applicable to the specific needs of Koskamp. The research question for this chapter is “What is an appropriate model to use for the situation of Koskamp?”

In Chapter 5, we test the algorithm that is applied in the model by doing experiments. In this phase, we add value to performance indicators like traveled distance, travel time, costs and emissions and compare this to the old situation. This gives an idea on how much the situation improves if the company decides to implement the advice. The primary research question for this chapter is: “How big is the improvement compared to the old situation when looking at the performance indicators?”

In Chapter 6, the last step is to give an advice to Koskamp and explain to them how and why this solution will fit in their strategy. If the model leads to big improvements, the company can consider to use the advice we create from the research. This advice comes with a model that can be implemented in the strategy of the company.

1.5 Scope and Limitations

For this thesis, the focus is on the headquarters of Koskamp: Den Ham. We use data from that location only. Koskamp van easily adjust the model to the situation of other locations, but we will not do this in this research.

A limitation regarding the research is that it is not possible to make a completely new planning tool for the company to use in their strategy, but create a model that can be implemented through further research. This prototype model generates routes while adhering to constraints that apply to Koskamp's situation. Another limitation is that it is not possible to do this for every location of Koskamp, as this will cost too much time. That is why we only conduct research at the Den Ham location.

We choose to use python as a programming language on forehand because Python is the most generally used program language and is relatively easy to learn (Okeke, 2023).

2 Current situation of the company

This chapter addresses the research question "What are the key aspects of Koskamp's current operational setup?". Section 2.1 outlines the locations Koskamp has in the Netherlands. Section 2.2 elaborates on the types of products Koskamp sells and the specifications of the vehicles used for deliveries. Section 2.3 outlines the customers of Koskamp. Section 2.4 covers how the route planning is currently done. In Section 2.5, we answer the primary research question, together with the corresponding sub questions of this chapter.

2.1 Locations of Koskamp

Koskamp currently operates from 12 locations in The Netherlands, while a 13th location is in the making. Their main building is located in Den Ham. The additional locations include Zutphen, Emmen, Bilthoven, Kampen, Leeuwarden, Arnhem, Lelystad, Groningen, Nijmegen, Assen, and Steenwijk. The 13th location will be in Hengelo. The locations obviously have interaction with each other, but not with regards to delivering to customers, as each location has allocated its own areas for delivering. All the vehicles that leave from a certain location at a specific time slot, will also return to that location before the end of that time slot.

2.2 Aspects of Koskamp

2.2.1 Products

The products sold at Koskamp are automotive parts, which can vary from cleaning tools to car tires. Koskamp aims to deliver within 90 minutes from the start of a time slot, provided the products are in stock. The company states that 95% of the products that are ordered is in stock. The other 5% Is assumed to be there the next day. However, there are exceptions that take a bit longer.

2.2.2 Vehicles

Type of vehicles
The vehicles are subdivided in three categories: small, medium and large. The small vehicles are named ‘bestelbus’ (delivery van), the medium ones are called ‘bus’ (van) and the large ones are called ‘Bus XL’ (XL van).

Fuel
The fuel used for the vehicles in Den Ham is diesel. There are three electric vehicles at this location: 2 Renault Kangoo’s and 1 Citroen E Berlingo. Table 2-1 displays the electrical vehicles in green.

Costs
The costs associated with delivery include fuel costs, salary costs, and depreciation costs. Fuel costs vary depending on the type of car (and sometimes even within the same type), depreciation costs vary based on the size of the car, while salary costs are uniform for all drivers at €18.84 per hour.

Emissions
The amount of CO_2 that is emitted depends on the type of car (and, similar to fuel costs, can also vary within the same type). We use the manufacturer's specification for the emissions.

Table 2-1 shows the specifications of the vehicles.

Vehicle	Type	Size	Costs per KM (€)	CO_2 per KM (g)	Diesel per KM (L)	Current Diesel price per L (€)
0	Renault Kangoo V-417-KP	Bestelbus (small)	0.080152	112	0.043	1.864
1	Renault Kangoo VHF-59-H	Bestelbus (small)	0.08388	117	0.043	
2	Opel Combo V-93-DFT	Bestelbus (small)	0.109976	155	0.059	
3	Opel Combo VVR-03-P	Bestelbus (small)	0.109976	155	0.059	
4	Citroen Berlingo VKZ-11-V	Bestelbus (small)	0.098792	140	0.053	
5	Renault Express VKZ-44-K	Bestelbus (small)	0.078288	111	0.051	
6	Renault Express VNB-72-P	Bestelbus (small)	0.078288	111	0.051	
7	Renault Express VPJ-56-Z	Bestelbus (small)	0.078288	111	0.051	
8	Renault Express VPS-26-R	Bestelbus (small)	0.078288	111	0.051	
9	Renault Express VXX-96-K	Bestelbus (small)	0.078288	111	0.051	
10	Fiat Doblo VPT-37-P	Bestelbus (small)	0.134208	169	0.072	
11	Renault Trafic VRN-95-N	Bus (medium)	0.134208	189	0.072	
12	Renault Trafic VVF-45-L	Bus (medium)	0.134208	189	0.072	
13	Renault Trafic VVZ-23-H	Bus (medium)	0.134208	189	0.072	
14	Citroen E-Berlingo VSP-28-N	Bestelbus (small)	0.0174	0		
15	Peugeot Partner VTN-34-J	Bestelbus (small)	0.080152	112	0.043	
16	Kangoo van E-Tech Electric VVH-10-F	Bestelbus (small)	0.0174	0		
17	Kangoo van E-Tech Electric VVJ-09-G	Bestelbus (small)	0.0174	0		

Table 2-1: Vehicle Characteristics

* The green rows are electrical vehicles.
* There are currently no large vehicles at the Den Ham location.
* Same type of models can have different specifications, as some models may be older.

2.3 Customers of Koskamp

2.3.1 Type of customers

Koskamp has a policy that sales are made only to companies and not to individuals. Most organizations are described as 'car companies' which are garages where cars are repaired. Other customers are car dealers for example.

2.3.2 Ordering process of Koskamp

When ordering at Koskamp, a customer can select the products they want to order on the website. When they have selected all the desired products, they are able to choose a time slot in which they want the products to be delivered. There is also an option that says 'as soon as possible', and then the customer will be allocated to the first time slot after the order is placed. Some customers have the opportunity to order at every time slot, while others only have the opportunity to choose from less time slots. This is based on their ordering frequency: frequent customers have more delivery options per day.

2.4 Route planning

2.4.1 Current route characteristics

The delivery system at Koskamp is currently focused on fixed allocation. To explain what this means, we explain the components related to this.

Area

An area in this context refers to a pre-compiled group of addresses created by Koskamp. The addresses within an area are relatively near each other, in comparison to addresses in other areas. This is because the creation of the group of addresses, is based on postcode. Currently, the Den Ham location has 10 of those area's.

Address

When an address is mentioned, it refers to an address of one of the customers of Koskamp. An address is always part of the same area, unless the planner changes the address to another area, after the routes are already generated. This is usually done if the planner thinks that address is not fitting within 90 minutes within a specific route.

Route

A route refers to the path taken from the head building to deliver at customers and then return to the head building. The routes are determined by the MobileNXT-system. We explain MobileNXT in detail in Section 2.4.2.

Time slot

The time slots are the specific periods allocated for deliveries at a particular time of the day. At the Den Ham location, there are six such time slots. The first one starts at 08:00 and the last one ends at 17:00. Customers can choose in which time slot they want their products to be delivered. Most customers who frequently order, have the possibility to choose from all time slots, while some customers can only choose from two or three slots. At the Den Ham location, the time slots have a duration of 90 minutes.

UNIVERSITY OF TWENTE.

Time window

Some customers made an oral agreement with Koskamp, so that their delivery will be delivered within a certain time window within a time slot.

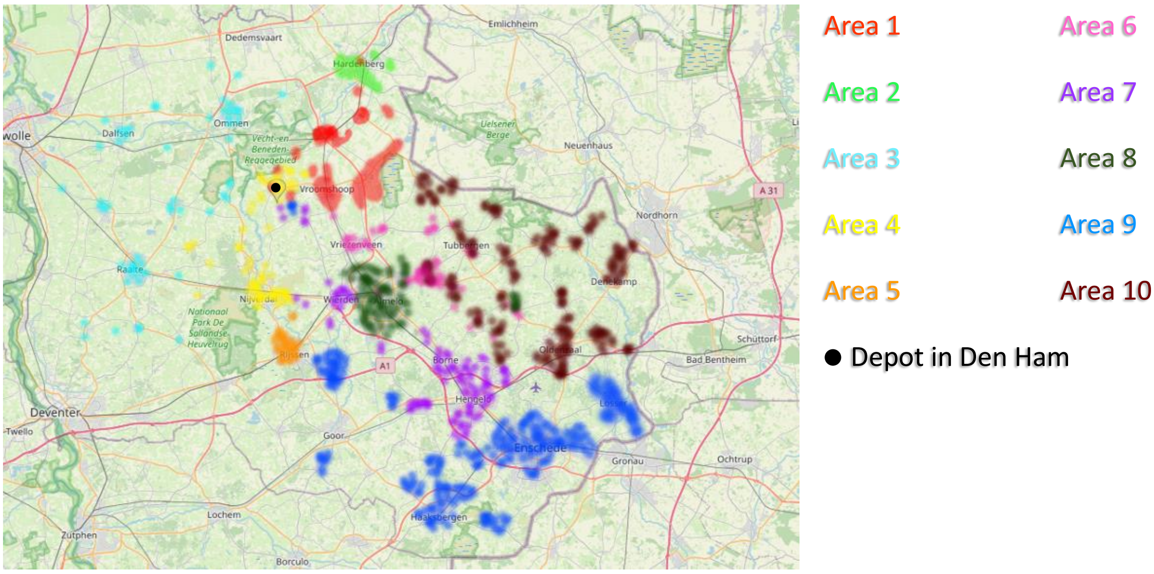


Figure 2-1: Areas visualized

Currently, the 10 areas are linked to 10 routes. We visualize the areas in Figure 2-1. The vehicles that travel to areas 1, 2, 3, 4, 5, 6 and 8 need to be back at the depot within 90 minutes. Areas 7, 9 and 10 are so far away that a vehicle cannot return within 90 minutes, and that is why they are designated as 3-hour routes. This means that the delivery must be done within 90 minutes, but the driver does not need to be back within those 90 minutes. The vehicles that leave for these areas do leave every time slot. This means that for these areas more drivers are needed than the other areas, as the driver will not be back in time for the start of the next time slot.

2.4.2 Current planning system

Currently, the system MobileNXT is used for operational purposes regarding the planning of the routes. The MobileNXT system uses an algorithm that does not take into account capacity and time constraints, and therefore creates routes that are not well-suited to Koskamp's requirement of delivering within 90 minutes.

MobileNXT is connected to a scanner that is operated by the drivers. This scanner contains the addresses in order for delivery and is also used for scanning the products after they have been delivered. An additional system, Silicos, makes receipts that are made after a customer places an order. MobileNXT uses this as input. On these receipts the specific area where the customer belongs to is noted and a time slot is indicated, based on the preference of the customer.

After the input from Silicos is put into MobileNXT, the driver indicates on the scanner that he is present at the depot. The articles that have to be delivered are all mixed up on the scanner. Now, the driver scans all the articles that have to be delivered, to check if they are all present. If everything turns out to be there, the scanner will indicate that the check has been finished. The person in charge of planning will give permission to this driver to leave. After this, the system will determine the routes for the corresponding area.

When arriving at a customer, the scanner shows the products that have to be delivered only. These products get one last scan and are given to the customer. When this is finished, the system states the delivery done. Then at the next customer, the same routine takes place. When a customer is not present, the driver can indicate this in the system.

When all customers have received their products and any returns are taken back, the scanner indicates that the driver may return to the depot.

