Modeling and analyzing the use of electric vehicles for Van Schoot Pompcentrum BVs routing logistics

University of Twente

Master Thesis IEM

Author: Jasper van den Brink

First supervisor: Dr. Ipek Seyran Topan

Second supervisor: Dr. Ir. Eduardo Lalla-Ruiz

Company supervisor: Jasper Blanke

Date: 29 August 2023

UNIVERSITY OF TWENTE.

VAN SCHOOT
MAAKT KLEDING PERSOONLIJK

Preface

This thesis is the end of my master's Industrial Engineering and Management. After studying at the University of Twente for 7 years, I am happy that this document is the end of my studying career. I am very glad that I could perform this research at Van Schoot Pompcentrum BV. From the first time I met with them, I had a really good feeling about the assignment. Therefore, I would like to thank my company supervisor Jasper Blanke for helping me throughout the whole process. I would also like to thank, Andrea van Schoot for giving me the opportunity of doing my master thesis at her company.

Furthermore, I would like to thank Dr. Ipek Seyran Topan for being my first supervisor. After being my second supervisor with my bachelor assignment, I wanted to ask her as my first supervisor for the master assignment as I enjoyed working with her. With this assignment, I also really enjoyed working together. She helped me with getting some structure into my way of working. Next to that, I would also like to thank my second supervisor Dr. Ir. Eduardo Lalla-Ruiz for his feedback. I think he can give really good feedback and he is interested in the topic of this thesis, which helped me get a better result and kept me motivated to work on this assignment.

I am really happy with the whole process of making this thesis and with the end result. I hope you enjoy reading my master's thesis.

Jasper van den Brink

Management summary

Van Schoot Pompcentrum BV (VSP) is a company that handles all types of services related to clothes. These services are for example embroidery, tailoring, cleaning, and printing of clothes. The logistic service of delivery and pick up of the clothes within a couple of days is the main component of their success. VSP is facing a problem related to this logistic service from 2025 onward. The problem is that from 2025 diesel vehicles are not allowed in many city centers anymore. This forces them to use Electric Vehicles (EVs) from that moment. Another regulation that is changing for business-related vehicle leasing, is that from 2025 all business lease vehicles need to be electric vehicles. VSP wants to get more insight into its logistic routing and optimize the current situation. Therefore, the goal of this research is to gain insight into the current performance of vehicle routing at VSP to be able to create new routes for the introduction of EVs. By doing this VSP is expanding its knowledge of its current performance and next to that, it will then also be able to handle the new regulations that are coming in 2025, while also pursuing sustainability goals. The main research question in this research is the following:

How can VSP introduce electric vehicles to their routing logistics, while also improving the existing routes simultaneously?

Current situation

VSP is dealing with 225 cities or villages where customers are located. Within a city, multiple customers can be present, but for the routing, a city or village is seen as one location. The 24 routes in total are therefore dealing with 225 locations. The locations are divided over the two sets of routes. Tuesday-Friday has 116 locations and Wednesday-Saturday has 109 locations. This division over the days is used to make sure that large customers are on different days, which ensures capacity is not a problem when visiting the locations. Next to the day, the division of the locations over the routes, each location has a handling time and some locations have time windows that need to be considered in the routes. The routes VSP use, give an outline of the cities and villages that need to be visited and one location can have multiple customers. The precise routes driven on these days are dependent on which exact customer placed an order or needs to get clothes returned. These routes for a day are based on the general routes.

Solution approach

To solve the problem VSP is facing, a 2-Phase solution approach is used. In Phase 1, the current situation (Vehicle Routing Problem with Time Windows, VRPTW) will be improved using a VNS/VND approach. These improved routes are then used as input for solving the new situation (Electric Vehicle Routing Problem with Time Windows, EVRPTW) where Phase 2 starts. The main problem with the new situation is where and how long to charge. The new situation is solved by using the improved routes and inserting electric chargers into these routes and determining how long a vehicle needs to charge at such a charging location. Briefly, Phase 1 improves the current situation by using VRPTW techniques and Phase 2 is inserting the charging locations to solve the EVRPTW for VSPs new situation. The insertion of chargers could be done by a heuristic for larger problems and by a solver for smaller problems.

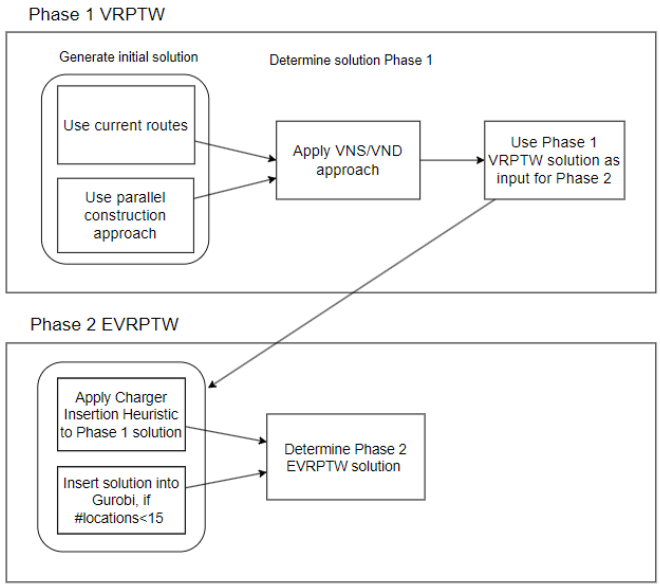

Figure 1. Two-phase solution approach for solving the EVRPTW

Results

Experiments are performed on small size benchmark instances with 5-10-15 locations and large size benchmark instances with 100 locations. Furthermore, real-life instances are randomly generated based on customer data from VSP. Where small size instances with 5-10-15 locations and medium size instances with 40-50-60 locations are created. The approach can find solutions around a 5-15% average difference for smaller instances to the solution found by the Gurobi solver (Table 1) and solutions around a 60% average difference for medium or larger instances to the LP relaxation (Table 1). Which for a relatively simple approach is acceptable. However, a limitation is that the algorithm does perform better in a situation where not all locations have time windows. The problem becomes more difficult when more time locations have time windows. Performance on the real-life instances is relatively better than the performance on the benchmark instances. However, the solution approach is still able to solve the situations with more time windows.

Table 1. Comparison performance on benchmark and real-life instances

	Small size benchmark	**Large size benchmark**	**Small size real-life instances**	**Medium size real-life instances**
Average difference	13.16%	62.11%	4.24%	63.55%
Min difference	0.00%	45.37%	-5.82%	56.93%
Max difference	34.08%	76.22%	13.49%	73.55%

Using the proposed solution approach, VSP can improve the current situation and determine routes including charging locations. For the routes on Tuesday-Friday, the routes for the situation including chargers are even shorter in terms of time compared to the current situation. However, for Wednesday-Saturday, the routes are longer in terms of distances considering the total amount of kilometers. Therefore, more charging is needed, resulting in longer routes when charge times are included (Table 2).

Table 2. Overview improvements VSP

	Current situation (VRPTW)	**New Situation**				
	*Times**	*Phase 1* (VRPTW)*	*Phase 2* (EVRPTW)*	*Distances*	*Charge times*	*Total time****
Tuesday-Friday	6780 minutes	6452 minutes (-328 min)**	6507 minutes (-273 min)**	4550 km	74.5 minutes	6581.5 minutes (-198.5 min)**
Wednesday-Saturday	6963 minutes	6619 minutes (-344 min)**	6779 minutes (-184 min)**	5970 km	231 minutes	7010 minutes (+47 min)**

Times are driving times + handling times, **difference compared to current situation, * times include charge times*

Conclusion and recommendations

The goal of this research is to gain insight into the current performance of vehicle routing at VSP to be able to create new routes for the introduction of EVs. When gaining these insights, VSP is also able to improve its current routing logistics. As the proposed solution method, first improves a generated initial solution or existing current routes, VSP could also improve their current routing, whereafter EVs are introduced into the Phase 1 solution to be able to establish routes for the new situation.

From achieving the goal of this research, the following recommendations can be given to the company. These recommendations are regarding the next steps VSP can take to further investigate how in the coming years their routing logistics can be made electric. The recommendations for VSP are:

1. Investigate the growth of the customers, as this has a lot of influence on the routing and the difficulty of determining good routes.
2. Develop an agile tool that by using the proposed algorithms can help determine routes for changing situations. When the tool can handle all sorts of different situations like adding or changing customers, more flexibility in the routing can be established.
3. Investigate if adding a hub at a more central location can help with reducing tour length for the locations that are most far away. The current location causes large travel distances in some situations, making the optimization of those routes difficult.
4. When using the proposed solution approach, run the heuristic for a long enough time to ensure it escapes local optima. As the performance for larger instances can fluctuate.

Contents

List of figures

List of tables

1 Introduction

In Chapter 1, we describe the company and give a motivation for the research. We state the problem statement and explain the research goal. Furthermore, we identify research questions that are needed to be able to understand the current situation, solve the core problem, and achieve the research goal. In the end, a scope, approach, and timeline of the research are given.

1.1 Company Description

Van Schoot Pompcentrum BV (VSP) is a company that handles all types of services related to clothes. These services are for example embroidery, tailoring, cleaning, and printing of clothes. In the ’70s, the company started small in the eastern parts of the Netherlands. In the meantime, the company and its customer base have grown a lot. VSP is at this moment the largest company in the Netherlands performing these kinds of services in a B2B environment. VSP’s main location is in Oldenzaal, which is a small atelier. Next to that, VSP has a location in Bad Bentheim, where around 30 people work, and a bigger facility in Poland. The location in Bad Bentheim is a depot and office space. A location in Poland is where all clothes are being handled, for example, tailored. Nowadays, they pick up and deliver thousands of clothes per week, where around 75% is tailoring. Tailoring is therefore also their main service. The logistic service of delivery and pick up of the clothes within a couple of days is the main component of their success.

This logistic process is completely in the hands of VSP. This is done to be able to provide the best service times to their customers. They guarantee that the clothes that are picked by them are processed and delivered back to the customer within 3 days. The clothes are picked up and brought to the depot in Bad Bentheim. From there, the clothes are driven to Poland by truck to be tailored. This is done in a big facility where around 200 people work. After tailoring, the clothes are brought back to Bad Bentheim, where the vehicle will distribute the clothes back to the customers.

One of the main goals that VSP has is to provide the best service for its customers. Therefore, they run the logistical process all for themselves. VSP gives the customer the ability to track their order through the track and trace system. VSP wants to provide a good and transparent insight into the logistical processes to their customers.

1.2 Motivation for the research

VSP is facing a routing problem from 2025 onward. The company is currently using diesel vehicles for the pickup and delivery of clothes to and from customers. These vehicles need to go into city centers a lot, as many customers of VSP are located there. The problem is that from 2025 diesel vehicles are not allowed in many city centers anymore (KVK, 2022). This forces them to use Electric Vehicles (EVs) from that moment. Another regulation that is changing for business-related vehicle leasing, is that from 2025 all business lease vehicles need to be electric vehicles (Wingerden, 2022). Therefore, the urgency to change to electric vehicle routing is there.

Next to the obligation, the company also wants to be more sustainable. Making this a good opportunity to combine the new regulation with a company goal. VSP starts by looking into these options before the new rules are applied to establish the new routes before its needed.

VSP is also getting questions from its customers about its sustainability goals. Many of these clothing companies have very extensive sustainability goals and therefore they find it strange when a diesel-powered vehicle picks up these sustainable clothes for one of their services. This research can contribute to the sustainability goals of the customers and VSP themselves. These things together make for enough incentive

to make sure that at least in 2025, but hopefully before that, EVs will be used to pick up and deliver clothes to their customers.

Lastly, VSPs routes are never optimized before. The routes currently used are primarily based on the experience of personnel and drivers. Therefore, information on the performance of their routes is lacking. With this research, VSP is hoping to also get more insight into their logistic routing and optimize the current situation. Introducing new types of vehicles into their routes without having a good insight into their performance is very difficult. Therefore, this type of research can be of help to their new situation when also helping them in their current situation.

In section 1.3, the core problem and knowledge problems are derived and explained. This results in the problem statement of this research.

1.3 Problem Statement and goal

To get to the core problem, first, all problems are stated. After knowing all problems, causes and effects are identified and put into a problem cluster. With the use of the problem cluster, the core problem is selected. After that, the core problem is made measurable.

VSP is facing multiple problems concerning the routing of its vehicles. The first problem is that VSP is forced to switch to EVs from 2025 onward. They lack the knowledge on how they are going to deal with the lower range of the vehicles within their current routes. Another problem is that VSP is not sure how good their routes are performing at the moment. Again, there is a lack of insight into the process. It is hard to implement these new electric vehicles into their route if they are not sure if they currently have good routes.

With both problems, the main component is the lack of insight into the performance of current routing processes. This is then also identified as the core problem (See Figure 2). As there is no insight into the current performance, routes are determined based on experience. This does not have to be a bad thing as in the current situation VSP is making a profit. The downside is that it is unclear if they could make more profit. Also, sometimes routes exceed the 10-hour work limit, which is not preferred by VSP.

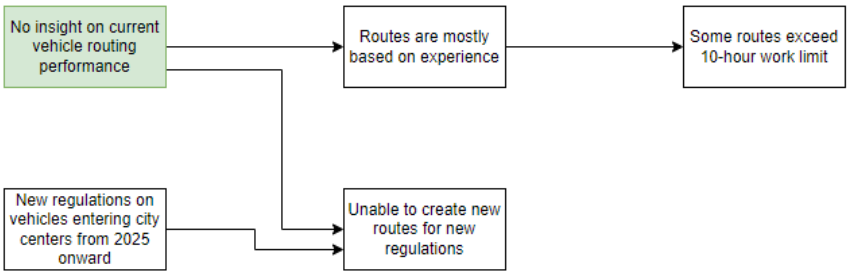

Figure 2. Problem cluster for determining core problem VSP

The core problem is made measurable for the current situation in the following way. Here, the goal is to improve the performance by 5%. This is based on the KPIs used for the objective function. Currently, the routes are done in 13609 minutes (226,8 hours), and the goal for the current routes is to do this within 12929 minutes (215,5 hours). After that, the focus will be on the introduction of EVs. As this situation is new, it is hard to make it measurable, however, we can evaluate the performance compared to the situation with the old vehicles.

The goal of this research is to gain insight into the current performance of vehicle routing at VSP to be able to create new routes for the introduction of EVs. By doing this VSP is expanding its knowledge of its current performance and next to that, it will then also be able to handle the new regulations that are coming in 2025, while also pursuing sustainability goals.

1.4 Research questions

In Section 1.4, research questions are stated concerning various parts of the research. First, the main research question is given, and after that the sub-questions.

1.4.1 Main research question

This research will contain an optimization of the current situation, however, the main goal is to create routes that can be performed by electric vehicles. Therefore, the main research question in this research is the following:

How can VSP introduce electric vehicles to their routing logistics, while also improving the existing routes simultaneously?

The main research question can be answered after completing Chapters 4 and 5. Before we can start solving the problem, the following sub-questions need to be answered to be able to answer the main research question.

1.4.2 Sub questions

Current situation

One of the most important parts of a good foundation for this research is to get a good understanding of the current situation. In Chapter 2, a current system analysis will be performed. Therefore, the following sub-research question is answered:

What does the current routing situation look like and what is the performance of these routes?

Modeling techniques

To be able to improve the current situation and introduce EVs to the routing processes of VSP, an appropriate modeling technique needs to be used. With the use of a literature review in Chapter 3, modeling techniques will be identified and selected to be able to model the problem at hand. The following questions are answered:

How are VRP problems formulated? How are different kinds of constraints modeled in VRP problems?

Possible solution methods

After knowing how VRP problems are modeled, it is important to have a good understanding of how a VRP is optimized. Within Chapter 3, the following research question is also answered about possible solution methods:

What are optimization techniques used when dealing with a VRP problem and how can we deal with VRP problems that have large instances?

Electric Vehicles

The main goal of this research is to be able to determine routes that are performed by EVs. Therefore, the following research questions are answered concerning this topic:

How does the formulation change when EVs are introduced to the VRP problem? What are the optimization techniques when dealing with the EVRP problem?

These questions will also be answered in Chapter 3, as we need to have a good understanding of how we can deal with VSP's problem to be able to answer the main research question.

1.5 Scope

In this research, the focus is on optimizing current routes and creating new routes when introducing EVs to the situation. VSP also mentioned that they were considering small depots in the form of hubs throughout the country to reduce travel times. This would be an extra problem in the form of a facility location problem. That is a new project and out of the scope of this research. This research focuses on the vehicle routing of new electric vehicles.

This research will furthermore focus on the general layout of the routes. This means that the routes determined in this research will have the cities or towns in them and not the exact customer locations as these customers can differ from day to day, however, the cities or towns are almost always the same.

1.6 Limitations

For the current situation, the data that is used is provided by VSP. VSP has a system that tracks every move of every vehicle through all routes. Travel times from the past and present can be evaluated using this system. These times will be made general and not include exceptions about for example days with a lot of traffic jams. These are situations that are excluded when evaluating and creating routes for VSP. The use of data will be more extensively explained in the chapters regarding the modeling of the routes.

1.7 Approach

First, the current situation is modeled and optimized. After that, EVs will be introduced into the model to be able to create new routes including the use of EVs. This will result in the following overview of the chapters, where Chapter 2 is linked to sub-question 1, and Chapter 3 is linked to sub-questions 2, 3, and 4. And the main research question can be answered after Chapters 4 and 5.

- Chapter 1: Introduction
- Chapter 2: Current situation
- Chapter 3: Literature review
- Chapter 4: Solution approach
- Chapter 5: Experiments
- Chapter 6: Conclusion and recommendations

This results in the following overview of steps that are performed during this research related to the given chapters above (see Figure 3):

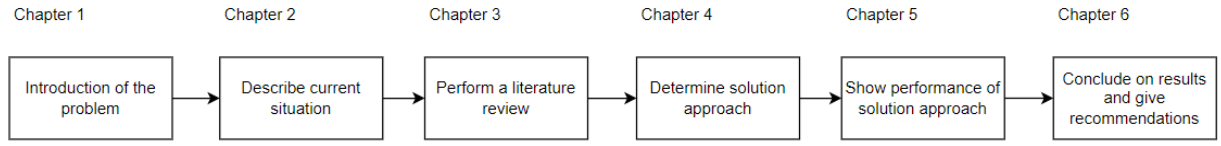

Figure 3. Overview of performed steps

2 Current situation

In Chapter 2, the current situation of the routing logistics of VSP is described and analyzed. Next to the routing process and routes, input parameters for the model are given. Chapter 2 answers sub-research question 1.

2.1 Routing process description

At this moment, VSP has two sets of routes each set having 12 routes. These two sets correspond to two sets of days. These sets are Tuesday-Friday and Wednesday-Saturday. The idea is that when clothes are picked up on Tuesday, the customer will have them back on Friday and vice versa. This is the same for the clothes picked up on Wednesday and Saturday. At the end of the day, the drivers bring the clothes to the depot in Bad Bentheim. Here, the clothes are put in a truck that brings the clothes to the facility in Poland. There, the clothes are handled. After handling, the clothes are brought back to the depot in Bad Bentheim and the drivers bring them back to the customers. This all needs to happen within 3 days because as said, clothes pickup on Tuesday are back at the customer on Friday or if the clothes are picked up on Friday, they are back on Tuesday. This is the same for Wednesday and Saturday.