2.4.3 Experience from drivers

To gain better insights into the routes, it is important to gather information from drivers who have the most experience with the current routes. That is why they can give insights on the efficiency of the current routes (See Appendix A.1 on page 49 for the full interview with one of the drivers).

MobileNXT routes can sometimes be inefficient. An issue arises when drivers are directed along intersecting routes instead of a direct circular path. This can cause confusion and inefficiency.

Areas 1, 2, 3, 4, 5, 6 and 8 are the areas with vehicles that need to return to the depot within 90 minutes. The tightest areas when it comes to this group of areas, are area 2 and 3. This is because these areas are relatively far away from the depot in comparison with the other areas from this group. From experience, the driver indicates that these areas contain more orders on average than the other areas. Area 3, which includes Raalte, Heino, Ommen, and Lemelerveld, have challenges due to the wide distances between these locations, making it difficult to complete deliveries within 90 minutes. Also, in Area 2, where all addresses are located in Hardenberg, the travel time alone amounts to 25 minutes by car. This leaves only 40 minutes to fulfill the orders. This problem is less frequent in areas 7, 9 and 10, as they do have enough time to return to the depot after the deliveries are done within 90 minutes.

The driver mentions that with reference to the areas they have to drive to, they prefer to drive to the same area as often as possible rather than driving different routes each time. This approach will enhance the relationship with customers, as customers will see a familiar face, and the driver will become more familiar with specific needs the customer might have.

Other things that bothers the drivers is the fact that the MobileNXT-system does not take into account the other traffic. The system expects them to be there at a certain timestamp, even though this is not always possible due to unexpected traffic circumstances.

2.5 Chapter conclusion

To conclude the chapter, the research questions are answered.

What are the specifications of the vehicles they are currently using?
Table 2-1 in Section 2.2.2 shows the specifications of the vehicles.

What type of customers does Koskamp have?
Koskamp's customers are exclusively companies, and cannot be individuals. Most customers are car companies, such as garages where they do a lot of car repair, and car dealers.

How are the routes currently planned?
Koskamp's route planning is currently done by the MobileNXT system and manual adjustments to this system if the person in charge of the planning sees that certain deliveries cannot be done within the 90 minutes or is not satisfied in general. The routes are generated separately for each of the 10 areas, rather than combining addresses from multiple areas, resulting in 10 independent routes.

How does the ordering system work?
Customers place orders on Koskamp's website, selecting desired products and choosing a time slot. There is a delivery option for 'as soon as possible', which assigns the customer to the first available time slot.

How do the drivers currently experience the routes, in terms of stress and schedule tightness?
Drivers have identified inefficiencies in the MobileNXT system, such as non-optimal routing. They prefer driving the same routes to enhance customer relationships and are affected by the system's failure to account for other traffic. The most challenging areas are 2 and 3 as they have heavy order volumes and are relatively far away for a vehicle that must return to the depot within 90 minutes.

3 Theoretical framework

In this chapter, we research literature of routing optimization problems while we consider the primary question of this chapter: "How can routing problems be solved, and what available methods and tools can be utilized?" Starting with Section 3.1, we explore different types of routing optimization problems, including the Traveling Salesman Problem (TSP) and multiple variants of the Vehicle Routing Problem (VRP). Section 3.2 focuses on the key methodologies that solve these problems, like exact methods, heuristic approaches, and metaheuristics. In Section 3.3, we review online code that is helpful for solving routing problems. Section 3.4 outlines a conclusion summarizing the insights we gain from the literature.

3.1 Types of routing optimizing problems

One of the most well-known challenges in routing optimization is the Vehicle Routing Problem (VRP). It is derived from the Traveling Salesman Problem (TSP), which only seeks the shortest route to visit multiple nodes and return to the starting point. VRP aims to optimize routes while aiming for a certain goal which can vary. These goals can include minimizing travel time between nodes or reducing the overall travel distance.

In addition to the TSP and VRP, the VRP itself is divided into multiple types (Anita Agárdi, 2022). We elaborate on the TSP and 5 different types of the VRP in this section.

3.1.1 Traveling Salesman Problem

The Traveling Salesperson Problem (TSP) involves visiting a set of locations exactly once, starting and ending at the same location, while minimizing the total travel distance (Urquhart, 2022). Unlike in vehicle routing problems, there is only used one vehicle in TSP: it exclusively focuses on determining the shortest path that connects all nodes in a route. Each node represents a location and needs to be visited exactly once before returning to the depot.

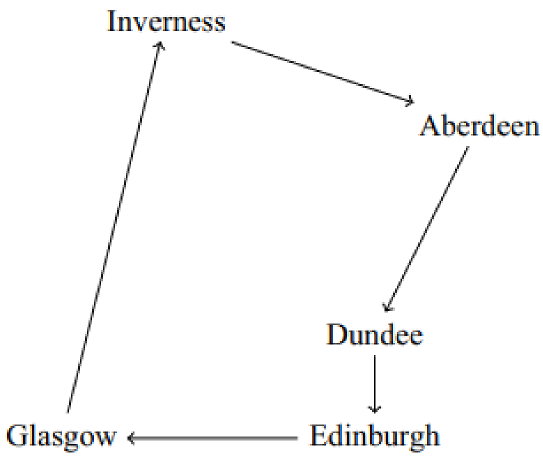


Figure 3-1: TSP example (Urquhart, 2022)

In Figure 3-1, there is an example of a TSP with 1 depot and 4 nodes (Urquhart, 2022). From the depot (see: Dundee) the shortest distance is chosen, which in this case is Edinburgh. From Edinburgh, the shortest distance to one of the remaining nodes is to Glasgow and in this way it goes on.

3.1.2 Classical Vehicle Routing Problem

The Classical Vehicle Routing Problem focuses on the minimizing of the distance traveled by selecting a certain number of vehicles (Ramser, 1959). The Classical VRP is also known as the Capacitated VRP (CVRP). The CVRP, a variant of VRP, adds a constraint that vehicles have a limit to carrying load (Ilhan, 2020). Characteristics of this variant are that there is only one central depot, each vehicle only travels one route and every vehicle has identical characteristics. The start and end of the route is at the depot and the capacity of the vehicles cannot be surpassed. This initial version of the VRP resulted in numerous variants in the spectrum of VRP methods (Kris Braekers, 2015). The CVRP is visualized In Figure 3-2.

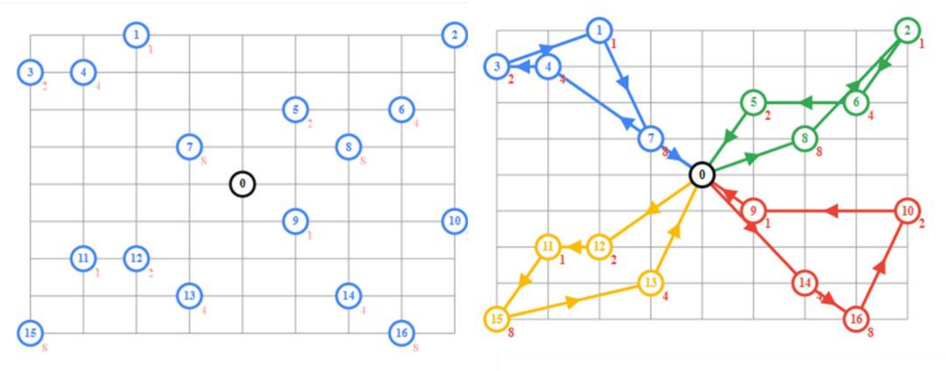


Figure 3-2: Visualization of an output of a CVRP (OR-ools, 2021)

3.1.3 Vehicle Routing Problem with Time Windows (VRPTW)

Vehicle Routing Problems with Time Windows (VRPTW) keeps into account a certain time window that a vehicle should be present for delivery at a customer. These time windows can vary from customer to customer. The goal is often to reduce the total travel time or total travel distance of a route. (Vitória Pureza, 2011).

3.1.4 Vehicle Routing Problem with Pickup and Delivery (VRPPD)

The Vehicle Routing Problem with Pickup and Delivery (VRPPD) involves designing vehicle routes that start and end at the depot. Pickup and deliveries within each route must be handled by a single vehicle, ensuring that the total load along the route never surpasses a specified maximum (Arild Hoff, 2009).

3.1.5 The Periodic Vehicle Routing Problem with Time Windows (PVVRPTW)

The Periodic Vehicle Routing Problem with Time Windows (PVRPTW) involves scheduling routes for vehicles to serve customers within specific time windows over a longer period, like daily or weekly. The challenge is to plan efficient routes that meet these time constraints while minimizing overall costs, distance or time (Suresh Nanda Kumar, Ramasamy Panneerselvam, 2012).

3.1.6 The Multi-Depot Vehicle Routing Problem (MDVRP)

The Multi-Depot Vehicle Routing Problem (MDVRP) is about finding the best routes for vehicles starting from several depots to visit all customers. The aim is to reduce the total distance or time of the routes while managing the tasks between different depots. This adds some complexity because coordination between multiple starting locations is necessary.

3.2 Key Methodologies

The Vehicle Routing Problem is a NP-hard problem (Shuhan Kou, 2023). NP-hard (nondeterministic polynomial time) means that the solution time for the optimal solution rapidly increases as the size of the problem (the number of nodes) increases (Hochba, 1997).

3.2.1 Exact

Exact solutions explore all possible solutions problems in general, so also for routing problems. It searches intensely through a certain area. Exact algorithms guarantee to find optimal solutions within given constraints. As the VRP is NP-hard, this indicates that the bigger the problem becomes, the more difficult it is to solve it with an exact approach.

An example of an exact way of optimizing a model is Mixed Integer Linear Programming (MILP). When tackling MILP problems, a common used approach is a branch-and-bound algorithm.

Branch-and-bound is an exact method. It involves branching, where a problem is divided into smaller subproblems, and bounding, where bounds on the optimal solution of each subproblem are determined. If the bound of a subproblem indicates it does not produce a better solution than the best solution so far, it is removed. This process continues until all subproblems are solved or removed, and the solution that remains is the optimal one (Kianfar, 2011). Figure 3-3 displays the Branch-and-Bound-algorithm in tree structure.

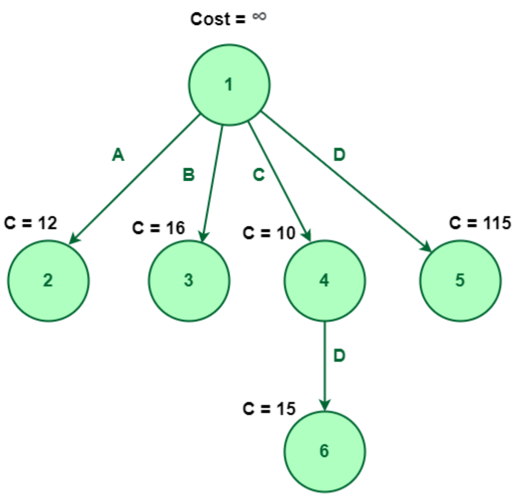


Figure 3-3: Visualization of Branch-and-Bound-algorithm (Geeksforgeeks, 2023)

Dynamic Programming (DP) is an approach to solving problems by breaking them into smaller, simpler steps. It stores solutions to these steps so they do not need to be recalculated, making it efficient for finding optimal solutions to complex problems. Each step is based on previous solutions, helping to solve the bigger problems by solving the smaller problems first. DP is effective for problems where solutions to smaller parts will help to find the best solution overall (Velimirovi, 2023). Figure 3-4 shows how each node evaluates the best next node from the remaining options available.

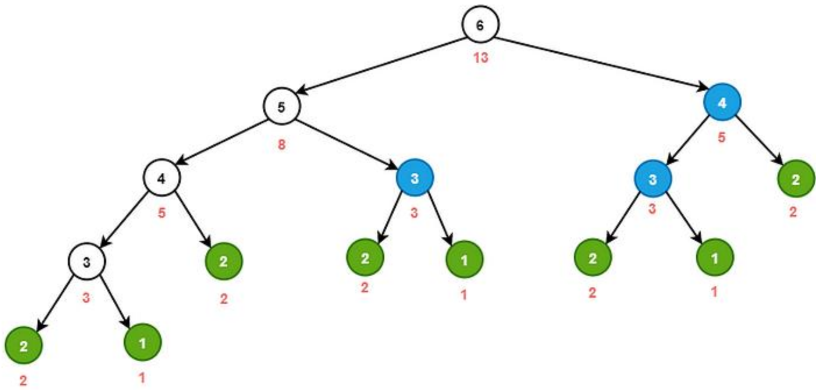


Figure 3-4: Visualization of DP-algorithm (Tobbileh, 2021)

3.2.2 Heuristics

NP-hard problems are more natural to be tackled by the means of heuristic algorithms. Heuristic algorithms find reasonably good solutions. There are many available heuristic algorithms available. Heuristics aim for a proper solution, but it may not always be the most optimal (L Zeng, 2006).

Cluster first, route second first divides the customers into areas and then determines a route for each area. One of the approaches Is the Gillet and Miller's sweep-algorithm. This approach starts with picking a start location and then conducts a forward or backward sweep, that determines clusters (Lalla-Ruiz, 2022). Figure 3-5 visualizes this approach.

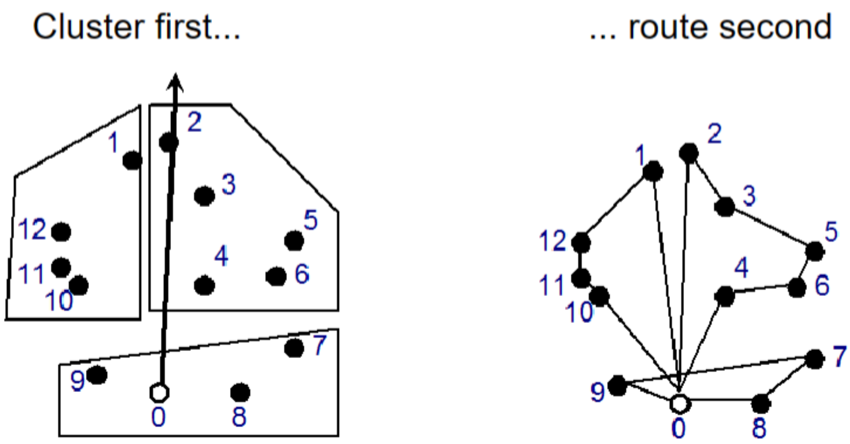


Figure 3-5: Visualization of the cluster first, route second approach (Lalla-Ruiz, 2022)

Route first, cluster second is an approach that first determines the optimal route for all nodes. Beasley's method starts by solving the Traveling Salesman Problem (TSP), where it minimizes the travel distance to visit all specified locations one time. Once the TSP solution is determined, it then determines the shortest path connecting these nodes in a way that minimizes the total distance traveled, and then solves the whole routing problem. (Lalla-Ruiz, 2022). Figure 3-6 visualize this approach.

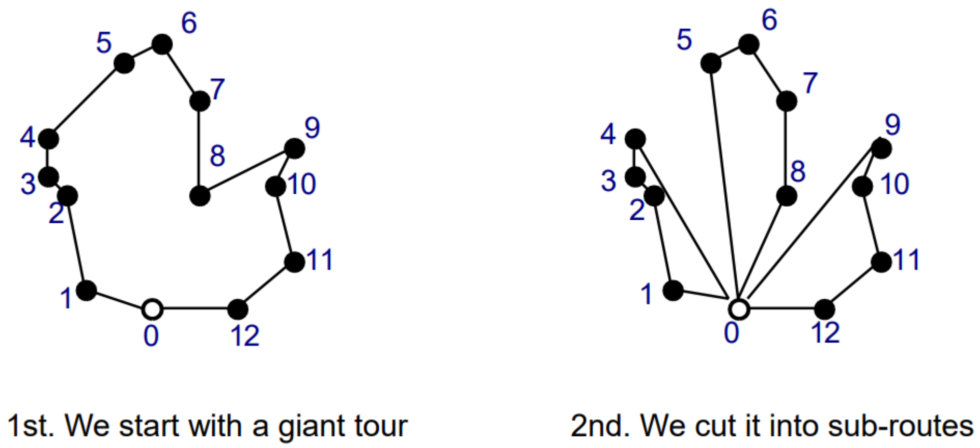


Figure 3-6: Visualization of the route first, cluster second approach (Lalla-Ruiz, 2022)

One of the most famous methods to solve VRP heuristics is the Clarke and Wright (CW) savings heuristic. This approach starts by assigning each node to a separate route and then merges these routes if it results in a reduction in the total route cost. Clarke and Wright could come in helpful due to its simplicity in matching real-world situations (Cordeau, 2002). Figure 3-7 visualizes the CW savings heuristic.

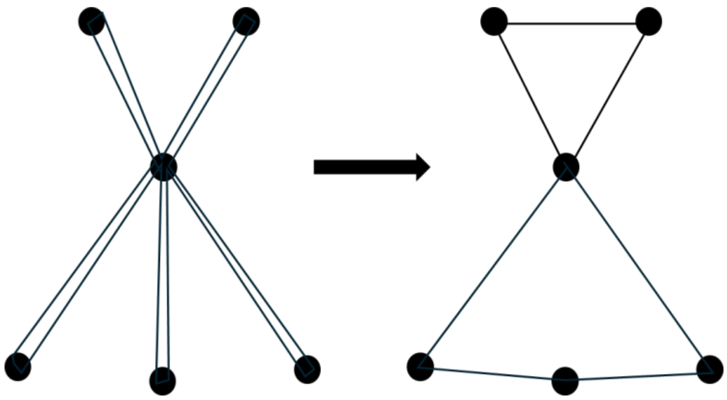


Figure 3-7: Visualization of the Clarke and Wright savings algorithm

Another heuristic method is the Nearest Insertion Heuristic. It selects the customer that is closest to an existing route for insertion. Nodes are added by the node with the nearest insertion positions. It will give an efficient method for solutions, but just like other heuristics, it will not necessarily be optimal (Lalla-Ruiz, 2022). Figure 3-8 visualizes the approach.

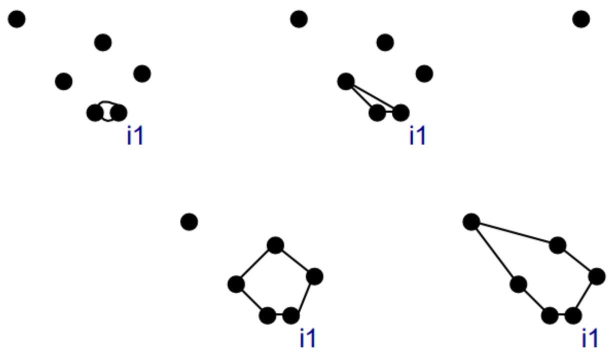


Figure 3-8: Visualization of the Heuristic Nearest Insertion method (Lalla-Ruiz, 2022)

A heuristic method that is similar to the nearest insertion heuristic is the Path Cheapest Arc (PCA). This method is used for constructing efficient routes in VRP by selecting the least expensive node as the next node in the current created route. The cheapest node is usually in terms of distance or time. It is a fast way of creating a route with a good, but not optimal solution. It is of good use when trying to simulate a real life situation in an efficient way (Dihin Muriyatmoko, 2023). The biggest difference with the Nearest Insertion Heuristic, is that the PCA can also aim for the shortest time or lowest costs.

3.2.3 Metaheuristics

There are two types of heuristics: (standard) heuristics and metaheuristics. Metaheuristics allow very complex moves and enable recombination of solutions. Complex moves means a more advanced approach and recombination refers to the fact that it is able to combining elements from different solutions to create new solutions (Sorensen, 2013).

With reference to VRP, Tabu search stands out (Cordeau, 2002). Tabu search starts with an initial solution and explores other options by making small changes to the initial solution. When a change turns out not to improve, it can still be allowed if future changes to that option will result in improving the initial solution. It is very effective for large and complex VRP's.

A Genetic Algorithm is an optimizing technique that generates a set of possible solutions repeatedly. It starts with random route solutions, then improve them iteratively by the idea of selection, crossover (blending in the routes from more efficient solutions), and mutation (to add variety). (Niels Wouda, 2024).

3.2.4 Distance Matrix Construction: Vincenty and Haversine Formulas

As the distance between nodes for conducting a VRP is needed for distance matrixes, there is a need of constructing the matrices quickly. The distances can be approximated by using several formulas. Two variants for this are the Vincenty Formula and Haversine Formula (M. Chalela, E. Sillero, L. Pereyra , 2021). The Vincenty and Haversine formulas are both used to find the distance between nodes, which are represented by coordinates. They both do it in different ways. The Haversine assumes the earth as a perfectly round sphere, which is suitable for everyday purposes and shorter distances. The Vincenty formula is a bit more complicated and accurate over long distances as it takes into account that the Earth is not a perfect sphere but an ellipsoid (M. Chalela, E. Sillero, L. Pereyra , 2021).

3.3 Online code

Online, there is numerous code available for routing optimization. We elaborate on 4 different codes in this section. We selected these four codes because they differ in both methodologies and problem types, while all being implemented in Python.

The first code is a code that uses a heuristic: the Nearest Neighbor algorithm. The code uses Excel sheets as input, which provides a clear and structured way to organize the input data. The code addresses the classic Capacitated Vehicle Routing Problem (CVRP), which involves a single depot, multiple vehicles, and customers with specific demands. The objective function is to minimize travel distance (Firdaus, 2023).

The next code utilizes an exact method. The code uses integer programming with Python as the programming language. The problem type is a classic Capacitated Vehicle Routing Problem (CVRP) and the objective is to minimize total travel distance. The code includes a part that generates a visualization of the route using Google Maps. The code uses random coordinates for the input of nodes (Kim, 2020).

The third code is a heuristic. The code solves the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) using Python. The input includes specified demand, time windows, vehicle capacities, and distance/time matrices. The method used is the Path Cheapest Arc (PCA). The objective function aims to minimize the travel time and the output shows the time, distance and load per route. Excel is used as an input (Singh, 2022).

The fourth code is an exact approach. In this code for the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW), Mixed Integer Programming is utilized, while using Python. The objective function minimizes the time (Cimren, 2019).

3.4 Chapter conclusion

To conclude this chapter, the research questions belonging to this part of the research methodology, are answered. The primary research question for this chapter was "How can routing problems be solved, and what available methods and tools can be utilized?". We answer the sub questions for this answer.

What are the type of optimization problems and key methodologies used in routing problems?

Types of routing optimization problems

the primary types of problems used for solving routing challenges are the Traveling Salesman Problem (TSP) and the Vehicle Routing Problem (VRP). A TSP allows for a more similar way of planning in the current situation of Koskamp by just determining whether two areas can be covered by 1 vehicle instead of 2 within 90 minutes. A VRP will give a more accurate solution as it takes into account more aspects like multiple vehicles, capacity and time windows. In Chapter 4, we elaborate on the type of optimizable problem.

Key methodologies

The text discusses different approaches to solving the Vehicle Routing Problem (VRP): We outline exact methods such as Mixed Integer Linear Programming (MILP) and Dynamic Programming (DP). Heuristic methods include Cluster First, Route Second (CFRS) and Route First, Cluster Second (RFCS). Other heuristic algorithms are Clarke and Wright Savings Heuristic, Nearest Insertion Heuristic and Path Cheapest Arc (PCA). Additionally, metaheuristics such as Tabu Search and Genetic Algorithms are frequently used to solve routing problems. In Chapter 4, we elaborate on the methodology we choose for this research.

What programming code is already available for routing problems?