In total, VSP is dealing with 225 cities or villages where customers are located. Within a city, multiple customers can be present, but for the routing, a city or village is seen as one location. The 24 routes in total are therefore dealing with 225 locations. The locations are divided over the two sets of routes. Therefore, Tuesday-Friday has 116 locations and Wednesday-Saturday has 109 locations. This division is kept in the optimization of the routes as it can give some problems to the customers if they are switched to the other day set. Furthermore, the routing also needs to take into account the amount of work that is sent to the facility in Poland. With the division of locations of the two sets of routes, the division of work is kept desirable. Next to the day the division of the locations over the routes, each location has a handling time and some locations have time windows that need to be considered in the routes. Typically in these types of problems, capacity can also be a constraint to be considered. VSP does not have this problem if the division of the days is kept. For now, the biggest customers are divided over different days and therefore capacity does not play a big part. This will be further explained in Chapter 4, as here the model will be given including all constraints.

The routes VSP use, give an outline of the cities and villages that need to be visited and one location can have multiple customers. The precise routes driven on these days are dependent on which exact customer placed an order or needs to get clothes returned. These routes for a day are based on the general routes displayed and described.

2.2 Current routes

Figure 4 displays all the locations and routes that are considered in the 24 routes. The red triangle is VSPs depot location. As mentioned, these are then divided over 2 sets of days Tuesday-Friday and Wednesday-Saturday.

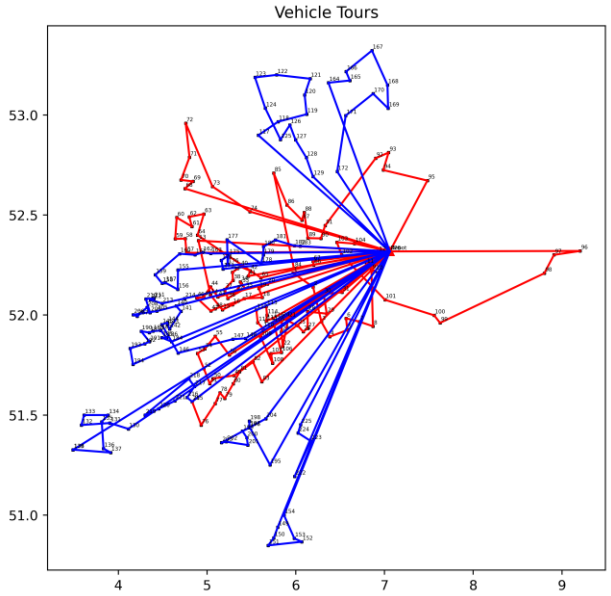

Figure 4. Visualization current routes, red = TueFri, blue=WedSat

Figure 4 gives the current routes. VSP makes a distinction between the two sets of days, where the color red are the routes from Tuesday-Friday and the color blue is the routes from Wednesday-Saturday.

With the locations clear, we show the current routes per day set in more detail. Section 2.4 evaluates the performance of these routes. Table 3 shows the routes for Tuesday-Friday. The sequence given from top to bottom is the sequence that is being driven by the drivers. All routes start and end at the depot in Bad Bentheim.

Table 3. Routes Tuesday-Friday

Route 1	Route 2	Route 3	Route 4	Route 5	Route 6
Apeldoorn	Amsterdam Centrum	Arnhem	Soesterberg (KPU)	Utrecht	Den Bosch
Zutphen	Utrecht Leidsche Rijn	Nijmegen	Amersfoort	Maarssen	Waalwijk
Doetinchem	Houten	Elst	Leusden	Harmelen	Kaatsheuvel
Ulft	Driebergen	Oosterbeek	Zeist	Woerden	Dussen
Aalten	Woudenberg	Duiven	Bilthoven	Ijsselstein	Werkendam
Lichtenvoorde	Lunteren	Zevenaar	Soest	Nieuwegein	Gorinchem
Winterswijk	Barneveld	Didam	Baarn		Leerdam
Sudlohn	Kootwijkerbroek	Doesburg	Bunschoten		Zaltbommel
Enschede		Zelhem	Nijkerk		
Hengelo		Vorden			
Oldenzaal		Borculo			

Route 7	Route 8	Route 9	Route 10	Route 11	Route 12
Schiphol	Alkmaar	Bavel	Apeldoorn Noord	Stadthagen	Heteren
Halfweg	Heerhugowaard	Baarle Nassau	Emmeloord	Minden	Nijmegen Zuid
Haarlem	Bergen	Tilburg	Kampen	Bad Oeynhausen	Wijchen
Beverwijk	Schagen	Udenhout	Hattem	Munster	Grave
Zaandam	Den Helder	Oisterwijk	Zwolle	Munster-Nienberge	Druten
Wormerveer	Hoorn	Vught	Raalte	Ahaus	Opheusden
Purmerend	Lelystad	Rosmalen	Lemelerveld	Rijssen	Rhenen
Amsterdam Noord		Oss	Emmen	Nijverdal	Veenendaal
Voorthuizen		Uden	Emmer-Compascuum	Almelo	Ede
Deventer			Klazienaveen		Bennekom
Schalkhaar					Wageningen
					Renkum

In Table 4, we give an overview of the locations that are in routes for Tuesday-Friday. Here, the sequence from top to bottom is also the sequence that is driven. And again, the routes all start and end at the depot in Bad Bentheim.

Table 4. Routes Wednesday-Saturday

Route 1	Route 2	Route 3	Route 4	Route 5	Route 6
Balk	Krabbendijke	Waddinxveen	Beek	Leimuiden	Leek
Joure	's-Gravenpolder	Gouda	Meerssen	Alphen ad Rijn	Haren
Jubbega	Vlissingen	Capelle ad Ijssel	Maastricht	Leiderdorp	Groningen
Drachten	Middelburg	Rotterdam Alexandrium	Kerkrade	Leiden	Appingedam
Surhuisterveen	Goes	Rotterdam Hesseplaats	Heerlen	Katwijk	Winschoten
Leeuwarden	Zaamslag	Rotterdam Hillegersberg	Sittard	Hoofddorp	Onstwedde
Franeker	Oostburg	Dordrecht		Amstelveen	Veendam
Sneek	Sluis	Geldermalsen		Weesp	Assen
Heerenveen		Tiel			Hoogeveen
Wolvega					
Steenwijk					
Meppel					

Route 7	Route 8	Route 9	Route 10	Route 11	Route 12
Hilversum	Ridderkerk	Weert	Pijnacker	Bavel	Roermond
Bussum	Rotterdam Zuidplein	Eindhoven	's-Gravenzande	Breda	Venlo
Laren Huizen	Rotterdam Ijsselmonde	Eindhoven Woensel	Naaldwijk	Oosterhout	Sevenum
Almere	Rotterdam Centrum	Veldhoven	Rijswijk	Made	Horst
Putten	Schiedam	Waalre	Den Haag	Bergen op Zoom	
Ermelo	Vlaardingen	Valkenswaard	Leidschendam	Roosendaal	
Nunspeet	Maassluis	Bladel	Voorburg	Etten Leur	
Epe	Hoogvliet	Reusel	Zoetermeer		
Oene	Spijkernisse	Helmond	Bodegraven		
	Hellevoetsluis				
	Middelharnis				

2.3 Current situation data

In this section, data from the current situation is described. This data is used in Chapter 4 as input data for the model that is made for the optimization of the current situation. To be able to focus on the models in Chapter 4 and illustrate the current situation here better, we give data regarding durations in this part. The matrix for the driving times is too large and therefore available in a supplement file in Appendix A. The travel times are extracted from the internet via an API called 'OpenStreetMap' that can be linked to Python. After collecting the driving times, the times are put in an Excel file to make them more readable.

Every location has a service or handling time. This is the amount of minutes the driver is at that location. As these are also two large lists, these times are available in two separate files in Appendix B. We use a GPS buddy system that tracks every vehicle to estimate the service times of a location. To ensure these times are representative of the real world, we verified the times by a driver of VSP.

Not every location has a time window. In total, 44 locations have one. A time window is the time before or after a location cannot be served anymore. This means that the driver needs to be at the location within the given time window for that location. These are displayed in Table 5. Here the end of the time windows are shown, every city also has an opening time window. The opening time window for most of the locations is 09:30, as the stores open around this time. Some locations open somewhat earlier.

Table 5. Time windows per location

Location	Time window	Location	Time window
Apeldoorn	11:00	Leeuwarden	12:00
Zutphen	11:00	Middelburg	11:00
Doetinchem	12:00	Goes	11:00
Enschede	11:00	Gouda	12:00
Amsterdam Centrum	11:00	Dordrecht	11:00
Barneveld	12:00	Maastricht	11:00
Arnhem	11:00	Alphen ad Rijn	12:00
Nijmegen	12:00	Leiden	11:00
Amersfoort	11:00	Groningen	12:00
Utrecht	11:00	Hilversum	11:00
Woerden	12:00	Bussum	11:00
Den Bosch	11:00	Putten	11:00
Haarlem	11:00	Rotterdam Centrum	10:30
Deventer	11:00	Weert	12:00
Alkmaar	11:00	Eindhoven	11:00
Tilburg	11:00	Helmond	11:00
Kampen	11:00	Rijswijk	11:00
Zwolle	12:00	Den Haag	11:00
Wijchen	12:00	Breda	11:00
Veenendaal	13:00	Oosterhout	11:30
Ede	11:00	Roermond	12:00
Venlo	12:00		

In Table 5, also some orange-marked numbers are present. VSP prefers to keep the time window, however, sometimes there is a possibility to park the vehicle just outside the city center and walk to the customer locations. Therefore, these can be considered soft time window constraints, while the time windows for the other locations are hard time window constraints. The difference between hard and soft time window constraints is that hard constraints need to be held at all costs and soft constraints could be violated if needed. Furthermore, the driver also needs to leave before the time window, meaning arrival time plus service times needs to be lower than these given times.

2.4 Performance current routes

For the performance of a route, we take the amount of minutes it takes to perform a route as the Key Performance Indicator (KPI). In Table 6, the times per route for both sets of days are given. These routes are calculated by adding up all the driving times between the locations in order of the route and handling times are also added for all locations in the route.

Table 6. Route times for current routes

	Tuesday-Friday	**Wednesday-Saturday**
Route 1	570 minutes (9.5 hours)	644 minutes (10.73 hours)
Route 2	569 minutes (9.48 hours)	732 minutes (12.2 hours)
Route 3	536 minutes (8.93 hours)	515 minutes (8.58 hours)
Route 4	549 minutes (9.15 hours)	504 minutes (8.4 hours)
Route 5	489 minutes (8.15 hours)	534 minutes (8.9 hours)
Route 6	563 minutes (9.38 hours)	562 minutes (9.37 hours)
Route 7	680 minutes (11.33 hours)	545 minutes (9.08 hours)
Route 8	615 minutes (10.25 hours)	640 minutes (10.67 hours)
Route 9	590 minutes (9.83 hours)	610 minutes (10.17 hours)
Route 10	590 minutes (9.83 hours)	594 minutes (9.4 hours)
Route 11	540 minutes (9 hours)	658 minutes (10.97 hours)
Route 12	489 minutes (8.15 hours)	425 minutes (7.08 hours)
Total Time	**6780 minutes (113 hours)**	**6963 minutes (116.05 hours)**

The times given in Table 6, show that most of the times for the current routes are somewhere between 500 and 600 minutes. This is also what VSP desires as drivers should work around 10 hours each day. However, in the times, it is visible that some routes do exceed this time. With the optimization of these routes, this is also a constraint that needs to be considered. VSP prefers to have no route exceeding this 10-hour limit. For now, as can be seen in the total time, on average the routes are below 10 hours. In the case that now all routes can be within 10 hours, VSP prefers to have an average of below 10 hours.

2.5 Conclusion

In Chapter 2, the goal was to understand the current situation. By analyzing the process, the performance of the current process is identified. Sub research question 1 is answered with the routing process clear and quantifying the performance in terms of time.

With this overview of the current situation, the next step is to get a better understanding of how VRP problems are solved and optimized. Next to that, we want to understand what the influence of EVs is on the routing of VSP logistics. In Chapter 3, a literature review is conducted to get information on these topics.

3 Literature review

In Chapter 3, a literature review is performed. Here, the focus is to answer the research questions stated in Section 1.4. In specific, sub-questions 3, 4, and 5 are answered, regarding modeling techniques, possible solution methods, and electric vehicles. First, a general introduction to Vehicle Routing Problem (VRP) problems is given. In the end, a conclusion on the found literature is given, mentioning which parts of the review will be used to be able to achieve the research goal and solve the core problem.

3.1 Vehicle Routing Problem

Laporte (2007) gives the following description of the VRP. It is explained as a process where the aim is to design a set of m minimum cost vehicle routes through n customer locations, so that each route starts and ends at a common location and some side constraints are satisfied. Examples of common applications mentioned in the paper are newspaper round, food delivery, and milk collection.

Dantzig & Ramser (1959) were the first to introduce a problem similar to the VRP. This problem was called the 'Truck Dispatching Problem'. Not that many years later Clarke & Wright (1964) generalized this problem to a linear optimization problem, making it the VRP known today. After all these years, the VRP is one of the most studied problems in Operations Research.

Braekers et al. (2016) mention in their literature review, that current VRP models differ a lot from the problem introduced around the early 60s. Nowadays, the aim is to incorporate real-life complexities into the problem. These complexities are, for example, traffic congestion, time windows for pickup and delivery, and input information that changes over time.

Lenstra & Kan (1981) studied the complexity of the VRP and showed that the problem is NP-hard. Therefore, exact algorithms can be used to solve small problem instances. If the problem instances become too large, heuristics or metaheuristics are more suitable to solve the VRP. Real-life problems are most of the time too large, which results in the use of these heuristics.

Cordeau et al. (2007) mention multiple variations of the VRP. The different types are: Classical VRP, VRP with Time windows (VRPTW), Inventory Routing Problem, and Stochastic VRPs. Lin et al. (2016) mentioned a new variant based on the Green Vehicle Routing Problem (GVRP) introduced by Erdoğan & Miller-Hooks (2012), which is the Electric Vehicle Routing Problem (EVRP). Fernández Gil et al. (2022) show in their review that a lot of research is focused on the reduction of emissions gasses when considering the GVRP. Also, it shows that the transition to electric vehicles is researched frequently. In this literature, the focus will be on the VRPTW and EVRP. First, a closer look is taken at the formulation of VRP in general, later than also adding the use of time windows. Later, a further look is taken at how to incorporate EVs into these VRP and VRPTW types of problems.

3.2 Mathematical formulation VRP and VRPTW

In this section, the informal description and mathematical formulation of both the VRP and VRPTW are given and explained. Section 3.2.1 focuses on VRP problems in general, whereas Section 3.2.2 focuses on VRPTW.

3.2.1 VRP formulation

In literature, the VRP has been researched extensively. Therefore, a lot of different formulations of the problem can be found. In this literature review, the formulation of Kallehauge et al. (2005) is used. They have a VRPTW formulation, but we adapt that formulation for the classical VRP formulation, and section 3.2.2 then shows their complete formulation for the VRPTW.

Kallehauge et al. (2005) define the problem as follows. The VRP has a fleet of vehicles V, a set of customers N, and a directed graph G. The fleet is considered homogeneous, which means that all vehicles are identical. The constructed graph consists of $|N| + 2$ vertices, where the customers are denoted $1,2, \dots, n$ and the depot is vertex 0. The set of arcs, A, represents direct connections between the depot and customers and among the customers. No arcs are ending at vertex 0 or originate from vertex $n + 1$. With each arc (i, j), where $i \neq j$, there is a cost c_{ij} and a time t_{ij}, which may include service time at customer i. Each vehicle as a capacity q and each customer i a demand d_i. It is assumed that q, d_i, c_{ij} are nonnegative integers. The model then contains two decision variables x and s. For each vehicle k and arc (i, j), where $i \neq j$, $i \neq n$, $j \neq 0$, x_{ijk} is define as

$$x_{ijk} = \begin{cases} 1, if\ vehicle\ k\ drives\ directly\ from\ i\ to\ j \\ 0, otherwise \end{cases}$$

The other decision variable s_{ik} is defined for each customer i and each vehicle k and it gives the time vehicle k starts to service customer i. In the case that vehicle k does not service customer i, the decision variable does not have a meaning and is therefore considered irrelevant. Kallehauge et al. (2005) state that the goal of the program is to minimize total cost in such a way that each customer is visited once and every route begins and ends at the depot.

This informal description is then formalized in the following way by Kallehauge et al. (2005).

$$min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \qquad (1)$$

s.t.

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1, \qquad \forall i \in N, \qquad (2)$$

$$\sum_{i \in N} d_i \sum_{j \in N} x_{ijk} \leq q, \qquad \forall k \in V \qquad (3)$$

$$\sum_{j \in N} x_{0jk} = 1, \qquad \forall k \in V \qquad (4)$$

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0, \qquad \forall h \in N, \forall k \in V \qquad (5)$$

$$\sum_{i \in N} x_{i0k} = 1, \qquad \forall k \in V \qquad (6)$$

$$s_{ik} + t_{ij} - M(1 - x_{ijk}) \leq s_{jk}, \qquad \forall i, j \in N, \forall k \in V, \qquad (7)$$

$$x_{ijk} \in \{0,1\}, \qquad \forall i, j \in N, k \in V \qquad (8)$$

In this formulation, M is a large enough number. The objective function given in (1), aims to minimize the travel cost. Constraint (2) makes sure that every customer is visited exactly once. With constraint (3), the program makes sure that the capacity of a truck is not exceeded. Constraints (4) and (6) ensure that every truck will start and end in the depot, as this is customer 0. Constraint (5) will make a truck leave a customer's

location if it goes there. Lastly, constraint (7) creates a relation between the departure time of a truck at a customer and the next customer on the route.

Kallehauge et al. (2005) also show a constraint that could incorporate a maximum amount of vehicles to be considered. This could be convenient if it is not sure if the routing could be done with fewer vehicles. The constraint looks like the following;

$$\sum_{k \in V} \sum_{j \in N} x_{0jk} \leq |V|, \qquad \forall k \in V, \forall j \in N \tag{9}$$

3.2.2 VRPTW formulation

To be able to consider time windows for each customer i, Kallehauge et al. (2005) add parameters a_i and b_i, which are the opening and closing time of the time window for customer i respectively. Next to these parameters, the following constraint also needs to be added.

$$a_i \leq s_{ik} \leq b_i, \qquad \forall i \in N, \forall k \in V \tag{10}$$

Constraint (10) makes sure that each customer is visited within their time window. If the vehicle arrives before the opening time, it needs to wait to serve customer i until the time window is opened.

3.3 Optimization methods VRP(TW)

As mentioned in Section 3.1, the VRP and VRPTW problems are NP-hard problems and therefore exact algorithms can only be used for small instances of the problems (Lenstra & Kan, 1981). Therefore, in this section, the goal is to identify heuristics and metaheuristics to be able to solve large instances of these problems. The aim is to get a solution that is near optimal for these large instances.

3.3.1 Constructive Heuristics

First, a look is taken at constructive heuristics that can make an initial solution. Afterward, we consider improvement heuristics that can improve these initial solutions. Solomon (1987) mentions in his paper that constructive heuristics can be divided into sequential and parallel methods. Sequential methods create routes one route at a time until all customers are put into a route. Where parallel methods are scheduling routes simultaneously.