There is more code available online for one method than for the other. The available code focuses on minimizing distance or time, and not minimizing the distance while taking into account the time. The chapter outlines 4 options: the first option utilizes the Nearest Neighbor algorithm, a heuristic method for the classic Capacitated Vehicle Routing Problem (CVRP). This code minimizes travel distance. The second option is an exact method that uses integer programming to address the CVRP. It aims to minimize total travel distance. The third option outlines another heuristic approach, using the Path Cheapest Arc (PCA) algorithm to solve the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW). This code minimizes travel time. The fourth option is an exact approach employing Mixed Integer Programming to address the CVRPTW, focusing on minimizing travel time. Chapter 4 outlines the code we choose to use for this research.

4 Model design

This chapter answers the research question: "What is the best model for Koskamp's situation?" The chapter follows this structure: Section 4.1 outlines the decision making that we do for choosing the right model for the needs of Koskamp. Section 4.2 reviews the available code that we found in Chapter 3 and the slight changes that are necessary to align with the requirements of Koskamp. Section 4.3 introduces the mathematical model used, outlining its main components. Section 4.4 concludes by answering the research question for this chapter.

4.1 Choosing the right model

Koskamp indicates that they are willing to completely change the way their delivery system is organized with a better alternative. Koskamp prefers to search for the best routes possible, even if it means significantly changing their current routing system. That is why it is important not to allocate all addresses to a specific area in advance.

We want to put addresses from multiple areas together in one route, so we use a VRP. As the number of addresses can be very high, it is too complex to use an exact method. The routes are generated for each time slot. This needs to be done just before the vehicles will leave the depot, because customers can still place an order for a specific time slot, right before that time slot starts. An exact method will take too long to finish in time before the drivers have to leave.

We use a heuristic to improve the routing situation. The heuristic we apply is a heuristic called Path Cheapest Arc (PCA), that utilizes a CVRPTW model, that we will elaborate on in Chapter 4.3. The PCA approach focuses on selecting 'the least expensive node', which in this case is the shortest time from the current node to the next node, which provides a good solution very quick. It is often used for simulating real-life scenarios efficiently (Dihin Muriyatmoko, 2023). PCA is also a heuristic that has a large selection of available code online in comparison to the other heuristics.

The model selected is the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW), utilizing the Path Cheapest Arc (PCA) heuristic.

4.2 Code Selection

We found several existing codes for the situation of Koskamp (see Section 3.3). We make the choice to use the code utilizing the Path Cheapest Arc method. The reason is that this code is a heuristic and needs less adjustment than the other codes. The two exact methods do not suit for the speed that is needed to calculate the routes and the other heuristic does not handle time windows, which is crucial for Koskamp, so the Path Cheapest Arc is the most suitable, as it closely matches real life situations very well. The objective function in this code minimizes the travel time, which is of importance for Koskamp. We use a code with an objective function that aims for time minimizing and not distance minimizing because this is the most important factor in Koskamp's strategy. A limitation coming from this is that sometimes shorter routes in terms of time, not always correspond to the shortest travel distance. Deliveries must be done within a specific time. After that, we adjust the code to handle the preferred input and output formats.

4.3 Mathematical model

We use the CVRPTW (Capacitated Vehicle Routing Problem with Time Windows) model for this research. The CVRPTW model best fits Koskamp's situation compared to the other models. The main features include that the vehicles have limited capacities, it uses strict time windows and utilizes one single depot. The model seeks for a proper solution while using the Path Cheapest Arc, which we explain in Section 3.2.2 (Dihin Muriyatmoko, 2023).

4.3.1 Assumptions

Assumptions are important for the model because they simplify the complexity of routing operations. Without assumptions, planning becomes very complicated and difficult to manage in an effective way. By making reasonable assumptions about vehicle capacities and constraints such as road conditions, we can simplify operations and make valid decisions. We outline the assumptions in this section.

Capacity of vehicles

The capacity of the vehicles are not all the same, just like the products that are delivered, as they vary in size. That is why it is hard to say how much can fit in a vehicle. With the sizes of the vehicles, we make assumptions on how much one vehicle can fit. We choose to base our assumptions on the capacity on the number of products. We use number over size as it is more easy to quantify the demand for the experiments this way. Also, Koskamp indicates that there is no system yet that classifies the products into groups with similar volume. If Koskamp wants to use size in future use, this is easy to adjust in the input. To make an accurate assumption, we conduct an interview with one of the regional managers of Koskamp (See appendix A.2).

The vehicles are categorized in 3 categories: small, medium and large. Table 4-1 displays which vehicle belongs to which category. At the location in Den Ham, they use small and medium size only. Typically, there is no distinction between the vehicles in the medium category, and this also goes for the small vehicles. There are minor differences. For example, the Fiat Doblo's are somewhat more square-formed and might be able to carry an exhaust, while other small vehicles cannot. However, according to the regional manager, this is negligible.

According to the regional manager, the weight is negligible as well. It does not happen very often that the load is too heavy.

Since the sizes of products vary a lot, the choice of an accurate number that will represent the vehicles capacity was challenging.

Model	Size	# at Den Ham
Renault Kangoo	Bestelbus (small)	2
Renault Trafic	Bus (medium)	3
Opel Combo	Bestelbus (small)	2
Citroen Berlingo	Bestelbus (small)	1
Renault Express	Bestelbus (small)	5
Fiat Doblo	Bestelbus (small)	1
Citroen E-Berlingo Van	Bestelbus (small)	1
Peugeot Partner	Bestelbus (small)	1
Renault Kangoo van e-tech electric	Bestelbus (small)	2

Table 4-1: Vehicle Size Categories (green row indicates an electrical vehicle)

In the future, the company plans to use containers of a certain size (the containers are approximately 20 by 20 by 30 cm). Some locations already use these containers. These containers are designed to transport items such as filters, bulbs, etc. We estimate that each container can hold approximately 3 objects of average size.

* A small vehicle is able to contain approximately 50 containers
* A medium vehicle is able to fit approximately 100 containers.
* A large vehicle is able to fit approximately 150 containers.

We base the insights on the experience of the regional manager. It is hard to be completely exact about this.

We assume that approximately 3 products fit in the container. Of course, a tire would not be able to fit. That is why we choose a margin of 1 to compensate the fact that a tire needs more space. For the capacity assumptions we assume that each container can contain 2 products, as huge products, like tires, will not fit.

For the small, medium and large vehicles that would be 100, 200 and 300 products respectively.

Condition of vehicles and roads
When trying to get to a solution that leads to specific routes, the condition of the vehicle is good. The roads are also in good condition. In practice, these aspects could lead to some delay.

Other traffic
Also, traffic jams are not taken into account. Traffic jams are a direct cause of probable delays, as well as traffic lights.

Returns
In real scenarios, Koskamp picks up products at the customers that customers want to return. This is done whenever a new product is delivered. In the model, we assume that the number of returned products in a route, is always less than the number of products that are delivered during that route. This way we neglect the fact that returns have impact on the capacity of the vehicle.

Service time
We assume that the service time per customer is the same and we take an average of 3 minutes for each customer. This number is determined by Koskamp.

4.3.2 Model Components

This section outlines the main components of the mathematical model that optimizes routing for Koskamp B.V. The model includes sets, parameters, decision variables, an objective function, and constraints (Sara Rodrigo, Dilina Kosgoda, W. Madushan Fernando, Peter Nielsen, Amila Thibbotuwawa, 2024). The sets represent the nodes, the depot and vehicles involved in the delivery system. Parameters are travel times, vehicle capacities, and time windows influence the routing decisions. Decision variables indicate whether a vehicle travels between specific nodes. The objective function aims to minimize the total time. Multiple constraints ensure that vehicle capacities are not exceeded, routes are structured in a logical way, and deliveries are done within indicated time windows.

Table 4-2 displays the sets, parameters and variables.

Sets
$V = Set\ of\ nodes, \quad 0\ in\ V$
$K = Set\ of\ vehicles$
Parameters
$t_i^k = The\ time\ the\ kth\ vehicle\ visits\ node\ i$
$t_{ij} = The\ travel\ time\ between\ nodes\ i\ \&\ j$
$q_j = The\ total\ quantity\ delivered\ to\ node\ j$
$Q^k = The\ capacity\ of\ vehicle\ k$
$ST_i = The\ average\ service\ time\ at\ node\ i$
$T = The\ maximum\ allowed\ time\ for\ a\ vehicle\ to\ complete\ a\ route\ within\ a\ time\ slot$
$e_i = The\ earliest\ time\ to\ start\ service\ at\ a\ customer$
$l_i = The\ latest\ time\ to\ start\ service\ at\ a\ customer$
Variables
$x_{ij}^k = A\ binary\ that\ takes\ a\ value\ of\ 1\ if\ \ vehicle\ k\ travels\ from\ i\ to\ j, and\ 0\ otherwise$

Table 4-2: Sets, Parameters and Variables

Constraint 1 states the objective function and Constraint 2 until 9 state multiple constraints that we use in the model.

$$\min \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} x_{ij}^k\, t_{ij}$$

Constraint 1: Objective function: minimizes the total time for completing the distribution process.

$$\sum_{i \in V} \sum_{k \in K} x_{ij}^k = 1 \qquad j \in V$$

Constraint 2: Ensures that each node is visited exactly once.

$$\sum_{j=1}^{n} x_{0j}^k = \sum_{i=1}^{n} x_{i0}^k = 1 \qquad k \in K$$

Constraint 3: Ensures that each vehicle starts and ends at the depot (depot = node 0).

$$\sum_{i \in V} x_{ij}^k = \sum_{i \in V} x_{ji}^k \qquad j \in V,\ k \in K$$

Constraint 4: Ensures that each vehicle entering a node, also leaves the node.

$$\sum_{i \in V} \sum_{j \in V} x_{ij}^k * q_j \le Q^k \qquad k \in K$$

Constraint 5: Ensures that the quantity transported by each truck does not exceed its capacity.

$$t_i^k + t_{ij} + ST_i \le t_j^k + M(1 - x_{ij}^k) \qquad i,j \in V,\ k \in K$$

Constraint 6: Ensures that the service time for customer j is starting after the service time for customer i, if customer j is visited immediately after customer i by the same vehicle.

.

$$e_i \leq t_i^k \leq \min\left(l_{i,} T - ST_i - t_{i0}\right) \qquad i \in V,\ k \in K$$

Constraint 7: Requires that kth vehicle arrival times at each node fall within specified time windows and does not exceed available time limit after taking into account service time and travel time.

$$t_i^k \geq 0 \qquad i \in V$$

Constraint 8: Ensures that the service start time at each node is non-negative.

$$x_{ij}^k \in \{0,1\} \qquad i,j \in V$$

Constraint 9: Ensures that the route decision variables are binary variables.

4.4 Chapter conclusion

The research question for this chapter is: “What is an appropriate model to use for the situation of Koskamp?” Chapter 4 outlines that an appropriate model for solving the VRP is Capacitated Vehicle Routing Problem with Time Windows (CVRPTW), which uses the Path Cheapest Arc in its algorithm. This model includes key components such as constraints on limited vehicle capacities and strict time windows, which are essential for meeting the delivery requirements of Koskamp. The use of the Path Cheapest Arc (PCA) heuristic is appropriate to this scenario, as it generates routes quickly, Koskamp needs as the routes sometimes need to be generated around the same time the vehicles need to leave.

UNIVERSITY OF TWENTE.

Koskamp

5 Experimental phase

In this chapter, we look at how well the route optimization model works. Section 5.1 shows what input needs to be put in the model. Section 5.2 shows the form of the output that comes from the model. Section 5.3 outlines the experimental design and in Section 5.4 we explain the experiments in more detail. Section 5.5 outlines how the model improves route efficiency and reduces costs. Section 5.6 compares the new value of the indicators to the old one. In Section 5.7 we validate the model. Section 5.8 answers the research question “How significant is the improvement compared to the old system based on performance indicators?”.