3.3.1.1 Savings Algorithm (VRP)

The first heuristic considered is the savings algorithm by Clarke & Wright (1964), which can be seen as a parallel method. This is a very well-known algorithm. Where every customer first has their own tour and the routes are determined by combining two routes based on the cost that can be saved from doing that. The savings of putting two routes together is done with the following equation:

$$savings_{ij} = d_{i0} + d_{0j} - d_{ij}$$

Here i and j are both customer locations at the end or beginning of an existing route. A list is created with all possible combinations of end customers. The combination with the most savings is put together in a route. This is done until no savings can be made.

3.3.1.2 Nearest Neighbor Insertion Algorithm (VRP)

Joshi & Kaur (2015) mention that the Nearest Neighbor Insertion Approach is one of the easier algorithms to solve optimization problems. The idea is that all routes for all vehicles are empty at the start and that the location closest to the last added location is added next. This is done for a vehicle route until capacity is reached and a new route for an empty vehicle is created. This is done until all customers are served by the

routes. In general, it is used to create an initial solution as this approach is not known for creating near-optimal solutions.

3.3.1.3 Parallel construction approach (VRPTW)

In the paper by Chiang & Russell (1996), a constructive heuristic is proposed. This is a two-phase approach to the VRPTW, it is a parallel construction approach with a simulated annealing tour improvement heuristic. The approach differs from other approaches as the improvement process is invoked periodically during the construction process. Applying an improvement procedure during route construction helps to improve solutions to the VRPTW when using a local search method (Chiang & Russell, 1996).

The parallel construction procedure given in the paper is based on the insertion heuristic of Solomon (1987) but differs from the sequential approach in that a specified number of routes are constructed in parallel rather than one at a time. This parallel approach by Chiang & Russell (1996) is described as follows. They assume that there are n customers that need to be served. For each customer i, let

$$q_i = demand\ for\ pickup\ or\ delivery\ for\ customer\ i$$

$$s_i = required\ service\ time\ for\ customer\ i$$

$$e_i = earliest\ time\ service\ can\ begin\ at\ customer\ i$$

$$l_i = latest\ time\ service\ can\ begin\ at\ customer\ i$$

$$t_{ij} = travel\ time\ between\ two\ customers\ i\ and\ j$$

$$d_{ij} = distance\ between\ two\ customers\ i\ and\ j$$

$$b_i = current\ time\ when\ service\ can\ begin\ at\ customer\ i$$

$$b_{ij} = time\ when\ service\ can\ begin\ at\ customer\ j\ given\ \ that\ customer\ i\ is\ inserted\ immediately\ before\ customer\ j\ in\ the\ route$$

$$Q_v = capacity\ of\ vehicle\ v, v = 1,2, \ldots, V$$

The calculation of b_{ij} is given by Chiang & Russell (1996) as the following:

$$b_{ij} = \max\{e_j, b_i + s_i + t_{ij}\}$$

They give that the arrival at customer j before e_j requires a wait time w_j, which is given as $w_j = e_j - (b_i + s_i + t_{ij})$. Arrival after l_j is infeasible, therefore time window constraints are treated as hard constraints.

Next, Chiang & Russell (1996) give that the parallel construction approach requires an initial estimate, V, of the number of vehicles required. This can be, for example, the number of existing routes. When V is determined, the parallel insertion heuristic requires n iterations, assigning one customer per iteration to the best available route. Customers are selected for route insertion in a particular order. Three ordering rules are used to make sure that the time window feasibility is kept during the construction phase. Rule one is for selecting the next customer based on the smallest early time window parameter e_j. Rule two is based on the tightness of the time window as calculated by $100(l_j - e_j) - d_{0j}$, where d_{0j} denotes the distances form the depot to customer j. The number 100 is used as a weight to emphasize the tightness of the time window

relative to the distance from the depot. The last rule is based on the largest value of d_{0j}. These rules together will generate three orderings of customers to be selected to be inserted into a route.

Chiang & Russell (1996) describe two measures that can be considered for inserting customer j between customers i and k on route r. These measures are c_1 and c_2 and are given as follows:

$$c_1(i,j,k) = d_{ij} + d_{jk} - d_{ik}$$

$$c_2(i,j,k) = b_{jk} - b_k$$

The measures are used in criterion $c_3(r)$ as follows:

$$c_3(r) = \alpha_1 c_1(i,j,k) + \alpha_2 c_2(i,j,k), \quad where \qquad \alpha_1 + \alpha_2 = 1$$

This criterion will then select the best route in the following way:

$$r = argmin\{c_3(v); v = 1,2,\dots,V\}$$

The solution of the parallel insertion heuristic represents the best of six passes. The parameters used are $(\alpha_1 = 1,\ \alpha_2 = 0)$ and $(\alpha_1 = 0,\ \alpha_2 = 1)$. Together with the three ordering rules six passes are created to determine the best solution.

Lastly, Chiang & Russell (1996) state that the initial choice for the number of vehicles can yield an infeasible solution with some customers unrouted. If this is the case, Solomon's insertion heuristic can be used to schedule the set of unrouted customers. If all customers are routed within the number of vehicles, another iteration of the construction procedure with one vehicle less can be performed, to see if the number of vehicles can be lowered.

3.3.1.4 Sequential Simulated Annealing, initial solution (VRPTW)

As later will be discussed, the sequential simulated annealing procedure proposed by Woch & Łebkowski (2009) yields good results specifically for the VRPTW. In their paper, an algorithm is proposed to make an initial solution taking into account time windows. That algorithm starts with ordering all customers ascending by time window. A list with the smallest time windows is created first, and those customers will be placed on a route first to ensure that the most problematic customers are put in a correct position. First, a customer is tried to be put in a place on an existing route, but when this is not possible, a new route for this customer will be created. Next to a list with customers that have a small time window, there also is a list created with customers that have a larger time window. When the first list is completely gone through, the second list with customers is tried to fit in the routes that are made by going through the first list. This is done until all customers are placed on a route. The number of customers in the small TW and large TW lists can be varied.

3.3.2 Improvement heuristics

With improvement algorithms, the idea is to improvement solutions with relatively simple adjustments to the current solutions to maybe find better neighbor solutions. With these types of approaches, an optimal solution is not guaranteed and the search process is stopped when no better neighbor solution can be found.

As stated in the work by Bräysy & Gendreau (2005), two acceptance strategies are commonly used in solution improvement methods. These strategies are first-accept (FA) and best-accept (BA). The first-accept strategy selects the first neighbor that satisfies the predefined acceptance criterion. The best-accept strategy examines all neighbors satisfying the criterion and then picks the best one among them.

In the article by Dror & Levy (1986), they talk about a Greedy Algorithm, that first looks at the savings from all possible neighborhood solutions and then takes the neighbor with the biggest savings. This is an approach that uses a BA strategy to select a neighbor solution. This is done until no improvement can be made anymore. Such an algorithm is called greedy because it will also take the option with the biggest savings only taking into account the current solution. With taking the option that has the biggest savings now, it is not known if taking fewer savings first could have resulted in bigger savings later. These types of approaches, therefore, produce local optimum, where these solutions may be far from the optimal solution.

3.3.3 Neighborhood Operators

To be able to explore different solutions within a neighborhood structure, operators are needed to change the current solution to a new solution. In the paper by Goel & Gruhn (2008), 5 operators are described in relation to a VRP problem. These operators are Insert, Remove, Relocate, Replace, and Swap.

Insert
The Insert-operator chooses an unscheduled customer randomly and inserts it into the tour of the vehicle with the lowest costs. If the location cannot be feasibly inserted, the solution is not changed.

Remove
The Remove-operator chooses a scheduled customer randomly and removes it from the tour is it in.

Relocate
The Relocate-operator, which is also known as the MOVE-operator, randomly chooses a scheduled customer removes it from the tour it is assigned to, and inserts the customer into a random other tour for the lowest possible costs.

Replace
The Replace-operator will randomly choose and remove two customers from different tours and the first customer is inserted in the tour of the second customer in the place with the lowest costs. The second customer will not be reassigned.

Swap
The Swap-operator chooses two random customers and removes them from their current tours and inserts them into the respective other tour at the place with the lowest cost.

3.3.4 Neighborhood structures

When making use of metaheuristics or heuristics in general, it is important to have a neighborhood to be able to get results, as mentioned before, the neighborhood structure can have a significant impact on the eventual results. Chiang & Russell (1996) define a neighborhood as the following, where a neighborhood is a configuration that consists of those configurations that result from perturbing the initial one by the shift of customers from one route to another or the interchange of the location of two customers. A configuration can be seen as the number of vehicle routes and the assignment of customers to those routes without violation of time window constraints (Chiang & Russell, 1996). They state two types of neighborhood structures specifically for a simulated annealing algorithm, which are a modified version of the k-node interchange mechanism of Christofides & Beasley (1984) and a neighborhood that is based on the λ-interchange mechanism of Osman (1993).

3.3.4.1 Neighborhood N1

The description of Neighborhood N1 is done by Chiang & Russell (1996) as the following, consider a customer i on route p and the L $(L = 1)$ customers that follow customer i on route p. Set $M1$ is defined by

customer i and the L customers that follow i on route p. Set $M2$ is defined as the set of two customers not on route p that are close to the customers in set $M1$. Specified, let the points α and β be the two customers on route p that immediately precede and succeed the set $M1$, respectively. The set $M2$ is generated by the two customers not on route p whose insertion cost using Euclidian distance metric between α and β is minimal, i.e. $minimize_{j \notin R_p} d(\alpha, j) + d(j, \beta)$. The sets $M1$ and $M2$ define the size of neighborhood in the search process. With $L = 1$, the number of customers in each of the sets $M1$ and $M2$ are two customers, which results in a total of four customers in $M1 \cup M2$. They define the $N1(S)$ neighborhood of a given solution S as the set of all neighboring solutions achievable by deleting the four customers in $M1 \cup M2$ and re-placing them in all possible routes, taking into account the time window constraints.

Each customer i then has a $U_i = M1 \cup M2$, with the complete subset $\mathcal{U}$ being $U_i, i = 1,2, \dots, n$. Within the simulated annealing heuristic at every iteration, the algorithm selects four customers in the subset U_i and will evaluate possible route insertions. If the complete neighborhood would have been searched through, it would involve V^4 possible route assignment combinations. This would yield an intractable computational effort for large instances. Making use of a preprocessing step can help to make it tractable, by first determining the best two routes for each point in U_i. This is done based on the criterion to determine the best two routes per customer c_3 from section 3.3.1.3. This pre-step reduces the possible route options per subset U_i to $2^4 = 16$. The idea is that the neighborhood search partially searches through the large neighborhood and only looks at promising moves within the neighborhood. With looking through the options, constraints are taken into account.

3.3.4.2 Neighborhood N2

The second structure Chiang & Russell (1996) is a search method that examines all possible combinations of pairs of routes for exchange. Let permutation σ be the order of route indices for a given solution $S = \{R_1, \dots, R_s, \dots, R_t, \dots, R_V\}$, in that case, all possible combinations of pairs of routes (R_s, R_t) can be examined without repetition in a following order as presented in the paper by Chiang & Russell (1996):

$$
\begin{aligned}
&(R_{\sigma(1)}, R_{\sigma(2)}), \dots, (R_{\sigma(1)}, R_{\sigma(V)}),\\
&(R_{\sigma(2)}, R_{\sigma(3)}), \dots, (R_{\sigma(2)}, R_{\sigma(V)}),\\
&\quad \vdots\\
&(R_{\sigma(V-2)}, R_{\sigma(V-1)}), (R_{\sigma(V-2)}, R_{\sigma(V)}),\\
&(R_{\sigma(V-1)}, R_{\sigma(V)}).
\end{aligned}
$$

Figure 5. Figure containing possible combinations of pairs of routes by Chiang & Russel (1996)

Neighboring solutions are generated using (0,1), (1,0), and (1,1) operators that represent either a shift or exchange process. The first two represent a move of a customer from one route to the other. The last one represents an exchange of two customers between the two routes. The customers of the selected pairs of routes would be searched completely for possible improvement. Candidate solutions are chosen in an ordered manner $S' \in N2(S)$, where S is the previous solution.

3.3.5 Metaheuristics for solving VRPTW

In section 3.3.2, metaheuristics that can be used for solving the VRPTW problem will be discussed. First, multiple approaches found in the literature will be discussed and performance will be evaluated. The approach that seems the most promising in VSPs situation will be discussed in more detail.

Chiang & Russell (1996) show that a Simulated Annealing algorithm can give good performance for VRPTW, but they do state that the choice for the type of neighborhood structure appears to have a significant effect on algorithm performance. Afifi et al. (2013) also show in their paper that a simulated annealing algorithm can perform very well for a VRPTW problem. Woch & Łebkowski (2009) also indicate that the sequential simulated annealing can be successfully applied to the vehicle routing problem with time windows. Tavakkoli-Moghaddam et al. (2011) show that their proposed simulated annealing algorithm can find good solutions in a reasonable time. In the paper by Breedam (2001), a comparison between a heuristic and two metaheuristics is made. The heuristic is a descent heuristic and the metaheuristics are simulated annealing and tabu search approaches. Here, the author shows that the metaheuristic can perform better, however, between the two metaheuristics, one does not outperform the other.

In the article by Kytöjoki et al. (2007), an efficient two-phase variable neighborhood search is presented. This method is specifically aimed at solving very large-scale real-life vehicle routing problems. Their search heuristic can find high-quality solutions for problem instances with up to 20,000 customers within reasonable times. Another variable neighborhood search approach is the one from Bräysy (2003), this is a modification of the variable neighborhood search of Mladenović & Hansen (1997), where with this new version also time windows are considered. The author reports good results for instances of up to 400 customers, where it also outperforms other presented heuristics or metaheuristics.

Another type of metaheuristic that is widely used to solve VRPTW problems are tabu search algorithm. An example of the implementation of a tabu search algorithm applied in a real-life case is done in the paper by Barbarosoglu & Ozgur (1999). They deal with the design of a heuristic algorithm to solve the VRP for a well-known distribution company in Turkey. They use a tabu search approach as this is known to provide good practical solutions. Their proposed theoretical performance was accepted and the company uses the method in its daily operations. In the paper by Montané & Galvão (2006), another tabu search algorithm is proposed. Their algorithm uses three types of movements in terms of the gathering of inter-route adjacent solutions: relocation, interchange, and crossover movements. To be able to achieve diversification and intensification, they used two different strategies for selecting new customers. Either the first admissible movement or the best admissible movement. They report low computational times and good results in general for their proposed tabu search procedure.

Lalla-Ruiz & Voß (2020) show in their paper that using a POPMUSIC matheuristic approach that uses reduced versions of the problem instance at hand as a sub-problem to solve the overall problem. They show that the obtained results indicate that it allows to improve majority of the best solutions provided by a solver while reducing the computational effort.

Another metaheuristic that can be used for solving VRPTW, is the genetic algorithm. An example of a genetic algorithm is proposed in the paper by Ursani et al. (2011). They developed a localized genetic algorithm which at that time was outperforming other heuristics on small-scale problems. In the paper by Qiu et al. (2023), an improved memetic algorithm is proposed, which combines a genetic algorithm with procedures like local search procedures. They that the algorithm has good performance for small instances.

As can be seen from the literature found, a lot of different heuristics or metaheuristics can be used and applied to solve the VRP or VRPTW. From all the given solution methods given, next, we will highlight one approach in more depth. Therefore, the idea behind the variable neighborhood search is discussed in a more detailed way.

Variable Neighborhood Search

A variable neighborhood search (VNS) is a metaheuristic that explores the solution space through systematic changes in neighborhood structures (Mladenović & Hansen, 1997). A basic VNS consists of a local search procedure and a shake procedure which are executed alternatively until a stopping criterion is met. The local search procedure tries to find the local optimum, which ensures intensification. The shaking procedure randomizes the solution which leads the solution to other parts of the solution neighborhood. The shaking procedure ensures diversification (Bezerra et al., 2023).

There are multiple VNS variants, but General Variable Neighborhood Search (GVNS) has an important property, as it uses Variable Neighborhood Descent (VND) as a local search method, it can return a local optimum concerning all used neighborhoods. This is an important property because these kinds of local optima are more likely to be a global optimum than a generated solution as a local optimum for one neighborhood (Bezerra et al., 2023).

In the paper by Marinho Diana & de Souza (2020), the default VND is presented in the following way. There is a set of different neighborhood structures $N_s = \{N_1(s), N_2(s), \dots, N_k(s)\}$. The procedure then starts with exploitation of the $N_1(s)$ neighborhood, according to a defined search strategy for this structure. After the neighborhood exploitation of the first neighborhood, the exploitation of the second neighborhood $N_2(s)$ starts. If improvement in the second is found, the exploitation of the first neighborhood is resumed. If no improvement is found, the exploitation of the third neighborhood $N_3(s)$ is started. This is repeated until the *k*th neighborhood structure is explored without improvement in the current solution or until some stopping criterion is satisfied (Marinho Diana & de Souza, 2020). The shake procedure is in essence a random search, where the procedure consists of selecting a solution of the neighborhood randomly (García-López et al., 2002).

3.4 Electric Vehicles in VRP(TW)

In section 3.4, we will take a look at what happens to the VRP problem when EVs are implemented. In the last decade, EVs have been considered in a growing number of models and methods for vehicle routing problems (Qin et al., 2021).

One element that needs to be added when dealing with EVs is charging. In general, the driving range of an EV is shorter and charging will take a longer time than refueling a diesel vehicle. In the paper by Qin et al. (2021), they state that the EVRP is a straightforward extension of the classic VRP by involving EVs and as mentioned the operations of recharging. Their presented model is based on the one given by Schneider et al. (2014).

Recharging can be done with different charging strategies. Either the batteries are recharged fully or the batteries are charged partially. The latter can lead to significant time reduction (Keskin & Çatay, 2016). Therefore, first, the additions to the mathematical formulation of the model that considers full recharging are given. Second, we give the constraints and parameters needed to consider partial recharging strategies.

In addition to the already given VRP formulation in section 3.2, some new sets, parameters, variables, and constraints need to be added based on the model given by Qin et al. (2021) and Schneider et al. (2014), for considering full recharging. The set N is extended into N' where $N' = N \cup F'$, with the dummy set F' related to the set F of recharging stations. Parameter d_{ij} gives the distance between location i and j. Next, a constant h is used for battery consumption (per unit distance) and next to that each vehicle will now also have a battery capacity Q. Being at a charging station, the difference between the present battery level and

a capacity of Q is recharged with a charging rate of g. A decision variable y_i is used as the remaining battery level on arrival at a next customer location i.