5.1 Input

The model optimizes delivery routes based on multiple input factors. The first input data includes a list of coordinates for addresses that ordered products. Each coordinate corresponds to a node (See Figure 5-1).

Each node has restrictions, including demand (number of products that specific customer has ordered), time windows (in what time window the delivery needs to be done) and a service time of 3 minutes per address for all customers (see Figure 5-2). When we conduct the experiments we indicate that every address has to be visited within 90 minutes, treating all customers the same without any special distinction.

node	Latitude	Longitude
0	52.[illegible]	6.[illegible]
1	52.[illegible]	6.[illegible]
2	[illegible]	7.0[illegible]
3	52.[illegible]	7.0[illegible]
4	52.[illegible]	7.0[illegible]
5	52.[illegible]	6.8[illegible]
6	52.[illegible]	6.[illegible]
7	52.[illegible]	6.8[illegible]
8	[illegible]	[illegible]

Figure 5-1: Input coordinates

Node	Time_Window_Start	Time_Window_End	Demand	Service Time
0	0	90	0	3
1	0	90	1	
2	0	90	1	
3	0	90	1	
4	0	90	1	
5	0	90	1	
6	0	90	1	
7	0	90	1	
8	0	90	1	
9	0	90	1	
10	0	90	1	
11	0	90	1	

Figure 5-2: Time window, demand and service time input

The third input covers the vehicles in the model. We indicate the capacity per vehicle and the average speed of the vehicles in the simulation can be adjusted here as well (See Figure 5-3).

Vehicle	Capacity	# Vehicles	Speed
0	100	19	50
1	100		
2	100		
3	100		
4	100		
5	100		
6	100		
7	100		
8	100		
9	100		
10	100		
11	100		
12	100		
13	100		
14	100		
15	100		
16	200		
17	200		
18	200		

Figure 5-3: Vehicle capacity and average speed input

5.1.1 Time and distance matrix

After we put in the coordinates, the distance matrix is approximated using the Haversine formula based on the coordinates provided (see Section 3.2.4). We decide to use the Haversine formula since it is more usable over the Vincenty Formula in approximating shorter distances. We adjust the matrix by a correction factor to get a better reflection of real-world distances. We conclude that this correction factor is 1.32. We determine this by comparing a Haversine-created matrix and a matrix with actual distances for the same coordinates. We take the average of both matrices and conclude that the actual distances are on average 1.32 times bigger than the distances the Haversine formula determines. The time matrix generates from the specified average speed in the input, which is set to 50 km/h.

5.2 Output

	A	B	C	D	E	F	H	I
1	Sheet Index	# Addresses	Average assumed demand per address	# Vehicles used	Total Time (Min)	Total Distance (KM)	Small	Medium
2	17	14	1	3	247	301.677477	3	0

Figure 5-4: Output for a list of coordinates for area 7, 9 and 10

The output provides a summary of the number of addresses, the number of vehicles used in the experiment, including the total time required for their return to the depot. It also shows the total distance of all the routes combined. Additionally, it displays the quantities of small and medium vehicles utilized. Figure 5-4 shows the output data of a time slot with 14 addresses.

For more detailed output there is a list of what the time and distance is between the specific nodes. As well as the fuel costs and emitted CO_2 per route. Figure 5-5 and 5-6 show the output in the correct form.

	A	B	C	D	E	F
1	Vehicle	Nodes	Distance (KM)	Time (Min)	Cost	CO2
2	1	0	0	0	0	0
3	2	0	0	0	0	0
4	3	0	0	0	0	0
5	4	0	0	0	0	0
6	5	0	0	0	0	0
7	6	0	0	0	0	0
8	7	0	0	0	0	0
9	8	0	0	0	0	0
10	9	0	0	0	0	0
11	10	0	0	0	0	0
12	11	0	0	0	0	0
13	12	0	0	0	0	0
14	13	0	0	0	0	0
15	16	0	0	0	0	0
16	17	0	0	0	0	0
17	14	0 -> 2 -> 4 -> 3 -> 7 -> 12 -> 9 -> 10	89.56862157	85	1.558494015	0
18	0	0 -> 8 -> 11 -> 6 -> 5	102.5813615	77	1.784915691	0
19	15	0 -> 14 -> 13 -> 1	109.5274939	85	1.905778394	0

Figure 5-5: Detailed output of the routes

	A	B	C	D	E	F	G
1	Vehicle	From Node	From Node Coordinate	To Node	To Node Coordinate	Distance (KM)	Time (Min)
2	0	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
3	1	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
4	2	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
5	3	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
6	4	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
7	5	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
8	6	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
9	7	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
10	8	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
11	9	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
12	10	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
13	11	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
14	12	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
15	13	0	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	0	0
16	14	0	52[illegible], 6.[illegible]	4	52[illegible], 6.[illegible]	0.723241976	4
17	14	4	52[illegible], 6.[illegible]	26	52[illegible], 6.[illegible]	16.18904152	23
18	14	26	52[illegible], 6.[illegible]	25	52[illegible], 6.[illegible]	1.183243248	5
19	14	25	52[illegible], 6.[illegible]	1	52[illegible], 6.[illegible]	7.118757812	12
20	14	1	52[illegible], 6.[illegible]	22	52[illegible], 6.[illegible]	11.04919351	17
21	14	22	52[illegible], 6.[illegible]	24	52[illegible], 6.[illegible]	8.499218531	14
22	14	24	52[illegible], 6.[illegible]	23	52[illegible], 6.[illegible]	1.584754447	5
23	14	23	52[illegible], 6.[illegible]	0	52[illegible], 6.[illegible]	7.165307295	9
24	15	0	52[illegible], 6.[illegible]	19	52[illegible], 6.[illegible]	11.46666041	17
25	15	19	52[illegible], 6.[illegible]	13	52[illegible]	15.24645583	22
26	15	13	52[illegible]	14	52[illegible], 6.[illegible]	0.639729162	4
27	15	14	52[illegible], 6.[illegible]	3	52[illegible]	13.96603594	20
28	15	3	52[illegible]	0	52[illegible], 6.[illegible]	0.914900741	2
29	16	0	52[illegible], 6.[illegible]	2	52[illegible], 6.[illegible]	9.518973663	15
30	16	2	52[illegible], 6.[illegible]	15	52[illegible], 6.[illegible]	0.474195075	4
31	16	15	52[illegible], 6.[illegible]	7	52[illegible], 6.[illegible]	10.70354344	16
32	16	7	52[illegible], 6.[illegible]	6	52[illegible], 6.[illegible]	0.103089112	4
33	16	6	52[illegible], 6.[illegible]	5	52[illegible], 6.[illegible]	3.869601762	8

Figure 5-6: Detailed output of the routes

5.3 Experiment Design

Table 5-1 presents the experimental design and outlines the experiments we conduct in this research.

Experiment	Title	Goal
1	Correction factor and speed testing	This experiment tests if the model works in the preferred way with the correction factor and vehicle speed that we choose. We have to make sure that the output is valid and is comparable to a real-world situation.
2	Input definition	We need to determine how to define the input data. We must decide whether it is more effective to input all coordinates at once or process them in several batches.
3	Varying number of addresses and demand in the model with randomized coordinates.	This experiment provides insights in what happens with different number of addresses and varying demand levels, as we conduct numerous experiments with randomly selected coordinates.
4	Historical data analysis	In this experiment we use historical data to validate if the routes improve when we use the algorithm in the model. We gain insights in change in time, distance and costs.

Table 5-1: Experimental design

5.4 Experiments

We conduct Experiment 1 first. To make sure the model works in the preferred way, we test the algorithm of the model using a small set of coordinates to be able to review the performance. We test the correction factor of 1.32, which we derive from comparing actual distance matrices to those calculated using the Haversine formula, as well as the average speed of 50 km/h.

First, multiple groups of coordinates of real-world data is put into the model. We choose this groups randomly, only to verify if the model works well. Table 5-2 displays this output. Then we compare these results with the route generated by Bing Maps, that we also display in Table 5-2. We also show the difference between the model generated route and the actual route in Table 5-2.

# Addresses	# Vehicles used	Model Generated Route		Actual Route in Bing Maps		Difference	
		Distance (km)	Travel Time (min)	Distance (km)	Travel Time (min)	Distance (km)	Travel Time (min)
6	1	40.7	70	38.2	68	2.5	2
5	2	167.8	120	170	116	-2.2	4
6	1	49.7	80	52	84	-2.3	-4
12	2	166.6	146	167	132	-0.4	14
6	1	35.2	63	36.8	64	-1.6	-1
8	1	45.6	83	46.6	83	-1.0	0

Table 5-2: Comparing the model output to the actual distance and time

From Table 5-2, we see that the correction factor and the average speed have some exceptions in the outcome, but give a very accurate representation of the real-world situation.

After we make sure that the model gives us the preferred output, we need to identify how we define our input for the generation of the routes in Experiment 2. Currently, Koskamp creates routes by dividing addresses into 10 different groups, with each group corresponding to an area. However, we decide to try a more efficient method by reducing the number of groups and increasing the number of addresses in each group. This approach allows more addresses to be grouped together into fewer, larger groups, which will lead to more potential for minimizing travel time, and thereby in most cases leads to shorter distance as well.

At first, we conduct three experiments. The first experiments aims to find which way of grouping the addresses will lead to the shortest distance. In Experiment 2a, addresses from 10 areas are combined into one list of address coordinates. In Experiment 2b we split the addresses into two groups: we put the areas that are in greater distance (area 7, 9 and 10) in separately. In Experiment 2c we split the addresses once more resulting into three groups (areas 1-4 and areas 5-6, 8 and 7, 9-10).

For the first set of experiments, we slightly adjust the model per input of addresses. In Experiment 2a, we us a single list of address coordinates, which means only one model setup is available, as the setup is only able to vary when we use multiple lists of coordinates. In this setup, we set the travel time from the customer to the depot to 0 for trips that exceed 45 minutes. This adjustment is necessary because addresses further than 45 minutes away can not be reached and returned within the required 90 minutes time slot that the model manages. When the travel time is set to 0, and the algorithm uses the corresponding address as the last node within a route, the model neglects this time and this route will represent a route that will not be back at the depot within 90 minutes. The distance is never neglected in any of the experiments.

In Experiment 2b and 2c, the addresses from areas 7, 9, and 10 are put in separately. This allows us to use different set ups for the inputs of the address coordinates. Area's 7, 9 and 10 are the areas where the travel time is likely to exceed 45 minutes. For all the addresses in these areas, we set the travel time to the depot to 0. For addresses in the remaining areas, we use the actual travel times. Our goal is to limit the number of vehicles that cannot return within 90 minutes compared to Experiment 2a.

The experiments use addresses based on the locations of real Koskamp customers, with locations selected randomly according to order history. For example, if Area 1 accounts for 30% of all orders, and Area 2 accounts for 20%, then the ratio of orders from Area 1 to Area 2 is 3:2. This is done with all 10 areas.

In this first set of experiments, we conduct 100 runs for each experiment. The number of addresses vary in each run but we use the same number of addresses for every experiment, meaning that if this was not the case, the outcome is not comparable. Also the demand is the same in every run. This ensures that our results are accurate and comparable, which allows us to analyze the impact of each experiment.

Experiment	Average Distance (km)	Average Time (min)
2a	820.578	856.986
2b	821.529	963.314
2c	845.493	993.557

Table 5-3: Outcome of the average distances and averages travel times

Table 5-3 displays the results of the experiments. At first sight, Experiments 2a and 2b appear to have the best outcomes. However, the average travel time in Experiment 2a is significantly lower than in Experiment 2b, despite the distances being similar. This indicates that Experiment 2a neglects the time required for the last customer to return to the depot more often, suggesting that the number of vehicles unable to return within 90 minutes is higher in Experiment 2a compared to Experiment 2b.

We conclude that the best way of distributing the addresses, is to do it in the same way as we do in Experiment 2b. That way we would limit the number of vehicles that cannot return back to the depot in 90 minutes, and have a relatively low distance compared to situations where we would divide the groups even more often.

In Experiment 3, we use the separation of the address coordinates into two groups. To give a proper insight, we vary the number of addresses and the average number of demand in every run. The addresses are ranging from 10 to 100 in steps of 10. In the real situations, the number of addresses is often between 30 and 90. The average demand per address are 3, 6, 9, 10, 11, 12, 13, 14 and 15. The demands vary a lot and it is hard to really give a proper indication, that is why a relatively low lower bound and relatively high upper bound is chosen. We expect that change in distance and travel time would only occur when the average demand is over 9, so that is why we decide to place numbers closer together at higher level of demands to be able to observe when changes in distance and travel occur. This remains accurate, as for each number of addresses, the same address coordinates apply to every average demand level. As a result, multiple average demands have the same output. Each combination tests 20 times, resulting in 1800 runs.

# Addresses	Average assumed demand per address	# Vehicles used	Average # Vehicles used	Average Total Time (min)	Average Total Distance (km)	# Small	# Medium
100	3,6	14 - 17	15.15	1238.55	1013.472	14 - 17	0 - 0
100	9	14 - 17	15.1	1238.15	1013.183	13 - 17	0 - 1
100	10	14 - 17	15.15	1239.25	1014.235	13 - 17	0 - 1
100	11	14 - 17	15.15	1238.6	1013.867	13 - 17	0 - 2
100	12	14 - 17	15.25	1239.65	1014.442	12 - 17	0 - 3
100	13, 14	14 - 17	15.4	1248.85	1026.206	10 - 14	2 - 5
100	15	15 - 18	16.4	1291.05	1072.9	10 - 16	1 - 5
90	3, 6, 9	14 - 15	14.3	1166.95	975.4549	14 - 15	0 - 0
90	10	14 - 15	14.35	1167.8	976.1346	13 - 15	0 - 1
90	11	14 - 15	14.35	1167.4	975.9255	13 - 15	0 - 1
90	12	14 - 15	14.5	1172	979.8741	11 - 14	0 - 3
90	13, 14	14 - 16	14.9	1187.2	992.7654	11 - 14	1 - 4
90	15	14 - 16	15.2	1206.4	1010.4	9 - 14	2 - 5
80	3,6	12 - 14	12.9	1047.4	880.1404	12 - 14	0 - 0
80	9	12 - 14	12.9	1047.45	880.1716	12 - 14	0 - 0
80	10	12 - 14	12.9	1047.55	880.2792	11 - 14	0 - 1
80	11	12 - 14	12.9	1047.95	880.7092	10 - 13	0 - 2
80	12	12 - 14	12.9	1048.1	880.7884	10 - 13	0 - 2
80	13, 14	12 - 14	13.05	1055.7	887.0129	9 - 13	1 - 4
80	15	12 - 15	13.45	1067.5	898.7814	8 - 12	2 - 6
70	3, 6, 9	11 - 13	12.2	969.8	834.9317	11 - 13	0 - 0
70	10	11 - 13	12.2	969.80	834.93	11 - 13	0 - 1
70	11	11 - 13	12.2	969.2	834.4587	10 - 13	0 - 1
70	12	11 - 13	12.25	970.4	835.4372	10 - 13	0 - 2
70	13	11 - 14	12.45	978.95	842.37	9 - 14	0 - 3
70	15	11 - 14	12.45	982.9	845.8958	7 - 12	1 - 4
60	3, 6, 9, 10	10 - 12	10.85	860.15	747.0697	10 - 12	0 - 0
60	11	10 - 12	10.85	861.05	747.8608	10 - 12	0 - 0
60	12	10 - 12	10.85	859.7	746.5959	9 - 12	0 - 1
60	13, 14	10 - 12	10.85	860.15	746.6506	8 - 12	0 - 3
60	15	10 - 12	11.1	870.8	755.8463	7 - 11	1 - 4
50	3, 6, 9, 10, 11	9 - 11	9.75	768.2	684.2191	9 - 11	0 - 0
50	12	9 - 11	9.75	768.2	684.2191	8 - 11	0 - 1
50	13, 14	9 - 11	9.8	769.9	685.555	8 - 11	0 - 1
50	15	9 - 11	9.9	776.2	690.9163	6 - 10	0 - 3
40	3, 6, 9, 10, 11, 12	7 - 9	8.5	662.9	595.5984	7 - 9	0 - 0
40	13, 14	7 - 9	8.5	662.9	595.5984	6 - 9	0 - 1
40	15	7 - 9	8.55	663.45	596.0328	6 - 9	0 - 2
30	3, 6, 9, 10, 11	6 - 9	7.45	567.35	531.003	6 - 9	0 - 0
30	12	6 - 9	7.45	567.35	531.003	6 - 9	0 - 1
30	13, 14	6 - 9	7.45	567.35	531.003	5 - 9	0 - 1
30	15	6 - 9	7.45	567.7	531.343	4 - 9	0 - 2

20	3, 6, 9, 10, 11, 12, 13, 14	5 - 7	5.8	437.2	411.1973	5 - 7	0 - 0
20	15	5 - 7	5.8	437.2	411.1973	4 - 7	0 - 1
10	3, 6, 9, 10, 11, 12, 13, 14, 15	3 - 5	3.95	271.35	262.4815	3 - 5	0 - 0

Table 5-4: Experiments done separating the areas into two groups

The results in Table 5-4 show that the model, when having 50 or more addresses, requires more than the usual 10 vehicles Koskamp uses per time slot. We expect this outcome as the model takes into account strict time windows. In the old situation, Koskamp often exceeds the time limit of 90 minutes with a higher number of addresses, which allowed them to use less vehicles. From the experiments, we observe that the distance and time shift differently at different demand levels based on the number of addresses. This shift occurs more quickly with a higher number of addresses than with a lower number. After we use randomized data, we conduct a number of experiments with real-world data for Experiment 4, using sets of addresses from historical data. Table 5-5, 5-6, 5-7, 5-8 and 5-9 display the results of using coordinates of an actual group of orders, using the developed model.

# Addresses	**Average assumed demand per address**	**# Vehicles used**	**Total Time (Min)**	**Total Distance (KM)**	**Actual Total Time (Min)**	**Actual Total Distance (KM)**
38	6	8	645	564.2326662	619	567.6
55	6	9	724	617.1910883	739	617.2
38	6	7	543	483.4806912	534	479.1
63	6	10	791	640.0032985	754	634.7
54	6	8	588	378.1014754	606	394.5
40	6	7	598	473.4361786	563	488.1
81	6	13	1008	826.1183902	1025	826.5

Table 5-5: Model output versus actual distance and time in Bing Maps

In Table 5-5, the output of the model appears in yellow. The same order of coordinates are put in Bing Maps, which shows an accurate time and distance of the route (including the assumed average of 3 minutes of service time at every address). This displays in green. It shows that the output of the model and the actual time and distance from Bing Maps are very similar.

# Addresses	Average assumed demand per address	# Vehicles used	Actual Total Time (Min)	Actual Total Distance (KM)	# of routes that exceed time limit	(Old) # Vehicles used	(Old) Total Time	(Old) Total Distance	(Old) # of routes that exceed time limit
38	6	8	619	567.6	1	10 ↑	660 ↑ 6.6%	594.6 ↑4.8%	0
55	6	9	739	617.2	1	10 ↑	748 ↑ 1.2%	650.4 ↑ 5.4%	3
38	6	7	534	479.1	0	10 ↑	645 ↑ 20.8%	569.3 ↑ 18.8%	1
63	6	10	754	634.7	0	9 ↓	753 ↓ -0.13%	612.3 ↓ -3.5%	4
54	6	8	606	394.5	2	9 ↑	654 ↑ 7.9%	563.6 ↑ 42.9%	2
40	6	7	563	488.1	0	9 ↑	606 ↑ 7.6%	551.6 ↑ 13.0%	0
81	6	13	1025	826.5	1	10 ↓	918 ↓ -10.4%	697.1 ↓ -15.7%	5

Table 5-6: Actual distance and time in Bing maps versus Old planned route in Bing Maps

Table 5-6 shows the new routes marked in green and the old routes marked in red, both after the coordinates are put in by Bing Maps. While the new routes offer improvements, some old routes still have shorter distances and durations. However, the table reveals that routes marked in red, despite being shorter, exceed time limits more frequently. This suggests that even though these routes are shorter, they are not as efficient in terms of time compared to the new routes.

# Addresses	# Vehicles used	Actual Total Distance (kg)	Fuel costs (€)	CO_2-emissions (kg)	(Old) # Vehciles used	(Old) Total Distance	(Old) Fuel costs (€)	(Old) CO_2-emissions (kg)
38	8	567.6	34.2535248	42.8538	10 ↑	594.6 ↑4.8%	40.0113475	51.84912
55	9	617.2	39.62753173	50.6104	10 ↑	650.4 ↑ 5.4%	43.7661965	56.71488
38	7	479.1	26.53666457	32.1681429	10 ↑	569.3 ↑ 18.8%	38.3088802	49.64296
63	10	634.7	42.70972464	55.34584	9 ↓	612.3 ↓ -3.5%	39.3129254	82.0482
54	8	394.5	23.807286	29.78475	9 ↑	563.6 ↑ 42.9%	36.1861257	75.5224
40	7	488.1	27.03516171	32.7724286	9 ↑	551.6 ↑ 13.0%	37.1178259	73.9144
81	13	826.5	62.09837815	82.7135769	10 ↓	697.1 ↓ -15.7%	46.9086955	60.78712

Table 5-7: Change in Distance, Fuel costs and CO2-emissions

In Table 5-7, the differences in distance, fuel costs and CO_2-emissions of the old and new routes are displayed.

For the final comparison, we determined the salary and depreciation costs per route for a single day (which is divided into six time slots) for the new situation and the old situation. The first six time slots from the real-world data represent a full day.

Time Slot	# Addresses	# Vehicles used	# Not back in time	# Employees needed	**Salary costs (€)**	**(Old) # Vehicles used**	**(Old) # Not back in time**	**(Old) # Employees working**	**(Old) Salary costs (€)**
08:00	38	8	3	8	226.08	10	3	10	282.60 ↑20.0%
09:30	55	9	4	12	339.12	10	3	14	395.64 ↑7.7%
11:00	38	7	3	11	310.86	10	3	14	395.64 ↑15.4%
12:30	63	10	4	13	367.38	9	3	14	395.64 0.0%
14:00	54	8	2	12	339.12	9	3	12	339.12 0.0%
15:30	40	7	2	8	226.08	9	3	9	254.34 ↑11.1%
Day total:					1808.64				2062.98 ↑14.1%

Table 5-8: Change in salary costs

From the results in Table 5-8 it turns out that the salary costs in the old situation are 14.1% higher for this day.

Time Slot	**# Addresses**	**Total Distance (km)**	**Depreciation costs (€)**	**(Old) Total Distance (km)**	**(Old) depreciation costs (€)**
08:00	38	567.6	41.624	594.6	43.604
09:30	55	617.2	45.261	650.4	47.696
11:00	38	479.1	35.134	569.3	41.749
12:30	63	634.7	46.545	612.3	44.902
14:00	54	394.5	28.930	563.6	41.331
15:30	40	488.1	35.794	551.6	40.451
Day total:			233.29		259.73 ↑11.3%

Table 5-9: Change in depreciation costs

We base the depreciation costs on the distance travelled by a vehicle. From the results in Table 5-9 it turns out that the depreciation costs in the old situation are 11.3% higher for this day. In Appendix B, we explain the calculations of the salary and depreciation costs.