With these new sets, parameters, and variables, the following constraints will be added to the model based on the additions from Qin et al. (2021) and Schneider et al. (2014).

$$\sum_{j \in N',\ i \neq j} x_{ijk} \leq 1, \qquad \forall i \in F', \forall k \in V, \tag{11}$$

$$s_{ik} + t_{ij}x_{ijk} + g(Q - y_i) - (M + gQ)\left(1 - x_{ijk}\right) \leq s_{jk}, \quad \forall i \in F', \forall j \in N', i \neq j, \forall k \in V, \tag{12}$$

$$0 \leq y_j \leq y_i - hd_{ij}x_{ijk} + Q\left(1 - x_{ijk}\right), \qquad \forall j \in N', \forall i \in N, i \neq j, \forall k \in V, \tag{13}$$

$$0 \leq y_j \leq Q - hd_{ij}x_{ijk}, \qquad \forall i \in F', \forall j \in N, i \neq j, \forall k \in V, \tag{14}$$

With new constraint (11), the model ensures that each charging station must be visited at most once. As these charging stations are not customers, these locations do not have to be visited. In constraint (12), charging time is considered. In constraints (13) and (14), the battery levels are updated per visit of a customer or charging station. These constraints also ensure that the battery level will not go below 0.

In the case of partial charging, the following new sets, parameters, variables, and constraints are added to the model. These additions are based on the model given by Keskin & Çatay (2016) and their model is based on the formulation given by Schneider et al. (2014). The difference in the model is that Keskin & Çatay (2016) add a decision variable Y_i which represents the battery state of charge on departure from recharging station i. Constraints (12) and (14) are adjusted to take the battery state of charge on departure into account in the following way.

$$s_{ik} + t_{ij}x_{ijk} + g(Y_i - y_i) - (M + gQ)\left(1 - x_{ijk}\right) \leq s_{jk}, \qquad \forall i \in F', \forall j \in N', i \neq j, \tag{15}$$

$$0 \leq y_j \leq Y_i - hd_{ij}x_{ijk} + Q\left(1 - x_{ijk}\right), \qquad \forall i \in F', \forall j \in N', i \neq j, \tag{16}$$

With the adjustment in constraint (15), recharging times are now considering the decision on how much needs to be charged. Furthermore, gives constraint (16) the new battery state at station j, considering the decision to charge a certain amount at charging station i. Next to these adjustments, constraint (17) is added to give the boundaries of the added decision variable Y_i.

$$y_i \leq Y_i \leq Q, \qquad \forall i \in F', \tag{17}$$

Bac & Erdem (2021) show a model that also considers a different objective function to deal with the feasibility of the solutions for the EVRPTW. They add penalties to the objective function, whereas in a minimization problem, these penalties need to be zero to have a feasible solution. The objective function is given in (18).

$$min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij}x_{ijk} + \sum_{i \in N} \theta_i + \sum_{k \in V} O_k \tag{18}$$

Here, two new variables are added, which are θ_i and O_k, which are the deviation of the time windows and the overtime of a route respectively. Furthermore, Bac & Erdem (2021) add two constraints to keep track of these penalties. These constraints are (19) and (20). Constraints (21) and (22) and nonnegativity constraints.

$$\left(\sum_{k \in V} s_{ik} + H_i x_{ijk}\right) - b_i \leq \theta_i, \qquad \forall i \in N, \tag{19}$$

$$O_k \geq s_{nk} - T_M, \qquad \forall k \in V, \qquad (20)$$

$$\theta_i \geq 0, \qquad \forall i \in N, \qquad (21)$$

$$O_k \geq 0, \qquad \forall k \in V, \qquad (22)$$

With the mathematical formulation of the EVRTPTW considering both full and partial recharging strategies, in Section 3.5 a look will be taken at heuristics for solving the EVRPTW.

3.5 Optimization methods EVRP(TW)

When extending the VRP to be able to use EVs, some new algorithms and heuristics to solve the problem need to be explored. Euchi & Yassine (2022) propose a hybrid metaheuristic algorithm that can solve the electric vehicle routing problem with battery charging stations. In their problem, the vehicles have limited delivery capacity and rely completely on their limited battery capacity. The Hybrid Variable Neighborhood Search that is proposed shows that it can detect good quality solutions.

Schneider et al. (2014) introduce the electric vehicle routing problem with time windows and recharging stations, which incorporates the possibility of recharging at any of the available stations using an appropriate charging scheme. Next to recharging, the proposed problem also includes freight capacity and customer time window constraints. To solve the problem, they propose a hybrid heuristic that combines a variable neighborhood search algorithm with a tabu search heuristic. Their results show that the proposed heuristic can have a high performance on newly designed instances, but also on benchmark instances.

Keskin & Çatay (2018) also introduce an Electric Vehicle Routing Problem with Time Windows (EVRPTW), they see it as an extension of the well-known VRPTW where EVs are used instead of normal vehicles. They formulated the problem as a mixed integer linear program and solve small instances using a solver. For solving large instances, they propose a matheuristic approach that couples the Adaptive Large Neighborhood Search (ALNS) with an exact approach. They show the effectiveness of the proposed matheuristic on benchmark instances.

Keskin et al. (2021) researched the EVRPTW and then also introduce stochastic waiting times at recharging stations. There is the possibility that there is a queue at the charging stations when not enough chargers are available. Their problem takes that situation into account. They present a two-stage simulation-based heuristic using ALNS. They show that their proposed simulation-based solution approach can provide good solutions in terms of quality and computational time. It also shows that the uncertainty in waiting times may have a significant impact on route plans.

The paper by Bruglieri et al. (2015) presents a variable neighborhood search branching for solving EVRPTW. They mention that especially the poor battery autonomy is the Achille's heel of EVs as many stops are needed. Their model aims to optimally route EVs for handling a set of customers in time considering the recharging needs during the trips. The paper proposes a mixed integer linear programming formulation of the problem, where the battery recharging level reached at each station is a decision variable to be able to generate more flexible routes. They minimize total travel, waiting, and recharging time, as well as they want to minimize the number of EVs deployed. Their solution method, which is the variable neighborhood search branching is designed for solving the problem within reasonable times. A comparison is also made with other models that let EVs always charge to full battery level. Where the presented model is able to outperform the models where full charging is considered.

In the paper by Keskin & Çatay (2016), a model for partial charging as an extension of the general EVRPTW is presented. They develop an adaptive large neighborhood search algorithm that can solve the

problem efficiently. Their algorithm can effectively find high-quality solutions and they show that partial charging options may significantly improve routing decisions.

Another approach to solving the EVRPTW is presented in the paper by Lam et al. (2022). They present a Branch-and-cut-and-price algorithm for the problem, considering Piecewise-linear recharging and capacitated recharging stations. They show that their algorithm is able to show good performance on benchmark instances.

In the paper by Wang & Zhao (2023), the benefits of partial recharging strategies are researched. They propose a formulation of a Path-Based model and develop a hybrid large neighborhood search algorithm that combines a large neighborhood search algorithm with a set partitioning component. Their algorithm is benchmarked against state-of-the-art methods on public large-scale instances and can find new local optimal solutions. Demonstrating the benefits of partial recharging strategies.

Erdelić et al. (2019) apply a metaheuristic approach based on the ruin-create principle to solve the EVRPTW. They consider full charging at the recharging stations and consider policies for single and multiple recharges during the route. Their metaheuristic was used to solve the problem for bigger instances and they used a commercial solver to solve the problem for smaller instances.

Bac & Erdem (2021) research the optimization of electric vehicle charging schedules considering partial recharges. They propose a framework for EVRPs with VNS and VND heuristics. Constraints considered are time windows and partial recharging. They show that their proposed method can perform well for large-size real-life problem instances.

Mao et al. (2020) investigate the integration of multiple recharging options, which are partial recharging and battery swapping. They present an improved ant colony optimization (ACO) algorithm that is hybridized with insertion heuristic and enhanced local search to solve the problem. They show that their approach can also help save costs when using partial recharging and battery swapping.

In the research by Raza et al. (2022), the recent advancements in reinforcement learning in solving VRP problems are presented. More and more research is published on using these techniques for solving VRP problems and its variants. Lin et al. (2022) propose an end-to-end deep reinforcement learning (DRL) framework to solve the EVRPTW. Their model is trained using gradient policy without rollout bassline. They show that their proposed model is able to efficiently solve EVRPTW instances of large sizes and performs better than the solution approach by Schneider et al. (2014). Another deep reinforcement learning application is done by Chen et al. (2022). Their paper proposes an end-to-end DRL method with a two-stage training strategy. Their experimental results show that the proposed method outperforms the state-of-the-art methods and is generalizable to different problem sizes as well.

In Table 7, we give an overview of found solution methods for the EVRPTW problem. Per reference, we give the different features and solution approaches used in the articles. These references are found using the following search strings:

- Electric vehicle routing problem with time windows
- Electric vehicle routing problem with time windows solution method
- Electric vehicle routing problem with time windows solution approach
- optimization Electric vehicle routing problem with time windows

Table 7. Overview solution method EVRPTW

	Features							
Reference	Fully charging?	Partial charging?	Energy Consumption model?	Solver or Heuristic?	Amount of customers?	Objective function?	Real-life or artificial instances?	**Solution Approach**
Schneider et al. (2014)	Yes	No	Battery level based on distance driven	Heuristic; Solver (CPLEX 12.2)	Heuristic: Up to 100; Solver: Up to 15;	Minimize distance;	Artificial	Hybrid VNS/TS;
Keskin & Çatay (2018)	Yes	No	Battery level based on distance driven	Heuristic; Solver (CPLEX 12.6.2)	Heuristic: Up to 400; Solver: Up to 15;	Minimize energy cost;	Artificial	Matheuristic;
Bruglieri et al. (2015)	Yes	Yes	Battery level based on distance driven	Heuristic; Solver (CPLEX 12.5)	Heuristic: up to 10; Solver: up to 10;	Minimize travel, waiting, recharging time;	Artificial	VNSB;
Keskin & Çatay (2016)	Yes	Yes	Battery level based on distance driven	Heuristic; Solver (CPLEX 12.6.1)	Heuristic: up to 200; Solver: up to 15;	Minimize distance;	Artificial	ALNS;
Lam et al. (2022)	Yes	No	-	Solver (Nutmeg);	Solver: up to 100;	Minimize cost;	Artificial	BPC;
Wang & Zhao (2023)	Yes	Yes	Battery level based on distance driven	Heuristic; Solver (Commercial solver)	Heuristic: up to 1000 customers; Solver: up to 15 customers;	Minimize distance and total fixed costs;	Artificial	Hybrid LNS;
Bac & Erdem (2021)	Yes	Yes	Battery level based on distance driven	Heuristic; Solver (CPLEX)	Heuristic: up to 1200; Solver: up to 16;	Minimize travel, waiting, recharging time;	Real-life; Artificial	VNS; VND
Mao et al. (2020)	Yes	Yes	Battery level based on distance driven	Heuristic;	Heuristic: up to 100;	Minimize total cost;	Artificial	ACO;
Lin et al. (2022)	Yes	No	Energy consumption given per arc(i,j)	Heuristic; Solver (CPLEX 12.1, Matlab.)	Heuristic: up to 100; Solver: up to 10;	Minimize distance;	Artificial	DRL;
Chen et al. (2022)	Yes	No	Battery level based on distance driven	Heuristic;	Heuristic: up to 100;	Minimize distance;	Artificial	DRL;
Erdelić et al. (2019)	Yes	No	Battery level based on distance driven	Heuristic; Solver (CPLEX)	Heuristic: up to 100; Solver: up to 15;	Minimize distance; Minimize vehicles;	Artificial	ALNS;

Table 7 gives a clear overview of the solutions method for solving EVRPTW. Section 3.6 gives a conclusion on the found literature and makes the connection between the literature and the situation at VSP.

3.6 Conclusion on literature

This chapter investigated on what is found in the literature and gave an overview of what of the found literature will be used in this research. With this literature review, the answers related to the research questions about Modeling techniques, Possible solution methods, and Electric Vehicles are found.

From this literature, it is clear that the VRP problem is a subject that is very well and broadly researched. A lot can be found about this type of problem, therefore a lot of different approaches and formulations can

be found. From the found literature, the formulation given for the VRPTW and EVRPTW is used for modeling the situation of VSP.

Chapter 4 shows a two phase solution approach where the goal is to break down the EVRPTW problem into two smaller sub problems which are solved in two phases, in a similar like fashion as shown in the paper by Lalla-Ruiz & Voß (2020). For the improvement of the current situation in Section 4.1, a VNS/VND approach is applied that is based on the approach by Bac & Erdem (2021). Which uses two neighborhood operators to select a neighbor solution. Section 4.1 will use two types of initial solutions; the current routes of VSP and solutions that are constructed by the parallel construction approach of Chiang & Russell (1996). For Section 4.2, an insertion heuristic is used to insert chargers in the best places with regard to the objective function. Here, the result from Section 4.1 is the initial solution where the chargers are inserted into.

4 Solution approach

Chapter 4 shows the approach used to obtain solutions for the problem that VSP is facing. Section 4.1 shows the structure of the solution approach. Section 4.2 and Section 4.3 explains the approach in detail, where Section 4.2 focuses on Phase 1 of the solution approach and Section 4.3 on Phase 2 of the solution approach.

4.1 Structure approach

To solve the problem VSP is facing, a Two-Phase solution approach is used. First, the current situation (VRPTW) will be improved using a VNS/VND approach. These improved routes are then used as input for solving the new situation (EVRPTW). As the main problem with the new situation is where and how long to charge. The new situation is solved by using the improved routes and inserting electric chargers into these routes and determining how long a vehicle needs to charge at such a charging location. Phase 1 is improving the current situation by using VRPTW techniques and Phase 2 is inserting the charging locations to solve the EVRPTW for VSPs new situation. The insertion of chargers could be done by a heuristic for larger problems and by a solver for smaller problems. This charger insertion by the solver is a combination between metaheuristics and mathematical programming techniques, which in the related literature is denoted as matheuristic (Maniezzo et al., 2010). As shown in the review paper by Corona-Gutiérrez et al. (2022), the decomposition of problems started to have more presence as they provide an attractive combination of metaheuristics and mathematical models by partially solving reduced versions of the problem at hand. The goal here is to reduce the more difficult EVRPTW problem into two less difficult sub-problems, which are solved in the two phases of the proposed solution approach. Figure 6 shows the steps that are performed within the solution approach. Each arrow represents the routes at that moment in the solution approach, where in the different steps techniques are applied.

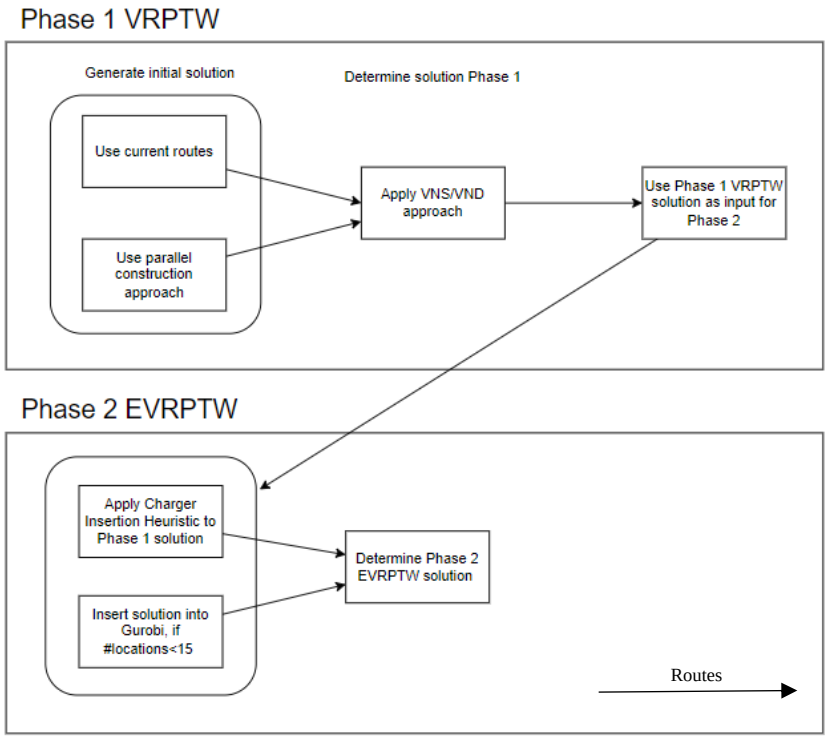

Figure 6. Two-Phase solution approach overview

The idea behind the approach is to split the difficult EVRPTW into two more simple problems. Solving these two simpler problems will then give a solution to the more difficult EVRPTW problem. The remainder of Chapter 4 will explain the approach in more detail.

4.2 Current Situation (Phase 1 VRPTW)

Section 4.2 gives the model and solution approach for VSPs current situation. Section 4.2.1 gives he mathematical formulation. Section 4.2.2 explains the improvement procedure which includes the VNS/VND part, initial solutions, the shaking procedure, and neighborhood operators. Section 4.2.3 gives a conclusion on how the result of this phase can be used as initial routes for solving the EVRPTW problem.

4.2.1 Mathematical formulation VRPTW

The mathematical formulation for the VRPTW problem faced in Phase 1 is the one given in Section 3.2.1 and Section 3.2.2. This is a widely used formulation by Kallehauge et al. (2005).

The model used is for the routes on Tuesday-Friday and Wednesday-Saturday. As the routes on these days have identical structures, only the inputs of the model change with the different locations that are put in for which belong to one of the day sets. The problem is therefore solved twice, once for each combination of days. Furthermore, the parameters are the inputs that are discussed and given in Section 2.2 and Section 2.3.

4.2.2 Solution approach current situation

To solve the problem at hand, an improvement procedure is needed. As this is an NP-hard problem (Lenstra & Kan, 1981), using exact approaches for large problem instances is likely not to be feasible. Therefore, a solver is used for small instances of the problem and a metaheuristic is used to solve the larger real-life problem instances.

Variable Neighborhood Search / Variable Neighborhood Descent

The metaheuristic that is used to be able to optimize the problem is the VNS in combination with a VND local search procedure. Section 3.3.5 explains this approach in more detail. Here, we explain and describe our approach used to be able to improve VSPs current situation.

The idea of the VNS heuristic is that a local search is used to find better neighborhood solutions and when no better solutions can be found, the algorithm ‘shakes’ the solution to be able to possibly escape that local optimum to be able to find better solutions. This ‘shake’ is a randomization of the solution for the algorithm to start using the local search again to find better solutions. It is called a Variable Neighborhood Search because the algorithm will first try to find the best possible solution using a neighborhood operator and it will continue to use that operator until no improvement can be found anymore. After that, it will use the next neighborhood operator until no better solution can be found. After all the operators cannot find an improvement anymore, the solution is shaken and the process starts over again. This is repeated until a stopping criterion is reached. In our case, the stopping criterion is the number of iterations performed by the algorithm.

The Two-Phase solution approach is based on the framework by Bac & Erdem (2021) that use a VNS/VND approach for solving the EVRPTW problem. We based our approach on their VNS/VND approach to solving the VRPTW problem shown in Section 4.1. Figure 7 shows the pseudocode.

```
function VNS(iterations):
        initialsolution = findinitialsolution()
        Bestsolution = initialsolution
        for i in iterations:
                if notfirstiterations:
                        newsolution = shakingprocedure(initialsolution)
                elif firstiteration:
                        newsolution=initialsolution
                newsolution2=VND(newsolution)
                if f(newsolution2)<f(bestsolution):
                        bestsolution=newsolution2
                        initialsolution=newsolution2
                else:
                        initialsolution=newsolution2
        return bestsolution

function VND(newsolution):
        initialsolution = newsolution
        bestsolutionvnd = initialsolution
        kmax=1
        for i in iterations:
                k=0
                while k <= kmax:
                        newsolution=findbestneighbor(initialsolution, k)
                        newsolution2=removeemptytours(newsolution)
                        if f(newsolution2)<f(bestsolutionvnd):
                                bestsolutionvnd = newsolution2
                                initialsolution = newsolution2
                                k=k
                        else:
                                initialsolution = newsolution2
                                k=k+1
        return bestsolutionvnd
```

Figure 7. Pseudocode VNS/VND approach

A pseudocode is a detailed yet readable description of what a computer program or algorithm should do. Both the VNS and VND part are shown, where the VND part is used within the VNS part. As a stopping criterion, the number of iterations is used for both functions. Furthermore, in the VNS part, the shaking procedure is implemented to be able to escape a local optimum. In the VND part, the algorithm goes through the different types of neighborhood operators to find better neighbor solutions.

To be able to start with the metaheuristic, an initial solution is needed. Section 4.2.2 gives two types of initial solutions.

Initial solutions

As input for the VNS/VND, initial solutions are needed. Here, we use two types of initial solutions. One is using a construction approach, called the parallel construction approach (Chiang & Russell, 1996). Section 3.3.1 explains this in full detail. Next to this construction approach, also the current routes that VSP use, are used as an input for the metaheuristic. These routes are given in Section 2.2.

Figure 8 and Figure 9 show the routes for Tuesday-Friday and Wednesday-Saturday respectively that are generated using the parallel construction approach by Chiang & Russell (1996) (Section 3.3.1).

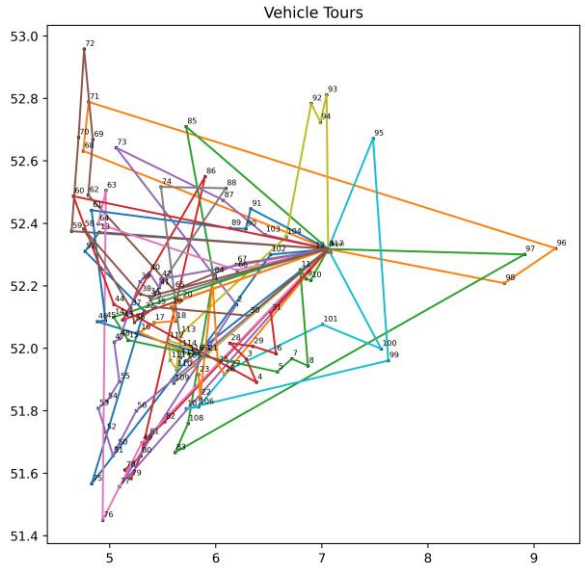

Figure 8. Parallel construction initial solution TUEFRI

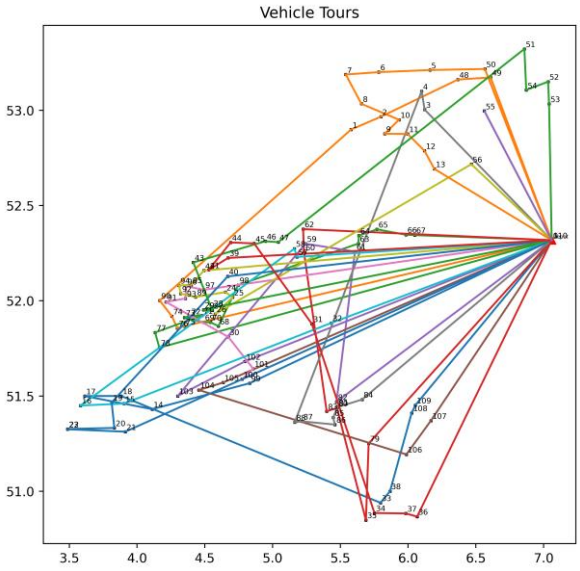

Figure 9. Parallel construction initial solution WEDSAT

As can be seen, the routes look quite random, but the solutions do satisfy the time window constraints as many locations do not have time windows. Furthermore, these initial solutions have more routes than the 12 routes that the company desires, to be able to get a time window feasible solution. This is dealt with in the improvement procedure by removing routes when a route is empty.

Neighborhood operators

To be able to find new solutions, neighborhood operators are used. In this case, we use two operators based on the operators found in Section 3.3.3. These operators are both based on a MOVE operator. Where the first operator picks a random location and places it in the best place in that current solution for the objective function. The second MOVE operator looks at all the locations and tries them separately on all places trying to find the best place considering the objective function and then picks the solution with the move of a single location that corresponds to the biggest improvement in the objective function.

Shaking procedure

The shaking procedure used is a straightforward one. Also, it does not destroy the whole structure of the solution and it considers feasibility in terms of time windows. The procedure picks a random location and searches for a location other than the current one that is feasible. The first place that is found will be picked as a new place for that location. This is done in total three times, so the solution is modified for three different locations in total.

4.2.3 Conclusion Phase 1

With the procedure for solving the VRPTW problem clear, the routes that Phase 1 results in can be used as routes for a normal VRPTW problem. However, Phase 2 will use these routes as an initial solution for solving the EVRPTW problem.

4.3 New situation with electrical vehicles (Phase 2 EVRPTW)

As the EVRPTW problem is a new situation, that is a more extensive problem, first, some new input data is given to complete the data needed for solving the EVRPTW problem (Section 4.3.1). Second, the mathematical model is given (Section 4.3.2). Last, the approach for solving the EVRPTW problem is described (Section 4.3.3).

4.3.1 New input data for Phase 2

As the problem is extended to the EVRPTW problem. New input data is presented to be able to assess the full problem. Charging locations are added to the overview with locations. Furthermore, the travel time matrix is extended with the charging locations. As the number of charging locations in The Netherlands is quite large, the travel time matrix becomes much larger. In total 139 charging locations throughout The Netherlands are considered. These charging locations are based on the charging locations given by the website by FastNed. In addition to the VRPTW problem, here the problem includes charging locations. In total, the problem then includes around 250 locations. Therefore, to be able to solve the problem also all travel times and distances are needed for the locations including charging locations. The times and distances are determined in the manner as in Section 2.3. These can be found in Appendix C and Appendix D.

The extension of the electric vehicle to the VRPTW problem comes with some data about this electric vehicle. Mercedes-Benz has provided data about the newest eSprinter that is released in the near future. This Mercedes can handle the load and has the largest driving range. This is the vehicle that VSP wants to use. Table 8 contains all data regarding the EV.

Table 8. Data Electric Vehicle (eSprinter)

Name Data	**Value**
Driving range	~400km
Battery capacity	113 kWh
Charging speed	10% to 80% within 42 minutes (1.667 kWh per minute charging)
Battery consumption	113 kWh to 0 for 400km (3.5km per kWh, 0.28 kWh per km)

This is the data provided by Mercedes and in Section 4.3.2 within the model formulation these are taken into account within the parameter part. Section 4.3.2 displays the mathematical formulation for the EVRPTW problem.

4.3.2 Mathematical formulation EVRPTW

The mathematical formulation of the EVRPTW is an extension of the VRPTW. Within the formulation, multiple sets, parameters, decision variables, and constraints are added. The formulation given is based on the formulation of Keskin & Çatay (2016). In specific, the model presented is Electric Vehicle Routing Problem With Time Windows Partial Recharging (EVRPTW-PR). We take this problem as in the case of VSP, the overall time of the route is of importance. Therefore, we do not want to waste time recharging fully when this is unnecessary. However, the model presented here does differ from the model from Keskin & Çatay (2016). We consider a set of vehicles to be able to set a maximum amount of vehicles. This is not done by Keskin & Çatay (2016). We include a set for the considered vehicles as is done in the VRPTW formulation by Kallehauge et al. (2005). Therefore, the formulation is a combination of these VRPTW and EVRPTW-PR formulations. Which results in the following sets, parameters, decision variables, and constraints for the EVRPTW-PR problem.

Sets:

C_0 $Set\ of\ customers\ including\ depot\ 0$

C $Set\ of\ customers\ excluding\ depot\ 0\ and\ n$

V $Set\ of\ Electric\ Vehicles$

F $Set\ of\ recharging\ stations$

$N' = N \cup F', \quad where\ set\ F'\ is\ related\ to\ set\ F\ of\ recharging\ stations\ excluding\ depots$

N'_n $Set\ of\ locations\ including\ charging\ stations\ excluding\ depot\ 0$

$N'_{0,n}$ $Set\ of\ locations\ including\ charging\ stations\ including\ depot\ 0\ and\ n$

Parameters:

t_{ij} $Number\ of\ minutes\ it\ takes\ to\ go\ from\ location\ i\ to\ j\ including\ serving\ i$

a_i $The\ time\ after\ which\ the\ driver\ can\ start\ serving\ location\ i$

b_i $The\ time\ before\ which\ the\ driver\ needs\ to\ leave\ location\ i$

d_{ij} $Distance\ between\ location\ i\ and\ location\ j$

h $Constant\ used\ for\ battery\ consumption\ per\ unit\ distance$

Q $Constant\ for\ battery\ capacity$

g $Constant\ for\ charging\ rate$

Decision Variables:

x_{ijk} $\begin{cases} 1, if\ electric\ vehicle\ k\ drives\ directly\ from\ location\ i\ to\ location\ j \\ 0, otherwise \end{cases}$

s_{ik} $The\ time\ at\ which\ electric\ vehicle\ k\ starts\ to\ service\ location\ i$

y_{ik} $remaining\ battery\ level\ on\ arrival\ at\ a\ next\ customer\ location\ i\ for\ vehicle\ k$

Y_{ik} $battery\ state\ of\ charge\ on\ departure\ from\ recharging\ station\ i\ for\ vehicle\ k$

Objective Function:

$$min \sum_{k\in V} \sum_{i\in N'_0} \sum_{j\in N'_n} t_{ij} x_{ijk},\ if\ i \neq j \tag{1}$$

Constraints:

$$\sum_{k\in V} \sum_{j\in N'\ if\ j\neq i} x_{ijk} = 1, \qquad \forall i \in C, \tag{2}$$

$$\sum_{j\in N} x_{0jk} \leq 1, \qquad \forall k \in V, \tag{3}$$

$$\sum_{i\in N'_0} x_{ihk} - \sum_{j\in N'_n} x_{hjk} = 0, \qquad \forall h \in N', \forall k \in V, \tag{4}$$

$$s_{ik} + t_{ij} - M_{ij}(1 - x_{ijk}) \leq s_{jk}, \qquad \forall i \in C_0, \forall j \in N'_n, \forall k \in V, i \neq j \tag{5}$$

$$a_i \leq s_{ik} \leq b_i \qquad \forall i \in C, \forall k \in V, \tag{6}$$

$$0 \leq y_{jk} \leq y_{ik} - hd_{ij}x_{ijk} + Q(1 - x_{ijk}), \qquad \forall j \in N'_n, \forall i \in C, i \neq j, \forall k \in V, \tag{7}$$

$$0 \leq y_{jk} \leq Y_{ik} - hd_{ij}x_{ijk} + Q(1 - x_{ijk}), \qquad \forall i \in F_0', \forall j \in N_n', i \neq j, \forall k \in V, \tag{8}$$

$$s_{ik} + t_{ij}x_{ijk} + g(Y_i - y_i) - (M_{ij} + gQ)(1 - x_{ijk}) \leq s_{jk}, \quad \forall i \in F', \forall j \in N_n', i \neq j, \forall k \in V \tag{9}$$

$$y_{ik} \leq Y_{ik} \leq Q, \qquad \forall i \in F_0', \tag{10}$$

In the formulation above, M_{ij} are large constants that can be decreased to $\max\{b_i + t_{ij} - a_j\}, (i,j) \in N$. (1) is the objective function, which minimizes the total distance of all tours. Constraint (2) ensures that every customer is visited exactly once. Constraint (3) is used to give every vehicle the ability to leave the depot. Constraint (4) makes every vehicle leave a location once it enters a location. Constraint (5) keeps track of the arrival times at the locations. It ensures that the arrival time of the next location is later than the current location. Constraint (6) considers the time windows of location i. Constraints (7) and (8) ensure the battery levels are updated per visit to a charging station or customer location. These constraints also ensure that the battery level does not go below zero. Constraint (9) considers charging times and makes sure that when a vehicle leaves a charging station the arrival at the next location is after the arrival at the charging station. Constraint (10) gives the boundaries for the state of charge on departure from recharging station i.

The model given is again solved two times. For both Tuesday-Friday and Wednesday-Saturday, the problem is solved. The problem for both sets is identical in terms of structure. Only the inputs change per day set.

4.3.3 Solution approach new situation

To solve the EVRPTW problem, we have split the problem into two phases. In Section 4.1, we explain these two phases in more detail. Section 4.3.3 will show how the result of Section 4.2 is used to be able to solve the EVRPTW problem given in 4.3.2.

To be able to account for the driving range restriction given by the battery of an EV. We are inserting charging locations in the best position to the found routes for solving the VRPTW problem of phase 1. This is done using an insertion heuristic. The idea of the insertion heuristic is to see at which location the battery is too low to be able to reach the next location. For that location, we need to find the charging location that has the lowest cost of insertion for the location the vehicle is at and the next location in the tour.

Figure 10 is an example of how a charging location is added using an insertion heuristic. In Figure 10, the white arrows show how the route would be driven by a normal vehicle. In other words, this is a solution for the normal VRP(TW) problem. In the case, the vehicle is electric, the battery would be empty before the end of the route as can be seen by the status of the battery that is updated per location in Figure 10. When the EV is arriving at Location 4, the battery is too empty to be able to reach Location 5. Therefore, a recharging location needs to be added. The two nearest recharging stations are Recharging Station 1 and Recharging Station 2. In this case, the insertion of Recharging Station 2 is more favorable than the insertion of Recharging Station 1 as the cost of insertion is lower. Therefore, a solution for the EVRP(TW) is the route given including the green arrows going by Recharging Station 2. For this approach to work the assumption is that recharging locations are well-distributed over the area of the locations. Otherwise, the insertion of a recharging location at the point the EVs battery is almost empty would give the situation that it would not be able to reach the next charger as well. Section 4.3.1 shows that for VSPs situation The Netherlands has a good distribution of recharging stations throughout the country.

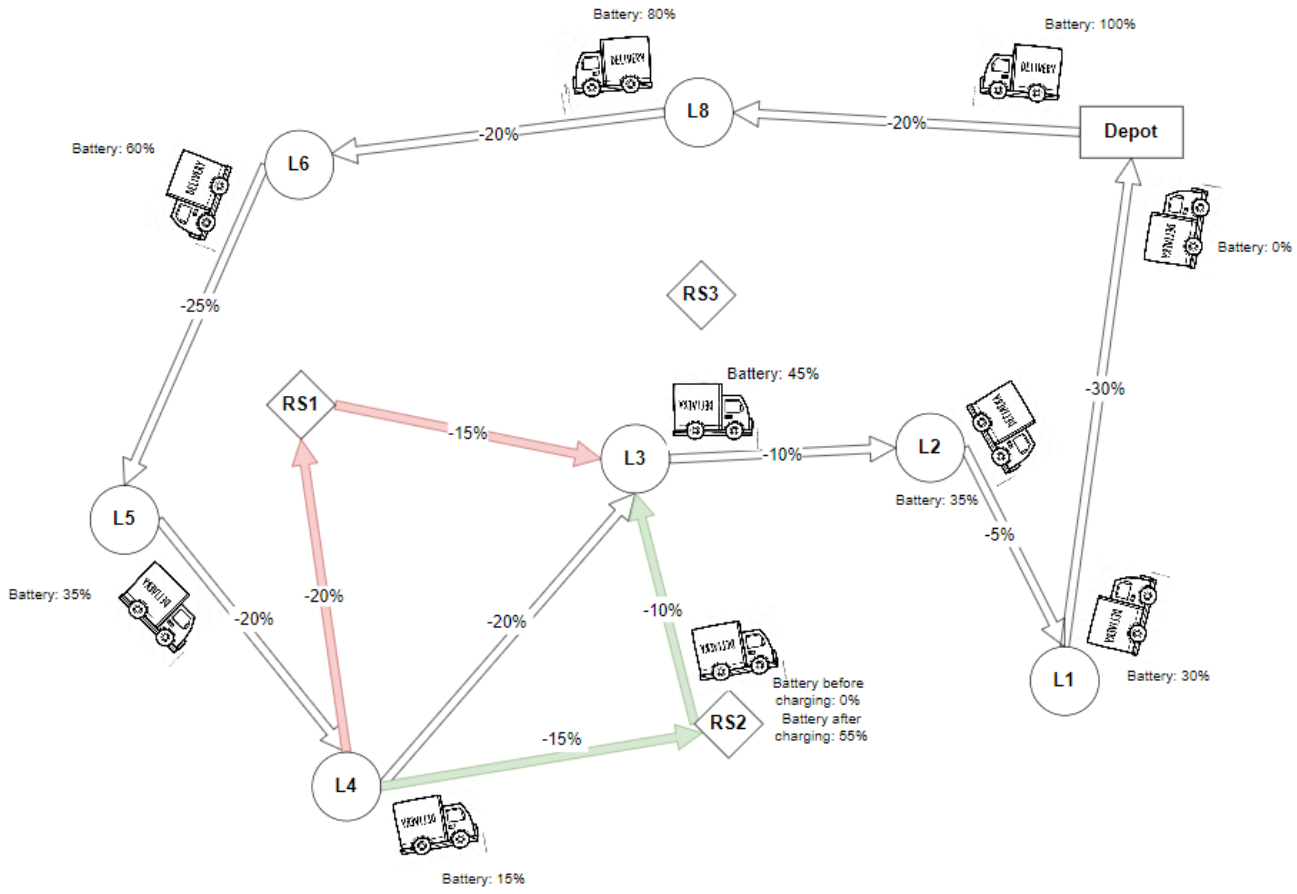

Figure 10. Example insertion of charging location for a single tour

As the remaining distance of the route to the ending depot is known. The charging decision at Recharging Station 2 is to charge the amount to be able to complete the route. This is considered a partial charging strategy.

The mathematical formulation in Section 4.3.2 is how this problem is described and how it can be implemented into for example a solver. When using a (meta)heuristic, it is more difficult to check for feasibility considering applicable constraints. Therefore, when using the heuristic to solve the problem. A different objective function is used to be able to check the feasibility of a solution. Constraint (1) gives the objective function that is based on the one that Bac & Erdem (2021) use to check feasibility.

$$min \sum_{k\in V} \sum_{i\in N} \sum_{j\in N} c_{ij} x_{ijk} + \sum_{i\in N} \theta_i + \sum_{k\in V} O_k \qquad (1)$$

As given in the literature review, two variables are added, which are θ_i and O_k, which are the deviation of the time windows and the overtime of a route respectively. Furthermore, Bac & Erdem (2021) add two constraints to keep track of these penalties. These constraints are (2) and (3). Constraints (4) and (5) and nonnegativity constraints. These constraints are easier to use when implementing a (meta)heuristic than the formulation given in Section 4.3.2.

$$(\sum_{k\in V} s_{ik} + H_i x_{ijk}) - b_i \leq \theta_i, \qquad \forall i \in N, \qquad (2)$$

$$O_k \geq s_{nk} - T_M, \qquad \forall k \in V, \qquad (3)$$

$$\theta_i \geq 0, \qquad \forall i \in N, \qquad (4)$$

$$O_k \geq 0, \qquad \forall k \in V, \qquad (5)$$

Next to applying the charger insertion heuristic, it is also possible to use the solution from the VNS/VND heuristic as input for a solver. This is called a matheuristic and for instances up to 15 locations, it can help speed up the solving process. The structure of the 2-Phase solution approach remains the same.