5.4.1 Limitations of the experiments

With real-world data, the model sometimes exceeds the time limit in Bing Maps, but never by more than 5 minutes. To fix this, we can either make the maximum duration of a route shorter in the input or move 1 or 2 addresses manually. By making the maximum duration shorter, the model will generate shorter routes and in the real life situation there will be more time to complete the route within the time limit. This should help solve the problem without overly increasing the travel distances.

Another limitation is the demand. In the experiments, we assume the average demand. This might not be completely accurate to the real situation. The model does not account for returns, which could affect route efficiency. This is an aspect that failed to be incorporated into the code. However, the old generation of the routes also does not take this into account, so the improvements are still accurate.

We conduct more experiments with randomized data than with real-world data due to the amount of time it costs. Although real-world data showed improvements, the limited number of experiments not fully capture all possible scenarios.

We estimate the fuel costs and CO_2-emissions based on the values in Table 2-1 in Section 2.2.2., using the cheapest available vehicles. For instance, when 8 vehicles are utilized, the average costs and CO_2-emissions per kilometer of the 8 cheapest vehicles are cumulatively averaged and then multiplied by the distance traveled. However, this method is not entirely precise when using the model, as the code requires two inputs: during the second input of coordinates from areas 7, 9, and 10, it does not account for the fact that the cheapest vehicles have already been utilized in the first input. In Table 5-6, we calculate manually to still be as precise as possible.

5.5 Insights

The main objective of the model is to reduce route distances, thereby lowering fuel costs and CO_2-emissions. Experiments show that from around 50 addresses, the number of vehicles utilized drop below 10, compared to the 10 vehicles typically used in the old system for 10 routes.

Overall, the experiments indicate that the model's routes become more efficient as the number of addresses decreases, specifically 50 or lower. However, real-world data shows that with a higher number of addresses, between 60 and 70, the routes also improve. Above this number, more vehicles are utilized, due to the strict time constraints. Although further validation is required, the improvements demonstrate with both the randomized and the real-world data suggest that the new routing approach offers a more accurate and efficient solution compared to the previous system.

Future work should focus on refining the model to address its limitations. In Section 6.3 we elaborate on this.

5.6 Comparison to start situation

It is hard to say what the exact improvement is on the routes. From the experiments we conclude that the model is able to improve the routes, but this is not in a consistent way. In the output of the model using real life data, it shows that one output of the total distance in a time slot is around 5% higher in the old situation but another is around 40% higher in the old situation. Also, some values are lower in the old situation, but still improve as less vehicles will exceed the time limit. When comparing the new situation with the old one, we believe that the model is effective in improving the routes.

5.7 Validation and reliability of the model

After we conduct the experiments, we present the routes to an executive of the company. The routes that came from the model are validated and seem to appear feasible in practice, but might need some minor manual adjustment in some cases. For example, when a route is too tight, meaning that it is hard to do all the deliveries within the set time, a small manual adjustment to another route can be made, to provide more flexibility for the tighter route.

5.8 Chapter conclusion

The research question for this chapter was: "How big is the improvement compared to the old situation when looking at the performance indicators?". The main performance indicator is the travel distance and therefore also the fuel costs and CO_2-emissions, as we calculate those based on the distance. Although it is hard to give an exact number, the model does show improvement. It shows that it has to use fewer vehicles and make routes shorter, which helps in reducing fuel costs and emissions. Some routes can improve a lot, while others only imply a small change. For situations where the number of addresses are above 50, the number of vehicles exceeds 10. For fewer than 50, this is not the case. In real-world data, the experiment with 38 addresses shows a 6.6% reduction in time and a 4.8% reduction in distance. Table 5-6 in Section 5.4 shows this for 8 experiments. Salary costs reduce by 14.1% and depreciation costs by 11.3% on a randomly chosen day. It also shows improvement in delivering within the specified time the delivery needs to be done. When the experiments show more vehicles are needed then the usual 10, we see that the time limits do not get exceeded as many times as it does in the old situation. Overall, the new model is better than the old one, but there is a need of testing and refining it to make sure it works as well as possible in real-life situations. Some minor manual adjustments to the generated routes can be helpful. As this might occur sometimes, the generation of the routes is not completely without help of manual adjustments.

6 Conclusions and recommendations

This chapter provides a summary of the findings of the research and offers recommendations based on those findings. It is divided into four sections: Conclusions, Recommendations, Post-Research, and Alternatives. Section 6.1 summarizes the answers of all the research questions. Section 6.2 provides advice for Koskamp to implement the model. Section 6.3 is the Further Research section and this section identifies areas that could use further research and improvement, while Section 6.4 outlines a suggestion considering other routing solutions available in the market.

6.1 Conclusions

The study aims to optimize the current delivery routes by analyzing multiple aspects and to discover how route optimizing methods would enhance that efficiency

The primary question in Chapter 2 is: "What are the key aspects of Koskamp's current operational setup?". Koskamp relies on the MobileNXT-system for the planning of their routes, in which manual adjustments are done when necessary. The vehicles have specific specifications when it comes to size and costs. Drivers identify inefficiencies in the current route planning, as not being able to deliver in time due to long distance or unexpected traffic circumstances.

Chapter 3 focuses on finding appropriate literature to use in the research. We formulate the research question: "How can existing literature on routing problems help Koskamp with their strategic implementation?". We analyze different methods for solving routing problems. We identify the Traveling Salesman Problem and the Vehicle Routing Problem as the most important approaches. We explore different variants of the VRP, including the CVRP, VRPTW and the CVRPTW. The solution approaches varies from exact ways of model optimization like Mixed Integer Linear Programming (MILP) and Dynamic Programming (DP). Those give optimal solutions but take a long time to generate routes, compared to heuristic and metaheuristic methods such as Cluster First, Route Second, Clarke and Wright Savings Heuristic, and Genetic Algorithms, which offer proper solutions that are not always optimal. We conduct additional research on the Haversine formula and eventually use this formula in the model.

In Chapter 4, the primary research question is: "What is an appropriate model to use for the situation of Koskamp?". We decide that The Capacitated Vehicle Routing Problem with Time Windows with the Path Cheapest Arc heuristic is the most appropriate method, since it is important to make sure the code can deliver within certain time windows and takes into account the capacity. Also, heuristics aim to find a solution very quick and since Koskamp needs to have the routes generated just before leaving, the Path Cheapest Arc was very suitable.

For Chapter 5 the primary research question is: "How big is the improvement compared to the old situation when looking at the performance indicators?". In comparison with the old way the routes are generated, the model shows improvement. The models output with randomized addresses shows a usage of less vehicles with a lower number of addresses than 50. However, when the address number is higher than 50, the usage of the number of vehicles is above 10. When we deal with 10, 20, 30, 40 or 50 addresses, the average vehicles used is 5.8, 7.45, 8.5 and 9.75 respectively, with an average demand per address of 6. This is below 10 for each number of addresses. When having 60, 70, 80, 90 or 100 addresses, the vehicles used is on average 10.85, 12.2, 12.9, 14.3 and 15.5 respectively, with an average demand per address of 6. This is above 10 for each number of addresses. We conclude that in these cases, the time restrictions are violated many more times in the old situation. When we look at the experiments we do with real-world data, we see that the total distance (and thereby fuel costs, CO_2-emissions), is 6.6% higher in terms of time and 4.8% higher in terms of distance in the old situation, with 38 addresses. In Table 5-6 in section 5.4 the changes are displayed for 8 different time slots. Although the degree of improvement varies per route, it certainly improves. The salary costs for a specific day (divided into 6 time slots) are 14.1% higher in the old situation. The depreciation costs are 11.3% higher in the old situation for this same day.

After we outlined the current setup of Koskamp and explored different ways of routing methods, we were able to determine which aspects needed improvement and how to improve them. The research then led us to the CVRPTW-model using the Path Cheapest Arc, which is effective in generating proper routes in a fast way while meeting time constraints and vehicle capacities. This model than helped us to optimize routes and thereby reducing the distance traveled or limit the number of vehicles that exceed the time limit.

6.2 Recommendations

While the algorithm of the model developed in this research is not yet perfect in simulating real-life scenarios, it represents an improvement over the current routing methods used by Koskamp. We expect the implementation enhance the efficiency and optimize delivery. The implementation can be done by Koskamp in the described way:

Start by applying the model in the real-world to some of the time slots to keep track on its performance in different conditions. Then, analyze and adjust the algorithm based on real-world feedback to address deviations and improve the accuracy of the model. In different scenarios, such as areas with high order density and different traffic conditions, the results will give a good insight on how the algorithm of the model performs under different circumstances. As the routes might be a bit tighter and therefore more restrictive for the drivers, it is important that the drivers adapt to this way of delivering. It is most important to check the new routes on costs saving and CO_2-emissions, as this is hard to approximate in the experiments we conduct.

After testing the algorithm and analyzing its results in performance, cost savings, and CO_2-emissions, Koskamp could start creating a software for actual usage. It is helpful to develop a software that works in a similar way as the MobileNXT-system, as that will not be a big change for the people working in the logistic department. Setting up a way to get feedback from users will help improving the software (P. Toth, 2014).

6.3 Post-Research

We concluded the primary research, but there are still some issues, such as customers who want to return products and tight delivery times, that are not being handled in the model. These challenges, along with the need for more real-world testing, point to areas for further improvement. Future work could focus on fixing these issues and testing the model in a wider range of situations to be able to realize its full potential.

6.4 Alternatives

There are companies that have many experience in creating routing algorithms. Koskamp might want to look at and compare some of these options. These companies often have the possibility to handle complex problems better, like delivering to specific areas within tight time limits or managing complex delivery needs. They might be able to deal with issues such as delivering to areas 7, 9, and 10 within 90 minutes without needing to return to the depot in that same time. They could also help manage things like capacity and returns more efficiently. By considering these alternatives, Koskamp might find a solution that works better for their needs.

6.4.1 Examples of alternatives

Two examples of these alternatives are outlined here:

RoutePilot.net ensures users to generate the most efficient delivery route with a single click. It uses advanced AI technology. It calculates the routes in the most optimal way by considering factors as distance, capacity and vehicle type. The price of this software is €35 per vehicle per month. As there currently are 18 vehicles at the Den Ham location, Koskamp would have to pay €630 a month for this software. However, this is for this location only. (RoutePilot, sd)

RouteLogic is a software with an application. With the software the best routes are generated with very accurate arrival times. This software also uses AI in generating the routes. It also is able to tell when a vehicle will need gas or needs to be charged (in case of electrical vehicles). The price of this software varies from €39 to €49 per vehicle per month. For the Den Ham location this would cost between €702 and €882 a month (RouteLogic, sd).

Bibliography

Anita Agárdi, L. K. (2022). Mathematical Model for the Generalized VRP Model. *Sustainability (Switzerland)*.

Arild Hoff, I. G. (2009). Lasso solution strategies for the vehicle routing problem with. *European Journal of Operational Research*.

Artur Alves Pessoa, R. S. (2020). A Generic Exact Solver for Vehicle Routing and Related. *Mathematical Programming*.

Cimren, E. (2019, December 15). *Solving Single Depot Capacitated Vehicle Routing Problem Using Column Generation with Python*. Retrieved from Github: https://emrahcimren.github.io/operations%20research/Solving-Single-Depot-Capacitated-Vehicle-Routing-Problem-Using-Column-Generation-with-Python/

Citroen Berlingo. (2024). Retrieved from hedin automotive: https://www.hedinautomotive.nl/citroen/modellen/berlingo?msclkid=1a129747a53a11658fdbf1732f7e4de4

Citroën Berlingo. (2024). Retrieved from Van Mossel: https://www.vanmossel.nl/voorraad/s/1402450-citroen-berlingo

Citroën E-Berlingo. (2024). Retrieved from Hedin Automotivr: https://www.hedinautomotive.nl/citroen/modellen/berlingo?msclkid=a14653bfa18510460e04358b01c43342

Cordeau, J.-F. (2002). A guide to vehicle routing heuristics. *Journal of the Operational Research Society*.