4.3.4 Conclusion Phase 2

With the insertion of chargers into the routes where necessary and the ability to check for feasibility, the solution approach for solving the EVRPTW problem is completed. The solution for VRPTW found in Phase 1 is extended to be a solution for the EVRPTW in Phase 2. Chapter 5 displays numerical experiments to show the performance of the solution approach.

5 Experiments

Chapter 5 investigates the performance of the proposed solution method by comparing results. Furthermore, various experiments are performed to see what influences solution quality and performance. Table 9 gives an overview of the section in Chapter 5.

Table 9. Overview of experiments

Section	Description
Section 5.1:	*Description experiments and overview*
Section 5.2:	*Benchmark instances*
- Section 5.2.1	Small benchmark instances (5-10-15 locations)
- Section 5.2.2	Large benchmark instances (100 locations)
Section 5.3:	*Randomly generated real-life instances*
- Section 5.3.1	Small size real-life instances (5-10-15 locations)
- Section 5.3.2	Medium size real-life instances (40-50-60 locations)
Section 5.4:	*Results of the company (VSP)*
- Section 5.4.1	Tuesday-Friday
- Section 5.4.2	Wednesday-Saturday
Section 5.5:	*Conclusion on experiments*

5.1 Description experiments and overview

Section 5.1 gives descriptions of the comparisons and experiments performed in Section 5.2, Section 5.3, and Section 5.4. Section 5.2 uses benchmark instances given by the paper of Schneider et al. (2014). Section 5.3 uses randomly generated instances. These instances are sampled from the real-life instances of the company (VSP). These instances are further explained within the given sections itself.

Comparison

With the comparisons, the goal is to identify the performance of the solution approach. In the different sections, the method is tested and compared in various manners. For smaller instances up to 15 locations, the comparison is done with the model in a solver (Gurobi). If the solver is not able to find the optimal solution within 3600 seconds, the MIP gap given by Gurobi is reported. The MIP gap is determined by the following formula in Gurobi (Miltenberger, 2023):

$$Gap = \frac{|\text{ObjBound} - \text{ObjVal}|}{|\text{ObjVal}|}$$

This formula gives the difference between the Objective lower bound and the best found heuristic feasible solution. These two come closer to each other when the problem is optimized by Gurobi. When they are the same, the found solution is optimal. However, when Gurobi is not done within the given time limit, this gap can indicate if the found feasible is close to a possible optimal solution. For the smaller instances, the comparison also includes the gaps between the different solution approaches, as well as the solving time and number of used vehicles.

For the larger instances, the solver is not able to find feasible solutions within a reasonable time, therefore for these instances, the approach is compared with the LP relaxation of the problem. This LP relaxation is calculated using the model in Gurobi, but the binary decision variables that determine the routes, are relaxed

to a continuous variable between 0 and 1. Furthermore, the comparison for all instances reports and evaluates the number of vehicles used and solving time in seconds.

Parameter selection
To be able to perform experiments, first, the parameters need to be determined based on what can influence the solution provided by the solution approach when these parameters are tuned. We conclude that a different objective function might influence the solutions as sometimes faster routes in terms of time have longer distances in terms of kilometers.

Furthermore, the expectation is that when driving range is increased by either increasing battery capacity or improving battery usage, solutions should improve. Next to that, charging speed can influence the total time of the routes and also the arrival time at a location after charging. Which could give more options in terms of feasibility considering time windows.

Experiment type 1: Objective function
The first experiment is done to see how the solution is influenced by the objective function that is used. Two objective functions are used:

- Total distance
- Total time

The goal is to identify the differences in the solutions and find the causes of these differences. Experiment type 1 is done in Section 5.2.1, Section 5.2.2, Section 5.3.1, and Section 5.3.2.

Experiment type 2: Input settings
With the second type of experiment, the goal is to determine what the influence of the various input parameters is. The varied input settings are the charging speed at a charging location, the battery capacity of the vehicle, and the battery usage of the vehicle. Here, the charging speed is the amount of unit distance charged in one unit of time. The battery usage is the amount of battery usage for one unit distance. An overview of the input settings is as follows:

- Experiment 2.1: Capacity
- Experiment 2.2: Charging speed
- Experiment 2.3: Battery usage

Experiment type 2 is done in Section 5.2.1, Section 5.2.2, Section 5.3.1, and Section 5.3.2.

Experiment type 3: Charging strategy
The third experiment is to see what kind of influence the charging strategy has on the determined solutions. Here, two charging strategies are used. The first strategy is a partial charging strategy (Section 4.3.3). The second strategy is a fully charging strategy, where every time a vehicle visits a charging station the battery of the vehicle is fully charged.

- Partial charging strategy
- Fully charging strategy

The goal is to see how the charging strategy influences the quality and performance of the solutions. Experiment type 3 is done in Section 5.3.1.

Section 5.1 shows what kind of experiments will be performed. Next, Section 5.2 will show the results for the benchmark instances and Section 5.3 will give the results for the real-life instances. Last, Section 5.4 gives the results determined for VSP.

5.2 Benchmark instances

Section 5.2 will show the performance of the solution approach compared to a solver (Gurobi) and Matheuristic. Next to that, multiple experiments are performed to get an idea of the influence of the input settings (Section 5.1).

5.2.1 Small size instances

Section 5.2.1 contains small size instances varying in the number of customer locations. The instances either contain 5, 10, or 15 locations. These instances are based on the instances from Schneider et al. (2014), but modified as the problem instances in that paper also consider load and capacity. These constraints are not considered in the situation of VSP and therefore also not in the solution approach. Therefore, the found solutions are not compared with the results reported by Schneider et al. (2014). Table 10 displays which experiments are done. Furthermore, various features and settings are displayed in Table 10. In the comparison, 36 instances are considered. With the experiments, 12 instances are used. The reason for this is time restrictions. Furthermore, the settings that are given are based on the input settings given by Schneider et al. (2014). In bold, the aspect that is changed or compared is highlighted. This gives an easy overview of what is done within the various experiments. The capacity given in Table 10, is the battery capacity of the electric vehicle. Furthermore, the velocity that is given, is the amount of distances unit travelled in one time unit.

Table 10. Overview Small size benchmark experiments

	Section 5.2.1 (Small size benchmark instances)				
	#Instances	Approaches	Objective function	Settings	Charging strategy
Comparison	36	**Gurobi vs Matheuristic vs VNS/VND**	Total distance	Velocity = 1, Capacity = (77.75, 60.63)* Charging speed = (3.49, 0.49, 0.39)* Battery usage = 1	Partial charging strategy
Experiment 1	12	Gurobi	**Total distance vs Total time**	Velocity = 2, Capacity = 77.75, Charging speed = 3.49, Battery usage = 1	Partial charging strategy
Experiment 2.1	12	Gurobi	Total distance	Velocity = 1, **Capacity = 70 vs 90** Charging speed = 3.49, Battery usage = 1	Partial charging strategy
Experiment 2.2	12	Gurobi	Total distance	Velocity = 1, Capacity = 77.75 **Charging speed = 2.5 vs 5,** Battery usage = 1	Partial charging strategy
Experiment 2.3	12	Gurobi	Total distance	Velocity = 1, Capacity = 77.75 Charging speed = 3.49, **Battery usage = 0.9 vs 1.1**	Partial charging strategy

**Input settings depending on which instances, varying between given inputs*

For the comparison in performance, Gurobi is used to obtain the optimal solution if possible within a reasonable time. The other approaches are compared with the found Gurobi solution. Table 11 contains the comparison, where the number of vehicles, objective value, solving time, and gap are reported.

Gurobi is able to find the optimal solution for almost every instance. However, Gurobi does difficulties with the problems containing 15 customer locations. Here, it is visible that multiple times gaps are reported around 10%-15%. As the time limit was set to an hour, it is unclear whether Gurobi would have found the optimal solution within a reasonable time.

As the matheuristic, uses Gurobi in the second part of solving the problem instances, the reported objective values are almost identical. However, for instance c103C15, the matheuristic is performing considerably worse than Gurobi. From the solutions, it is unclear why this is exactly the case, as both approaches used Gurobi in the end. Furthermore, the goal of the matheuristic is to improve solving time by giving an initial solution to Gurobi. From the 36 instances, this type of improvement concerning solving time happened 5 times.

VNS/VND-CIH can find the same solution as Gurobi on 5 occasions. Next to that, for 6 instances the result is within 5% of the Gurobi solution. Another 6 instances are within a 5%-10% difference compared to the Gurobi solution. The rest of the instances are all above a 10% difference, where it seems the solution approach sometimes seems to have difficulties with overcoming a local optimum where it would have been beneficial to add an extra vehicle to the solution. The large reported differences are all having fewer vehicles than the Gurobi solution. As there are no load capacity and demand constraints, some of these instances can be solved with fewer vehicles to give a feasible outcome. However, the objective value could have been improved using more vehicles, but therefore the VNS/VND needs to overcome these local optima. In some cases, it can do that and it some cases, it is not able to do that. However, the VNS/VND is no ran for 10 iterations, where if this number is increased, the chance of finding better solutions is increasing. Here, that is not considered due to time restrictions.

Table 11. Small size benchmark instances Comparison

	Solver (Gurobi)					VNS/VND-Solver (Matheuristic)			VNS/VND-CIH (Heuristic)			
Instances: C#locations	#vehicles	Obj.	LP relaxation	Time(s)	MIPgap (%)	#vehicles	ΔObj%	Time(s)	#vehicles	ΔObj%	ΔLPrelax%	Time(s)
c101C10.txt	4	393.56	145.08	60.47	0.00%	4	0.00%	51.89	4	0.00%	54.73%	68.85
c101C5.txt	4	247.15	129.45	0.62	0.00%	4	0.00%	2.78	4	0.00%	47.13%	22.43
c103C15.txt	3	265.81	168.14	3600.90	13.37%	4	39.73%	3606.77	2	27.00%	53.83%	112.94
c103C5.txt	3	165.67	104.89	1.64	0.00%	3	0.00%	2.35	1	30.84%	56.21%	19.58
c104C10.txt	2	273.93	167.30	19.60	0.00%	4	0.00%	28.62	1	17.27%	49.47%	54.37
c106C15.txt	3	275.13	105.72	12.41	0.00%	3	0.00%	30.90	2	34.08%	74.67%	120.22
c202C10.txt	2	243.20	124.57	7.30	0.00%	2	0.00%	10.45	1	27.75%	62.99%	37.20
c202C15.txt	3	369.56	197.14	137.23	0.00%	3	0.00%	167.40	1	4.03%	48.81%	102.31
c205C10.txt	2	228.28	114.17	2.01	0.00%	2	0.00%	3.80	2	21.85%	60.91%	62.59
c206C5.txt	3	236.58	122.82	1.42	0.00%	3	0.00%	2.46	1	8.17%	52.32%	24.72
c208C15.txt	2	300.55	168.76	60.20	0.00%	3	0.00%	49.71	1	23.45%	57.01%	115.60
c208C5.txt	1	158.48	76.14	0.56	0.00%	1	0.00%	3.29	1	6.57%	55.11%	21.53
r102C10.txt	3	249.19	129.66	9.60	0.00%	3	0.00%	14.56	2	4.17%	50.14%	46.83
r102C15.txt	5	418.80	162.94	3601.23	8.95%	6	0.09%	3604.98	5	0.00%	57.81%	149.61
r103C10.txt	3	202.85	105.77	490.90	0.00%	3	0.00%	497.08	1	12.27%	54.26%	67.29
r104C5.txt	2	136.69	85.09	1.03	0.00%	2	0.00%	1.43	1	26.22%	54.07%	25.78
r105C15.txt	4	336.15	154.48	175.34	0.00%	5	0.00%	92.17	1	8.39%	57.90%	114.96
r105C5.txt	2	156.08	107.36	0.55	0.00%	2	0.00%	1.82	1	20.35%	45.21%	22.06
r201C10.txt	3	217.68	137.54	8.44	0.00%	4	0.00%	7.29	2	23.68%	51.78%	61.26
r202C15.txt	4	358.00	168.46	3600.98	9.42%	4	0.00%	3604.46	1	13.61%	59.35%	124.69
r202C5.txt	1	128.78	95.84	2.71	0.00%	1	0.00%	5.67	1	10.98%	33.75%	22.47
r203C10.txt	1	218.21	121.14	41.20	0.00%	1	0.00%	44.37	1	24.37%	58.01%	59.71
r203C5.txt	1	179.06	115.99	2.85	0.00%	1	0.00%	4.28	1	11.13%	42.43%	23.53
r209C15.txt	2	293.20	173.10	414.32	0.00%	2	0.00%	453.14	1	5.66%	44.30%	124.67
rc102C10.txt	4	423.51	215.56	5.57	0.00%	4	0.00%	24.08	2	2.38%	50.31%	50.07
rc103C15.txt	4	394.65	155.86	3600.86	17.44%	4	0.00%	3606.23	3	19.57%	68.24%	130.30
rc105C5.txt	3	238.05	105.41	4.46	0.00%	3	0.00%	3.95	3	0.00%	52.42%	21.05
rc108C10.txt	3	345.93	197.82	19.37	0.00%	3	0.00%	20.34	2	6.14%	46.33%	57.20
rc108C15.txt	4	371.40	160.10	3601.12	16.02%	4	0.00%	3604.88	2	15.40%	63.53%	131.34
rc108C5.txt	3	253.93	128.61	4.29	0.00%	3	0.00%	4.58	1	2.96%	50.86%	17.79
rc201C10.txt	3	310.06	135.57	4.89	0.00%	3	0.00%	7.42	2	0.24%	56.38%	65.43
rc202C15.txt	4	405.53	160.92	205.12	0.00%	5	0.00%	173.63	1	4.95%	62.28%	140.14
rc204C15.txt	2	310.58	200.96	3601.33	12.52%	2	0.00%	3604.74	1	0.54%	35.65%	125.17
rc204C5.txt	1	176.39	84.23	5.30	0.00%	1	0.00%	5.48	1	26.24%	64.78%	25.79
rc205C10.txt	2	325.98	169.88	7.42	0.00%	2	0.00%	9.27	1	32.65%	64.90%	64.38
rc208C5.txt	1	167.98	83.84	1.78	0.00%	1	0.00%	1.92	1	0.94%	50.56%	25.85
Average	2.69	271.57	138.34	647.64	2.16%	3.14	1.11%	648.84	1.92	13.16%	54.12%	68.33

Experiment 1: Objective function

Table 12 displays the results of using a different objective function for solving the problems at hand. Here, it is visible that the solutions will come out the same. The values in terms of distance and number of vehicles are identical, however, solving time does differ for multiple instances. These differences in solving time are also not consistent for one of the two objective functions used. Furthermore, when considering total time, on average fewer vehicles are used. An expected outcome here could have been that by optimizing in terms of time, routes could go by longer routes in terms of distances that are quicker to ride. However, that is not what happened with these problem instances.

Table 12. Small size benchmark instances Experiment 1: Total Distance vs Total time

Objective:	**Total Distance (Gurobi)**				**Total time (Gurobi)**		
Instances: C#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101C10.txt	4	393.56	19.97	0.00%	4	0%	37.98
c101C5.txt	3	247.15	0.45	0.00%	3	0%	1.29
c103C15.txt	3	350.01	3600.73	7.00%	3	0%	3601.15
c103C5.txt	3	165.67	1.50	0.00%	3	0%	1.23
c104C10.txt	4	273.93	19.91	0.00%	3	0%	20.16
c106C15.txt	3	271.21	10.73	0.00%	3	0%	15.17
c202C10.txt	2	243.20	4.55	0.00%	2	0%	4.62
c202C15.txt	5	369.56	125.35	0.00%	3	0%	89.73
c205C10.txt	2	228.28	2.19	0.00%	2	0%	1.23
c206C5.txt	3	236.58	1.16	0.00%	3	0%	0.87
c208C15.txt	2	300.55	24.71	0.00%	4	0%	18.38
c208C5.txt	1	158.48	0.84	0.00%	1	0%	0.87
Average	*2.92*	*269.85*	*317.67*	*0.58%*	*2.83*	*0.00%*	*316.06*

Experiment 2.1: Capacity

Table 13 gives the results for Experiment 2.1 for the small size benchmark instances. The expectation is that fewer vehicles are needed when increasing the capacity. Furthermore, the distance can go lower because when a vehicle has more capacity, less charging is needed. This will reduce the total distance when charging stations can be skipped. Table 13 shows that these expected observations are indeed happening. Most of the problem instances show less total distance and less number of vehicles. When capacity is reduced, Gurobi also was not able to find the optimal solution on 2 occasions, where on one occasion the problem even became infeasible (indicated with (-) in Table 13).

It is clear that more battery capacity results in better solutions as fewer charges are needed and therefore the problem becomes more like the normal VRPTW problem, where finding feasible solutions is much easier. The total distance or time of the solutions can be reduced due to fewer visits to charging stations, where driving distance is saved, next to also saving charging time at these charging locations.

Table 13. Small size benchmark instances Experiment 2.1: Capacity

Capacity:	**Capacity-70 (Gurobi)**				**Capacity-90 (Gurobi)**			
Instances: C#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Gap (%)*	*Time (s)*
c101C10.txt	4	402.15	118.54	0.00%	3	-4.46%	0.00%	16.70
c101C5.txt	4	250.04	0.85	0.00%	3	-6.09%	0.00%	0.56
c103C15.txt	3	374.12	3600.86	13.19%	3	-8.09%	0.00%	2310.93
c103C5.txt	3	165.73	1.31	0.00%	2	-3.52%	0.00%	0.60
c104C10.txt	3	273.93	11.35	0.00%	2	-1.00%	0.00%	24.61
c106C15.txt	4	323.34	61.19	0.00%	3	-16.12%	0.00%	7.64
c202C10.txt	2	251.95	3.39	0.00%	4	-3.47%	0.00%	3.90
c202C15.txt	3	375.90	261.49	0.00%	3	-2.58%	0.00%	384.25
c205C10.txt	-	-	-	-	2	-	0.00%	1.77
c206C5.txt	4	236.58	0.89	0.00%	2	-6.17%	0.00%	1.31
c208C15.txt	2	300.55	9.79	0.00%	3	-0.71%	0.00%	47.37
c208C5.txt	1	164.34	0.66	0.00%	2	-3.74%	0.00%	0.74
Average	*3.00*	*283.51*	*370.03*	*1.20%*	*2.67*	*-5.09%*	*0.00%*	*233.37*

Experiment 2.2: Charging speed

Table 14 shows the results for Experiment 2.2 on the small size benchmark instances. Here, the goal is to see what the influence of charging speed is on the objective value. The comparison includes Charging speeds 2.5 and 5.0. This means that one distance unit is charged in 2.5 or 5.0 time units. The expectation is that the faster the charging speed, the better the performance also in time of distances and number of vehicles. The results also show that this is the case. The reasoning behind this is that when less time is needed to charge, time windows constraints that are infeasible with slower charging speed become feasible due to charging time savings. For this reason, the number of needed vehicles can then also be reduced. However, in many cases, the solution is the same for both charging speeds and when there is a difference in the total distance of the routes, these differences are not that large.

Table 14. Small size benchmark instances Experiment 2.2: Charging speed

Charging speed:	**Charging speed-2.5 (Gurobi)**				**Charging speed-5.0 (Gurobi)**		
Instances: C#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101C10.txt	3	392.36	43.38	0.00%	4	0.31%	34.47
c101C5.txt	3	247.15	1.05	0.00%	3	0.00%	0.79
c103C15.txt	2	350.00	3601.28	6.86%	4	6.16%	3600.89
c103C5.txt	3	165.67	1.39	0.00%	3	0.00%	1.97
c104C10.txt	2	273.93	29.79	0.00%	2	2.19%	35.13
c106C15.txt	3	271.21	7.05	0.00%	4	7.50%	12.49
c202C10.txt	2	243.20	5.42	0.00%	2	0.00%	3.48
c202C15.txt	3	369.56	91.68	0.00%	4	1.96%	274.43
c205C10.txt	2	228.28	2.00	0.00%	2	0.00%	1.54
c206C5.txt	2	236.21	1.36	0.00%	3	0.16%	1.37
c208C15.txt	2	300.55	14.43	0.00%	2	0.00%	19.39
c208C5.txt	1	158.48	0.65	0.00%	1	0.00%	0.59
Average	*2.33*	*269.72*	*316.62*	*0.57%*	*2.83*	*1.52%*	*332.21*

Experiment 2.3: Battery usage

Table 15 gives the results of Experiment 2.3. Different amounts of battery usage per distance unit are compared. This is comparable to increasing the battery capacity, nevertheless, these experiments are performed to ensure the results are the same. It is possible that soon, one of the two options will be cheaper to realize for the car manufacturer and therefore driving range could be extended by either increasing the battery efficiency or battery capacity.

Table 15 displays results that are indeed comparable with Experiment 2.1. The total distance of the found solutions is less in most instances. Next to that, the more efficient battery can use fewer vehicles due to the larger driving range. Again, the reasoning behind this is that fewer charges are needed which reduces distances to possible charging locations. Next to that, the time at these charging locations is not needed when fewer charges are done, leading to a shorter total time.

Table 15. Small size benchmark instances Experiment 2.3: Battery usage

Battery usage:	**Battery usage-0.9 (Gurobi)**				**Battery usage-1.1 (Gurobi)**		
Instances: C#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101C10.txt	3	384.81	29.25	0.00%	4	2.27%	68.96
c101C5.txt	3	245.42	1.05	0.00%	4	1.88%	1.05
c103C15.txt	2	343.84	3600.80	4.04%	2	8.81%	3600.74
c103C5.txt	2	159.90	1.12	0.00%	3	3.65%	3.19
c104C10.txt	3	273.93	13.62	0.00%	3	0.00%	51.63
c106C15.txt	3	271.21	3.52	0.00%	4	19.22%	35.44
c202C10.txt	2	243.20	2.13	0.00%	2	3.59%	9.51
c202C15.txt	3	368.05	92.54	0.00%	4	2.13%	222.50
c205C10.txt	2	228.28	2.26	0.00%	-	-	-
c206C5.txt	2	221.98	0.83	0.00%	3	6.58%	6.28
c208C15.txt	3	298.41	15.64	0.00%	2	0.72%	139.57
c208C5.txt	1	158.20	0.56	0.00%	1	3.88%	4.75
Average	*2.42*	*266.44*	*313.61*	*0.34%*	*2.91*	*4.79%*	*376.69*

5.2.2 Large size instances

This section shows a comparison and the experiments, however, this time the comparison is done between the VNS/VND-CIH and the LP relaxation solution as Gurobi is not able to solve large instances for the EVPRTW. Next to that, the experiments are all performed with the VNS/VND-CIH approach. Table 16 gives the experiments done in Section 5.2.2. Here, also the input settings are given based on the benchmark instances by Schneider et al. (2014).

Table 16. Overview of large size benchmark Experiments

	Section 5.2.2 (Large size benchmark instances)				
	#Instances	Approaches	Objective function	Settings	Charging strategy
Comparison	26	**VNS/VND vs LP relaxation**	Total distance	Velocity = 1, Capacity = (79.69, 117.66, 66.28)* Charging speed = (3.39, 2.29, 0.45)* Battery usage = 1	Partial charging strategy
Experiment 1	9	VNS/VND	**Total distance vs Total time**	**Velocity = 2,** Capacity = 77.75, Charging speed = 3.49, Battery usage = 1	Partial charging strategy
Experiment 2.1	9	VNS/VND	Total time	Velocity = 1, **Capacity = 70 vs 90** Charging speed = 3.39, Battery usage = 1	Partial charging strategy
Experiment 2.2	9	VNS/VND	Total time	Velocity = 1, Capacity = 77.75 **Charging speed = 2.5 vs 5,** Battery usage = 1	Partial charging strategy
Experiment 2.3	9	VNS/VND	Total time	Velocity = 1, Capacity = 77.75 Charging speed = 3.39, **Battery usage = 0.9 vs 1.1**	Partial charging strategy

**Input settings depending on which instances, varying between given inputs*

Table 17 shows the overview of the comparison between the found solution of VNS/VND-CIH and the LP relaxation solution. The gap is around 50%-75% for most instances. Comparing these percentages to the gaps for the smaller instances (Table 11) shows that in general, these percentages are a bit larger. However, it does not deviate a lot from the smaller instances. Taking into account that the gaps found for the smaller instances are mostly within in 25% gap of the optimal solution, the performance from the large instances can be considered okay. An advantage of this approach is that it is able to find a solution overnight, whereas a solver would never be able to do that. Therefore, a lot of time saving is done and solutions can be found relatively quickly. Furthermore, the solutions can improve when the algorithm is run longer.

Table 17. Large size benchmark instances comparison

	LP relaxation (Gurobi)	**VNS/VND-CIH**			
Instances	*Obj.(Distance)*	*#vehicles*	*Obj. (Distance)*	*ΔLPrelax%*	*Time (s)*
c101_21.txt	308.66	8	1232.66	74.96%	5691.84
c102_21.txt	308.66	6	1164.63	73.50%	14529.72
c103_21.txt	308.61	6	931.42	66.87%	11458.62
c104_21.txt	308.58	5	1104.93	72.07%	8118.86
c105_21.txt	308.57	7	1260.56	75.52%	8307.83
c106_21.txt	308.57	6	1055.56	70.77%	6384.28
c107_21.txt	308.57	10	1297.61	76.22%	5732.23
c108_21.txt	308.57	7	990.22	68.84%	7761.02
c109_21.txt	308.57	7	1140.47	72.94%	10638.24
c201_21.txt	460.38	6	1090.79	57.79%	3598.02
c202_21.txt	460.38	5	1218.76	62.23%	7347.88
c203_21.txt	460.38	1	1071.27	57.03%	3228.91
c204_21.txt	460.38	1	851.50	45.93%	2610.84
c205_21.txt	460.38	3	1112.22	58.61%	2765.77
c206_21.txt	460.38	4	1037.94	55.65%	5469.42
c207_21.txt	460.38	2	972.21	52.65%	4127.60
c208_21.txt	460.38	4	1056.65	56.43%	5592.57
r101_21.txt	566.09	16	1796.85	68.50%	8152.04
r102_21.txt	566.09	13	1676.87	66.24%	7674.49
r103_21.txt	566.09	8	1274.27	55.58%	3178.34
r104_21.txt	566.09	3	1073.43	47.26%	4062.73
r105_21.txt	566.09	9	1449.99	60.96%	2959.12
r106_21.txt	566.09	10	1477.71	61.69%	5655.38
r107_21.txt	566.09	6	1213.72	53.36%	3549.06
r108_21.txt	566.09	3	1036.29	45.37%	2898.15
r109_21.txt	566.09	7	1343.97	57.88%	4163.65
Average	*444.43*	*6.27*	*1189.71*	*62.11%*	*5986.79*

Experiment 3: Objective function

Table 18 displays the influence of a different objective function. Here, it seems that when using Total time as an objective function the solutions perform better in terms of total distance when comparing the solutions to optimizing in terms of Total distance. However, when the total time performs worse, it performs way worse. It seems that it gets easier stuck in worse local optima and when Total distance optimization gets stuck in local optima, it already found a better solution compared to when optimizing on Total Time gets stuck in local optima. It is hard to say or not running the algorithm for more iterations could benefit one of the two objective functions more than the other, however, running more iterations will almost certainly in both cases increase the quality of the found solution. Nevertheless, due to time restrictions, the algorithm is run for only 5 iterations.

Table 18. Large size benchmark instances Experiment 1: Total distance vs total time

Objective:	**Total Distance (VNS/VND-CIH)**			**Total Time (VNS/VND-CIH)**		
Instances	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101_21.txt	8	1559.51	4979.55	8	-0.30%	8829.44
c102_21.txt	9	1546.03	3160.75	9	35.76%	1702.49
c103_21.txt	5	1302.24	6212.53	5	-5.53%	8411.88
c104_21.txt	6	1156.90	4541.13	6	-10.49%	9364.43
c105_21.txt	8	1340.32	2620.30	9	84.95%	1414.86
c106_21.txt	11	1562.58	2150.41	8	-20.89%	6672.96
c107_21.txt	10	1383.48	2559.86	11	40.45%	1359.59
c108_21.txt	6	1190.43	2373.82	7	77.59%	1293.92
c109_21.txt	8	1288.04	3006.46	8	-18.59%	8862.67
Average	*7.89*	*1369.95*	*3511.65*	*7.89*	*20.33%*	*5323.58*

Experiment 4.1: Capacity

Table 19 gives an overview of Experiment 2.1 for large size benchmark instances. Here, the expectation is again that more capacity leads to better solutions. With 6 out of the 9 instances, this is the case. For instance c102_21, it seems that the algorithm got stuck in a local optima as the solution is a lot worse than the less capacity solution and the solving time is a lot shorter. The shorter solving time is an indication that no improvements could be found anymore as the algorithm would have continued if it was improving. Nevertheless, the expectation to find better solutions with more capacity is achieved in most cases.

Table 19. Large size benchmark instances Experiment 2.1: Capacity

Capacity:	**Capacity-70 (VNS/VND-CIH)**			**Capacity-90 (VNS/VND-CIH)**		
Instances	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101_21.txt	8	1232.26	5219.64	8	1.10%	4948.81
c102_21.txt	6	1143.39	11084.97	9	115.71%	602.82
c103_21.txt	6	969.60	9848.69	6	-1.30%	10486.13
c104_21.txt	5	1144.17	7099.26	5	-4.93%	7101.02
c105_21.txt	7	1297.38	7548.76	9	47.82%	575.84
c106_21.txt	9	1844.94	548.63	6	-43.46%	5933.32
c107_21.txt	10	1321.24	3766.22	10	-1.57%	3710.11
c108_21.txt	8	1074.29	5567.88	8	-4.12%	6223.88
c109_21.txt	7	1177.65	7438.70	7	-0.78%	7700.30
Average	*7.33*	*1244.99*	*6458.08*	*7.56*	*12.05%*	*5253.58*

Experiment 4.2: Charging speed

Table 20 shows the results of Experiment 2.2 for large size benchmark instances. Here, the results are not as expected. The expectation is that the faster charging time would lead to better results. However, the results show that charging speed does not seem to be a big problem when optimizing in terms of distance for larger instances. The time windows allow for comparable solutions, even when longer stops need to be made for the extra charging time.

Table 20. Large size benchmark instances Experiment 2.2: Charging speed

Charging speed:	**Charging speed-2.5 (VNS/VND-CIH)**			**Charging speed-5.0 (VNS/VND-CIH)**		
Instances	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101_21.txt	8	1243.31	5345.77	8	0.00%	4543.70
c102_21.txt	6	1164.06	13570.03	6	0.00%	13382.30
c103_21.txt	6	935.46	12804.53	6	0.00%	12776.16
c104_21.txt	5	1110.32	7463.32	5	0.00%	7724.80
c105_21.txt	7	1245.40	8336.70	7	0.00%	8155.35
c106_21.txt	6	1032.39	6428.33	6	0.00%	6132.73
c107_21.txt	10	1325.64	4068.84	10	0.17%	4205.56
c108_21.txt	8	1031.11	6353.12	7	-4.59%	8753.36
c109_21.txt	7	1182.56	7413.33	7	0.00%	7345.39
Average	*7.00*	*1141.14*	*7976.00*	*6.89*	*-0.49%*	*8113.26*

Experiment 4.3: Battery usage

Table 21 shows that in general when battery usage is higher and thus battery efficiency is lower, the solutions are more likely to be worse. This is in line with the results seen for Experiment 2.1 for large size benchmark instances. Where the reasoning, that more driving range leads to fewer charge visits and less charging time still can be applied. Nevertheless, for some of these larger instances, the solutions for both the amounts of battery usage yield the same objective value. With instance c102_21 for battery usage 1.1, the algorithm seems to get stuck into a local optimum, as the solution is a lot worse than battery usage 0.9. This can be derived from the solving time. The heuristic will have a lower running time because as long as the heuristic can keep improving the found solution, it will continue. In the case, the heuristic is not able to escape the local optimum with the shaking procedure, it will be done by improving the solution relatively quickly and the total solving time will be a lot faster. This faster running time as a result of fewer improvements, results in a worse overall solution.

Table 21. Large size benchmark instances Experiment 2.3: Battery usage

Battery usage:	**Battery usage-0.9 (VNS/VND-CIH)**			**Battery usage-1.1 (VNS/VND-CIH)**		
Instances	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
c101_21.txt	8	1243.31	5085.33	8	0.00%	5192.987
c102_21.txt	6	1164.06	15286.77	6	-9.95%	17134.66
c103_21.txt	6	935.46	10830.53	9	138.95%	590.5741
c104_21.txt	5	1110.32	7424.56	5	0.00%	7756.334
c105_21.txt	7	1245.40	8195.27	9	53.40%	573.1777
c106_21.txt	6	1032.39	6099.70	6	0.00%	6090.414
c107_21.txt	10	1325.64	4058.50	10	0.00%	4064.435
c108_21.txt	7	970.91	7549.59	8	6.20%	6720.405
c109_21.txt	7	1182.56	7133.45	6	-3.57%	10804.97
Average	*6.89*	*1134.45*	*7962.63*	*7.44*	*20.56%*	*6547.55*

After evaluating the results from the comparisons and experiments for small and large benchmark instances, the following can be concluded. Creating a larger driving range by increasing capacity or improving battery

efficiency results in better solutions in general. Charging speed seems to play a smaller role in the outcome of the solutions.

Next to the experiments, the benchmark instances show that the heuristic can be performed quite okay. On average 13% gap with the solution given by Gurobi using a one-hour time limit. However, the solving time of the heuristic is on average a lot faster (Table 11).

Section 5.3 continues with comparisons and experiments on instances based on real-life data from VSP.

5.3 Generated real-life cases

This section performs an analysis of cases based on real-life data. Here, the instances are randomly generated from the data that is available from VSP. Section 5.3.1 will perform experiments (Section 5.1) on small instances containing 5, 10, or 15 locations (Appendix G). Each of these amounts has 5 instances, which results in a total of 15 instances. Section 5.3.2 contains an analysis of medium size instances. Here, the instances again are randomly generated based on the data from VSP. The reason medium size instances are used is that there are around 115 locations in the dataset from VSP. If 100 locations would be sampled, all instances would look similar. Therefore 40, 50, or 60 locations are sampled with each having 5 instances, resulting in 15 medium size instances (Appendix H).

5.3.1 Small size instances

This section evaluates performance and experiments in the same manner as done in Section 5.2.1. Additionally, the goal is to see if, from these different instances, the same conclusion can be drawn. Table 22 gives an overview of the experiments done in Section 5.3.1 with their settings and features. Here, the settings are based on the real-life situation of VSP. Therefore, the battery capacity is 380, as this stands for the driving range in km of the used Mercedes eSprinter. The other input settings are also based on the data that is used for VSPs situation.

Table 22. Overview Small size real-life experiments

	Section 5.3.1 (Small size real-life instances)				
	#Instances	Approaches	Objective function	Settings	Charging strategy
Comparison	15	**Gurobi vs Matheuristic vs VNS/VND**	Total time	Velocity = 1, Capacity = 380 Charging speed = 0.1667 Battery usage = 1	Partial charging strategy
Experiment 1	15	Gurobi	**Total distance vs Total time**	Velocity = 1, Capacity = 380 Charging speed = 0.1667 Battery usage = 1	Partial charging strategy
Experiment 2.1	15	Gurobi	Total time	Velocity = 1, **Capacity = 250 vs 500** Charging speed = 0.1667, Battery usage = 1	Partial charging strategy
Experiment 2.2	15	Gurobi	Total time	Velocity = 1, Capacity = 380 **Charging speed = 0.1 vs 0.2,** Battery usage = 1	Partial charging strategy
Experiment 2.3	15	Gurobi	Total time	Velocity = 1, Capacity = 380 Charging speed = 0.1667, **Battery usage = 0.5 vs 1.5**	Partial charging strategy
Experiment 3	15	Gurobi	Total time	Velocity = 1, Capacity = 380 Charging speed = 0.1667 Battery usage = 1	**Partial charging strategy vs Fully Charging strategy**

Table 23 shows the comparison between the Gurobi, matheuristic, and VNS/VND-CIH. First off, it is visible that when the number of locations are increasing the running time and gap are increasing. Even Gurobi reports gaps between 10% and 30% when running for one hour. Especially when 15 locations are considered, the running time of Gurobi reaches the time limit and the optimal solution is not found. This occurred in Section 5.2.1 in the same manner and confirms that these problems become increasingly more difficult when locations are added.

The matheuristic shows a similar performance as Gurobi. However, 2 times Gurobi found a better solution and 1 time the matheuristic found a better solution. In all these cases, these solutions were not the optimal solution. Furthermore, in two cases the matheuristic found the optimal solution faster than Gurobi. Nevertheless, the goal of trying to improve solving time by giving Gurobi an initial solution does not work as consistently as expected.

The VNS/VND-CIH solution approach shows promising results compared to Gurobi. Most instances are within a 10% difference with the Gurobi solution. However, as can be seen, the gap reported by Gurobi is around 15%-30% for the 10 and 15 location instances. This means that the gap for the VNS/VND-CIH solution is even bigger. The biggest advantage of the VNS/VND-CIH is the running time. These are constant and do not increase a lot when larger problems are being optimized.

The results are comparable with the results in Section 5.2.1. Taking into account that Gurobi displays some larger gaps and the VNS/VND-CIH smaller gaps, the difference with the possible optimal seems to be the same.