Dihin Muriyatmoko, A. D. (2023). Heuristics and Metaheuristics for Solving Capacitated Vehicle . *Elsevier*, https://doi.org/10.1016/j.procs.2024.03.032.

Dihin Muriyatmoko, A. D. (2023). Heuristics and Metaheuristics for Solving Capacitated Vehicle Routing Problem: An Algorithm Comparison. https://doi.org/10.1016/j.procs.2024.03.032.

El-Sherbeny, N. A. (2010). Vehicle routing with time windows: An overview. *Journal of King Saud University*, https://doi.org/10.1016/j.jksus.2010.03.002.

Fiat Doblo. (2024). Retrieved from Fiat Professional: https://www.fiatprofessional.com/nl/doblo/nieuwe-doblo/motor

Firdaus, A. Z. (2023, July 21). *Solving Vehicle Routing Problems with Python & Heuristics Algorithm*. Retrieved from medium: https://medium.com/@writingforara/solving-vehicle-routing-problems-with-python-heuristics-algorithm-2cc57fe7079c

Geeksforgeeks. (2023, Mei 8). *Inleiding tot Branch and Bound – Tutorial voor datastructuren en algoritmen*. Retrieved from geeksforgeeks: https://www.geeksforgeeks.org/introduction-to-branch-and-bound-data-structures-and-algorithms-tutorial/

Gonzalez, M. E. (2022, July 20). *PyCharm vs Spyder: Best Choice for Python Programming*. Retrieved from Career Karma: https://careerkarma.com/blog/pycharm-vs-spyder-2/

Gurobi. (n.d.). *Mixed Integer Programming Basics*. Retrieved from Gurobi Optimization: https://www.gurobi.com/resources/mixed-integer-programming-mip-a-primer-on-the-basics/

Heerkens, H. &. (2021). *Solving Managerial Problems Systematically*. Retrieved from https://doi.org/10.4324/9781003186038

Hochba, D. S. (1997). *Approximation Algorithms for NP-Hard Problems.*

Ilhan, I. (2020). A population based simulated annealing algorithm for capacitated vehicle routing problem. *Turkish Journal of Electrical Engineering and Computer Sciences*, https://doi.org/10.3906/elk-1902-122.

Johns, R. (2024, January 26). *10 Best Python IDE & Code Editors [2024 Update]*. Retrieved from hackr: https://hackr.io/blog/best-python-ide

Johns, R. (2024, Janaury 26). *10 Best Python IDE & Code Editors [2024 Update]*. Retrieved from hackr: https://hackr.io/blog/best-python-ide

Kianfar, K. (2011). Branch-And-Bound Algorithms. 10.1002/9780470400531.eorms0116.

Kim, K. (2020, April 30). *Capacitated Vehicle Routing Problem (CVRP) with Python+Pulp and Google Maps API*. Retrieved from Medium: https://medium.com/jdsc-tech-blog/capacitated-vehicle-routing-problem-cvrp-with-python-pulp-and-google-maps-api-5a42dbb594c0

Kris Braekers, K. R. (2015). The vehicle routing problem: State of the art classification and review. *Elsevier*.

L Zeng, H. O. (2006). A generalized crossing local search method for.

Lalla-Ruiz, E. (2022). *Lecture 2 - Routing problems: Lecture 2 VRP*. Retrieved from https://canvas.utwente.nl/courses/12324/pages/lecture-2-routing-problems?module_item_id=397807

Lalla-Ruiz, E. (2022). *Sourcing, SCND, Transportation Powerpoint slides*. Retrieved from Canvas.

M. Chalela, E. Sillero, L. Pereyra . (2021). GriSPy: A Python package for fixed-radius nearest neighbors search. *Elsevier*, https://doi.org/10.1016/j.ascom.2020.100443.

MathWorks. (n.d.). *Mixed-Integer Linear Programming (MILP) Algorithms*. Retrieved from MathWorks: https://www.mathworks.com/help/optim/ug/mixed-integer-linear-programming-algorithms.html

Mizux. (2024). *vrp_capacity.py*. Retrieved from Github: https://github.com/google/or-tools/blob/stable/ortools/constraint_solver/samples/vrp_capacity.py

Mizux. (2024). *vrp_time_windows.py*. Retrieved from github: https://github.com/google/or-tools/blob/stable/ortools/constraint_solver/samples/vrp_capacity.py

Niels Wouda, L. L. (2024). PyVRP: A High-Performance VRP Solver Package. *INFORMS*. Retrieved from 10.1287/ijoc.2023.0055

Okeke, F. (2023, july 21). *5 programming languages database administrators should learn*. Retrieved from TechRepublic: https://www.techrepublic.com/article/5-best-programming-languages-for-database-administrators-to-learn/

OPEL COMBO. (2024). Retrieved from Hedin Automotive: https://www.hedinautomotive.nl/opel/modellen/combo?msclkid=58a1c56a5b081937de8ba35edc8ea9d5

OR-ools. (2021). *Capacity Constraints*. Retrieved from Developers: https://developers.google.com/optimization/routing/cvrp

P. Toth, D. V. (2014). *Vehicle Routing: Problems, Methods, and Applications.*

Peugot. (2024). Retrieved from Peugot: https://www.peugeot.nl/showroom/peugeot-partner.html

Ramser, D. a. (1959).

Renault Express. (2024). Retrieved from Van Mossel: https://www.vanmossel.nl/renault/bedrijfswagens/express/?hic_camp=microsoft_ads:531333059:1349103297811601:84319360280266&utm_device=c&utm_term=www.vanmossel.nl&utm_source=microsoft&utm_medium=search_paid&utm_campaign=Van%20Mossel%20Performance%20Max%20Sa

Renault Express. (2024). Retrieved from Autoweek: https://www.autoweek.nl/autonieuws/artikel/in-detail-renault-express/?referrer=https%3A%2F%2Fwww.bing.com%2F

Renault Trafic. (2024). Retrieved from Autoweek: https://www.autoweek.nl/auto/105645/renault-trafic-l1h1-blue-dci-110-generique/

RouteLogic. (n.d.). *RouteLogic*. Retrieved from https://www.routelogic.nl/

RoutePilot. (n.d.). Retrieved from RoutePilot: https://routepilot.net/

Sara Rodrigo, Dilina Kosgoda, W. Madushan Fernando, Peter Nielsen, Amila Thibbotuwawa. (2024). OPTIMIZING FRESH-CUT FLOWER DISTRIBUTION USING THE . *Scientific Journal of Logistics*, https://www.logforum.net/volume20/issue1/abstract-3.html.

Shuhan Kou, B. G. (2023). An improved model for estimating optimal VRP solution values. *Optimization Letters*.

Singh, S. (2022, Janaury 15). *Solving CVRPTW using OR Tools in Python*. Retrieved from Medium: https://satpal-singh.medium.com/solving-cvrptw-using-or-tools-in-python-214247f6d680

Sorensen, K. (2013). A History of Metaheuristics. *Research gate*.

Suresh Nanda Kumar, Ramasamy Panneerselvam. (2012). A Survey on the Vehicle Routing Problem and Its Variants. *Scientific Research*, http://dx.doi.org/10.4236/iim.2012.43010.

Tobbileh, Y. (2021, November 7). *Dynamic Programming Techniques with Examples*. Retrieved from yasir-tobbileh.medium: https://yasir-tobbileh.medium.com/dynamic-programming-techniques-with-examples-89b684f406ec

Urquhart, N. (2022). *Nature Inspired Optimisation.* https://doi-org.ezproxy2.utwente.nl/10.1007/978-3-030-98108-2_1.

Uzayr, S. b. (2021). *Optimizing Visual .* Open Access.

Velimirovi, L. Z. (2023). Determining the Optimal Route of Electric Vehicle. *IEEE TRANSACTIONS ON FUZZY SYSTEMS*.

Vitória Pureza, R. M. (2011). Vehicle routing with multiple deliverymen: Modeling and heuristic approaches. *Elsevier*.

Appendices

Appendix A: Interviews

Appendix A.1: Interview held with an experienced driver at Koskamp

What are the common challenges faced with specific routes? The routes going to areas 2 and 3 are challenging due to the demand being often high and tight scheduling. Area 3 contains addresses in Raalte and Heino and is hard to reach when you have to be back in time, as they are far away. The same applies to area 2 in Hardenberg. **So you would say that area 2 and 3 are the most challenging?** Yes. **Do you prefer driving to the same area every time you work or to drive to different areas?** I would prefer to drive to the same area as the contact you have with customers is better. The customer relationships will improve because of this. **Would you not like variation in the areas you drive to?** No, because even if you drive to the same areas every day, no day is the same. **What are aspects of the current delivery system that could use improvement?** The way the routes are generated. Sometimes last minute a customer gets added and this one is not put into the scanner. When you follow the scanner, sometimes it leads you crisscross instead of circular. And sometimes the scanner thinks you can be at a place within a certain time, but it is not even possible due to traffic. Especially not when you leave 5 minutes later than planned. I would prefer if for more customers there will be less time slots, as I sometimes drive for one small product.

Appendix A.2: Interview held with one of the regional managers about the capacity of the vehicles

How do you determine the capacity of each vehicle type (Small, Medium, Large, XL) in terms of fitting auto parts, such as tires and containers? We aim to use a system that requires boxes of 20x30x20 cm that are put in the vehicle. The small vehicle is approximately 4 cubic meters and will fit approximately 50 boxes, the medium is approximately 6 cubic meters and will fit approximately 100 boxes and the large vehicle is about 8 cubic meters and will fit approximately 150 boxes. **And how many products will fit in a box based on number?** 3 average sized products will fit in a box. **Is there any distinction within size within the same size category?** There is no distinction between the vehicles within the same type size. **How do you keep into account larger products like vehicles when the capacity is based on number and not size?** This is hard to say as all the products vary a lot in size. If I would have to make a guess, I think you could say that a tire takes the space of 3 boxes. But it is very difficult to be precise in this. **Do you ever encounter situations where the weight is to heavy?** No, this is negligible. **How does Koskamp group addresses into an area?** This is done automatically by a system that scans the postcode and this postcode is already linked to an area, so the address will be allocated to the corresponding area.

Appendix B: Calculations

Appendix B.1: Calculations of salary costs

To calculate salary costs, we use the gross hourly salary for drivers, which is €18.84. To calculate for each time slot, this salary is multiplied by 1.5 as the duration of the time slot is 90 minutes. For the new situation, we multiply this by the number of drivers needed for each time slot. In the previous situation, we use the number of employees who were working during that time slot. Table 5-7 displays the change in salary costs, comparing the new situation (in green) to the old situation (in red).

UNIVERSITY OF TWENTE.

Koskamp

Appendix B.2: Calculations of depreciation costs

We calculate depreciation costs per time slot based on the distance traveled by the vehicles in that timeslot. We calculate per vehicle, as small and medium vehicles have different catalog values, and we need this value to determine the depreciation costs. For this comparison, we only consider small vehicles, as in this situation only small vehicles are utilized. We consider the traveled distance of a vehicle. We determine the costs by the vehicle's catalog value: €22.000 for small vehicles and €26.000 for medium vehicles, and how much of that value is used up in the route based on the traveled distance. Each vehicle has an expected lifespan of 300.000 kilometers. For a small vehicle, we use the proportion of the traveled distance in relation to this lifespan to calculate the corresponding depreciation cost from the €22.000 catalog value. For example, if a vehicle travels 30 kilometers in a route, this represents (30/300.000) = 0.01% of the vehicles total life span. Therefore we state the depreciation costs for that route to be 0.01% of €22.000, which equals €2.20. Table 5-9 in Section 5.4 displays the change in depreciation costs, comparing the new situation (in green) to the old situation (in red). (The percentage change per time slot is the same as displayed in Table 5-6 and 5-7 in Section 5.4, as the costs are based on distance).