Table 23. Small size real-life instances Comparison

	Solver (Gurobi)					**VNS/VND-Solver (Gurobi)**			**VNS/VND-CIH**			
Instances c#locations	*#vehicles*	*Obj.*	*LP relaxation*	*Time(s)*	*MIPgap (%)*	*#vehicles*	*ΔObj %*	*Time (s)*	*#vehicles*	*ΔObj%*	*ΔLPrelax %*	*Time (s)*
Instance 1c5	1	696.00	596.00	0.65	0.00%	1	0.00%	1.56	1	0.14%	14.49%	5.71
Instance 2c5	1	468.00	311.00	0.96	0.00%	1	0.00%	2.22	1	13.49%	42.51%	6.36
Instance 3c5	2	666.00	514.00	1.04	0.00%	2	0.00%	3.51	1	4.31%	26.15%	9.40
Instance 4c5	1	587.00	435.00	0.60	0.00%	1	0.00%	3.13	1	5.17%	29.73%	9.90
Instance 5c5	1	393.00	305.00	0.96	0.00%	1	0.00%	3.04	1	0.00%	22.39%	6.76
Instance 6c10	1	679.00	477.00	395.41	0.00%	1	0.00%	444.95	1	1.74%	30.97%	6.50
Instance 7c10	1	773.00	570.00	3600.70	8.80%	1	0.00%	3602.97	1	3.25%	28.66%	4.87
Instance 8c10	2	1061.00	618.00	3601.65	18.47%	2	0.00%	3603.88	3	10.16%	47.67%	5.09
Instance 9c10	1	590.00	458.00	91.03	0.00%	1	0.00%	61.70	1	1.34%	23.41%	5.02
Instance 10c10	1	683.00	466.00	3600.83	13.76%	1	0.00%	3603.05	1	0.00%	31.77%	5.45
Instance 11c15	3	1469.00	819.00	3606.93	33.22%	3	0.95%	3605.83	3	10.26%	49.97%	9.65
Instance 12c15	2	1281.00	795.00	3601.52	22.17%	2	0.00%	3605.37	2	7.38%	42.52%	9.11
Instance 13c15	2	1146.00	822.00	3601.62	14.57%	2	1.57%	3610.73	2	12.05%	36.91%	9.15
Instance 14c15	2	1161.00	788.00	3601.82	18.43%	2	-3.27%	3606.44	2	0.09%	32.19%	9.56
Instance 15c15	3	1510.00	897.00	3601.77	31.26%	3	0.00%	3606.48	3	-5.82%	37.14%	9.93
Average	*1.60*	*877.53*	*591.40*	*1953.8*	*10.71%*	*1.60*	*-0.05%*	*1957.66*	*1.60*	*4.24%*	*33.10%*	*7.50*

Experiment 5: Objective function

Table 24 gives the results of Experiment 1 for the small size real-life instances. The results show that for instances with 5 locations, both objective functions are capable of finding the optimal solution. When the amount of locations is increased to 10, the performance of both objective functions is similar but a MIPgap is reported by Gurobi. When 15 locations are considered, the two objective functions are

displaying different results. The reported MIP gaps are considerable and for each instance, one of the two objective functions outperforms the other.

Table 24. Small size real-life instances Experiment 1: Total distance vs total time

Objective:	**Total Distance (Gurobi)**				**Total time (Gurobi)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c5	1	643.00	0.68	0.00%	1	0.2%	1.80
Instance 2c5	1	491.00	0.71	0.00%	1	0.0%	2.89
Instance 3c5	2	774.00	1.11	0.00%	2	0.1%	1.75
Instance 4c5	1	548.00	0.77	0.00%	1	0.0%	0.79
Instance 5c5	1	357.00	0.94	0.00%	1	0.3%	1.39
Instance 6c10	1	487.00	236.85	0.00%	1	0.0%	371.36
Instance 7c10	1	661.00	3600.85	15.28%	1	0.0%	3600.79
Instance 8c10	2	904.00	3600.78	26.99%	2	0.0%	3601.04
Instance 9c10	1	470.00	15.85	0.00%	1	0.2%	49.41
Instance 10c10	1	522.00	3600.75	24.33%	1	0.9%	3600.89
Instance 11c15	3	1144.00	3602.11	55.86%	3	1.0%	3607.48
Instance 12c15	2	1047.00	3601.71	31.14%	2	-17.8%	3609.23
Instance 13c15	2	988.00	3601.49	32.19%	2	-11.1%	3601.89
Instance 14c15	2	887.00	3601.70	20.74%	2	6.0%	3601.98
Instance 15c15	3	1043.00	3602.06	46.02%	3	11.6%	3601.87
Average	*1.60*	*731.07*	*1937.89*	*16.84%*	*1.60*	*-0.57%*	*1950.30*

Experiment 6.1: Capacity

Table 25 displays the outcome of Experiment 2.1 for the small size real-life instances. The expectation is that more capacity leads to better solutions as less charging is needed. When less charging is needed, the vehicles save time not driving to charging stations as well as saving time charging at these stations.

Table 25 shows that more capacity indeed confirms this expectation. The solutions where more capacity is available are all better except for the last instances. It is unclear, why in this case the lower capacity is better than having more capacity as having less capacity should also be an option for having more capacity as in both situations a partial charging strategy is used. It could be that the solution space for having a smaller capacity is smaller and Gurobi can find better solutions quicker, whereas the solution space for having more capacity is larger, and therefore Gurobi could have more difficulty going through this larger space results in a worse solution in the given time limit.

Table 25. Small size real-life instances Experiment 2.1: Capacity

Capacity:	**Capacity-250 (Gurobi)**				**Capacity-500 (Gurobi)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c5	-	-	-	-	1	0% (MIPgap)	1.42
Instance 2c5	1	556.00	3.68	0.00%	1	-13.24%	1.95
Instance 3c5	-	-	-	-	1	0% (MIPgap)	2.21
Instance 4c5	2	608.00	1.66	0.00%	1	-10.95%	1.43
Instance 5c5	1	372.00	2.66	0.00%	1	-3.91%	1.30
Instance 6c10	1	554.00	1469.92	0.00%	1	-13.99%	264.29
Instance 7c10	1	718.00	3602.80	12.18%	1	-8.62%	3602.46
Instance 8c10	2	1070.00	3602.97	27.75%	2	-21.87%	3602.41
Instance 9c10	1	488.00	62.98	0.00%	1	-4.72%	117.41
Instance 10c10	1	540.00	3601.49	15.42%	1	-3.25%	3604.74
Instance 11c15	3	1215.00	3609.34	35.88%	3	-5.84%	3614.11
Instance 12c15	2	1084.00	3605.41	19.29%	2	-7.75%	3610.32
Instance 13c15	2	1013.00	3605.55	23.30%	2	-8.69%	3608.61
Instance 14c15	2	1052.00	3606.66	20.70%	2	-27.67%	3606.62
Instance 15c15	2	1019.00	3603.67	24.13%	3	4.77%	3609.00
Average	*1.62*	*791.46*	*2336.83*	*13.74%*	*1.53*	*-9.67%*	*1949.89*

Experiment 6.2: Charging speed

Table 26 shows the outcome of Experiment 2.2 for the small size real-life instances. It shows that for the instances up to 10 locations no substantial differences can be seen. However, the instances containing 15 locations, show different results. With these instances, the expected outcome is seen. Which is that a faster charging speed (0.1 time unit for charging one distance unit) results in better solutions in general. However, for two instances, the slower charging speed outperforms the faster speed. The results in general correspond with the outcome of Experiment 2.2 in Section 5.2.1.

Table 26. Small size real-life instances Experiment 2.2: Charging speed

Charging speed:	**Charging speed-0.1 (Gurobi)**				**Charging speed-0.2 (Gurobi)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c5	1	644.00	2.54	0.0%	1	0.00%	2.56
Instance 2c5	1	491.00	7.55	0.0%	1	0.20%	3.12
Instance 3c5	2	775.00	3.45	0.0%	2	0.00%	3.91
Instance 4c5	1	548.00	2.55	0.0%	1	0.00%	2.43
Instance 5c5	1	358.00	2.87	0.0%	1	0.00%	3.05
Instance 6c10	1	487.00	1391.67	0.0%	1	0.00%	439.31
Instance 7c10	1	661.00	3606.04	8.9%	1	0.00%	3604.07
Instance 8c10	2	904.00	3602.86	18.9%	2	0.00%	3603.90
Instance 9c10	1	471.00	250.59	0.0%	1	0.00%	90.04
Instance 10c10	1	527.00	3604.15	13.3%	1	0.00%	3604.22
Instance 11c15	3	1170.00	3604.85	34.3%	3	1.35%	3606.23
Instance 12c15	3	1142.00	3605.39	23.8%	3	-1.96%	3603.86
Instance 13c15	2	903.00	3604.69	16.8%	2	10.33%	3608.00
Instance 14c15	2	875.00	3613.37	12.1%	2	10.90%	3615.27
Instance 15c15	3	1047.00	3604.46	27.4%	2	-8.50%	3604.80
Average	*1.67*	*733.53*	*2033.80*	*10.37%*	*1.60*	*0.82%*	*1959.65*

Experiment 6.3: Battery usage

Table 27 gives the results for Experiment 2.3 for small size real-life instances. Lower battery usage should result in better solutions, as more driving range is achieved when the battery uses less charge when driving the same amount of distance. However, the results do not present these expectations immediately. For the instances with 5 locations, the lower battery usage (0.9) achieves better solutions when solving to optimality. When the number of locations is increased, both situations have trouble finding optimal solutions and the reported MIP gaps increase. When 15 locations are considered, these gaps enlarge and no battery usage is consistently better than the other.

Table 27. Small size real-life instances Experiment 2.3: Battery usage

Battery usage:	**Battery usage-0.9 (Gurobi)**				**Battery usage-1.1 (Gurobi)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c5	1	644.00	5.62	0.00%	1	8.65%	6.26
Instance 2c5	1	491.00	8.66	0.00%	1	0.20%	9.38
Instance 3c5	1	778.00	9.51	0.00%	1	0.00%	8.17
Instance 4c5	1	548.00	4.87	0.00%	1	0.00%	5.14
Instance 5c5	1	358.00	9.85	0.00%	1	3.76%	5.14
Instance 6c10	1	487.00	2399.61	0.00%	1	1.22%	3452.29
Instance 7c10	1	661.00	3605.08	9.95%	1	0.00%	3604.47
Instance 8c10	2	904.00	3602.66	14.89%	2	5.44%	3604.76
Instance 9c10	1	471.00	71.98	0.00%	1	0.00%	71.26
Instance 10c10	1	527.00	3604.57	14.06%	1	0.00%	3602.47
Instance 11c15	3	1156.00	3606.59	33.70%	3	12.16%	3605.67
Instance 12c15	2	1077.00	3606.13	20.86%	2	-0.37%	3612.94
Instance 13c15	2	979.00	3606.44	18.61%	2	-8.42%	3610.76
Instance 14c15	2	951.00	3614.91	17.00%	2	-8.07%	3605.76
Instance 15c15	2	909.00	3604.31	20.09%	2	17.06%	3605.07
Average	*1.47*	*729.40*	*2090.72*	*9.94%*	*1.47*	*2.11%*	*2160.64*

Experiment 7: Partial charging strategy vs Fully charging strategy

Table 28 gives the results for Experiment 3, where the goal is to see whether charging strategies influence the quality of the solutions. In terms of distances for the first 10 instances, it is visible that there is no difference between the two strategies. The extra charging times do not conflict with the time window constraints, resulting in the same performance of the routes. When considering 15 locations, the two strategies do differ in terms of performance. It shows that using a full charging strategy can outperform the partial charging strategy within the given time limit. However, with the partial charging strategy, fully charging is also an option, therefore the solution found by the fully charging strategy should be achievable by the partial charging strategy. The reason for this result could be the same as for Experiment 2.1, where having less capacity leads to a smaller solution space where Gurobi can maybe find better solutions within this space in the given time limit. Where having more capacity or in this case more charging options, leads to a bigger solution space which gives the model more difficulty finding the same solution within the same time frame.

Table 28. Small size real-life Experiment 3

Charging strategy:	**Partial charging strategy (Gurobi)**				**Fully charging strategy (Gurobi)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*MIPGap (%)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c5	1	696	0.65	0.00%	1	0.00%	1.53
Instance 2c5	1	468	0.96	0.00%	1	0.00%	1.44
Instance 3c5	2	666	1.04	0.00%	2	0.00%	1.68
Instance 4c5	1	587	0.60	0.00%	1	0.00%	2.43
Instance 5c5	1	393	0.96	0.00%	1	0.00%	1.66
Instance 6c10	1	679	395.41	0.00%	1	0.00%	56.79
Instance 7c10	1	773	3600.70	8.80%	1	0.00%	2648.54
Instance 8c10	2	1061	3601.65	18.47%	2	0.00%	3602.74
Instance 9c10	1	590	91.03	0.00%	1	0.00%	25.77
Instance 10c10	1	683	3600.83	13.76%	1	0.00%	3602.12
Instance 11c15	3	1469	3606.93	33.22%	3	0.94%	3606.25
Instance 12c15	2	1281	3601.52	22.17%	2	-6.13%	3608.34
Instance 13c15	2	1146	3601.62	14.57%	2	0.35%	3605.01
Instance 14c15	2	1161	3601.82	18.43%	2	3.09%	3606.60
Instance 15c15	3	1510	3601.77	31.26%	2	-12.52%	3607.79
Average	*1.60*	*877.53*	*1953.83*	*10.71%*	*1.53*	*-0.95%*	*1865.25*

5.3.2 Medium size instances

This section considers medium size instances that are randomly generated from the VSP data. The instances consider 5, 10, and 15 locations, where for each amount of locations 5 instances are sampled (Appendix H). Section 5.3.2 shows the results of the experiments given in Table 29.

Table 29. Overview Medium size real-life experiments

	Section 5.3.2 (Medium size real-life instances)				
	#Instances	Approaches	Objective function	Settings	Charging strategy
Comparison	15	**VNS/VND vs LP relaxation**	Total distance	Velocity = 1, Capacity = 380 Charging speed = 0.1667 Battery usage = 1	Partial charging strategy
Experiment 1	15	VNS/VND	**Total distance vs Total time**	Velocity = 1, Capacity = 380 Charging speed = 0.1667 Battery usage = 1	Partial charging strategy
Experiment 2.1	15	VNS/VND	Total time	Velocity = 1, **Capacity = 250 vs 500** Charging speed = 0.1667, Battery usage = 1	Partial charging strategy
Experiment 2.2	15	VNS/VND	Total time	Velocity = 1, Capacity = 380 **Charging speed = 0.1 vs 0.2,** Battery usage = 1	Partial charging strategy
Experiment 2.3	15	VNS/VND	Total time	Velocity = 1, Capacity = 380 Charging speed = 0.1667, **Battery usage = 0.5 vs 1.5**	Partial charging strategy

Table 30 shows the comparison between the LP relaxation and the best-found heuristic solution for the medium size real-life instances. The gaps reported lie around 60%-70%. To put these numbers into perspective, these numbers are compared to the gaps with the LP relaxation for the small size real-life instances (Table 23). It shows that the gaps reported for the medium size instances are around 30% larger than the gaps reported for the small size instances. Some of these solutions for the smaller instances are close to optimality, but others also still report a MIP gap in a range from 10% to 30%. This would indicate that these solutions for the medium size real-life instances are not great. However, these solutions could possibly be improved if the algorithm is run for more iterations. When a company wants to ensure that their solutions improve, the iterations can be increased. This will increase solving time. That is the tradeoff the company has to consider.

Table 30. Medium size real-life instances Comparison

	LP relaxation (Gurobi)	**VNS/VND-CIH**			
Instances c#locations	*Obj.(Distance)*	*#vehicles*	*Obj. (Distance)*	*ΔLPrelax% (Distance)*	*Time (s)*
Instance 1c40	584	4	2040	71.37%	84.49
Instance 2c40	864	5	2139	59.61%	92.79
Instance 3c40	759	4	2072	63.37%	89.50
Instance 4c40	906	3	2293	60.49%	71.37
Instance 5c40	760	5	2176	65.07%	95.20
Instance 6c50	887	4	2221	60.06%	117.10
Instance 7c50	1066	4	2946	63.82%	133.23
Instance 8c50	782	5	2320	66.29%	143.64
Instance 9c50	1016	4	2359	56.93%	114.47
Instance 10c50	877	4	2448	64.17%	86.96
Instance 11c60	1082	5	2839	61.89%	335.21
Instance 12c60	1115	5	2889	61.41%	299.84
Instance 13c60	731	5	2764	73.55%	194.41
Instance 14c60	1003	5	2754	63.58%	246.70
Instance 15c60	936	5	2440	61.64%	156.68
Average	*891.20*	*4.47*	*2446.67*	*63.55%*	*150.77*

Experiment 8: Objective function

Table 31 shows that when considering two different types of objective functions, optimizing in terms of total time outperforms optimizing in terms of total distance. Especially when 10 or 15 locations are considered, the solution of optimizing for total time seems to perform better. Looking at Experiment 1 for other instances, it is visible that this is not always the case. There, one of the two objective functions is not outperforming the other consistently. With these randomly generated instances, especially when the number of locations is increasing, times and distances can differ a lot. When a lot of locations within the Randstad region are considered, times can increase a lot when distances are not that big. Therefore, the importance of some connections between locations increases when optimizing is done in terms of time. This can lead to better solutions. However, when looking at the smaller instances, this behavior is not seen. Therefore, it is hard to conclude what influences the performance to be better for optimizing based on total time.

Table 31. Medium size real-life instances Experiment 1: Total distance vs total time

Objective:	**Total Distance (VNS/VND-CIH)**			**Total Time (VNS/VND-CIH)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c40	4	1903.00	76.97	4	-0.05%	74.75
Instance 2c40	3	2063.00	60.55	5	4.76%	68.77
Instance 3c40	4	2238.00	50.72	4	-11.62%	59.59
Instance 4c40	4	1995.00	51.05	3	9.24%	49.23
Instance 5c40	4	2134.00	51.81	4	5.49%	71.40
Instance 6c50	5	2417.00	104.45	5	0.45%	112.69
Instance 7c50	4	2911.00	65.71	4	-0.76%	98.57
Instance 8c50	5	2701.00	93.79	5	-4.25%	120.95
Instance 9c50	4	2450.00	121.17	4	-4.43%	94.71
Instance 10c50	4	2427.00	105.28	4	-0.58%	76.51
Instance 11c60	5	2787.00	171.79	6	9.84%	173.46
Instance 12c60	5	2520.00	114.97	5	7.79%	164.73
Instance 13c60	6	3194.00	106.82	6	-9.38%	170.77
Instance 14c60	5	2887.00	148.65	5	-11.38%	167.57
Instance 15c60	5	2571.00	136.55	5	-3.71%	122.41
Average	*4.47*	*2479.87*	*97.35*	*4.60*	*-0.57%*	*108.41*

Experiment 9.1: Capacity

Table 32 displays the results of Experiment 2.1 for medium size real-life instances. The expectation is that more capacity leads to better solutions. Here, in general, that is the case. Where having a bigger capacity leads to better solutions. From the 15 instances considered 10 instances have better solutions when having more capacity. This is in line with the results found for Experiment 2.1 in Section 5.2.1, Section 5.2.2, and Section 5.3.1.

Table 32. Medium size real-life instances Experiment 2.1: Capacity

Capacity:	**Capacity-250 (VNS/VND-CIH)**			**Capacity-500 (VNS/VND-CIH)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c40	4	2007.00	105.30	4	-7.33%	112.5603
Instance 2c40	4	2229.00	114.12	4	-1.78%	102.0815
Instance 3c40	4	2188.00	165.36	4	-7.73%	120.6476
Instance 4c40	4	2085.00	164.77	4	-18.00%	125.9284
Instance 5c40	3	2061.00	147.95	4	4.54%	134.2486
Instance 6c50	5	2484.00	250.59	5	-1.47%	343.7665
Instance 7c50	4	3015.00	252.57	4	-12.71%	259.4468
Instance 8c50	4	2584.00	282.70	5	-24.89%	291.3974
Instance 9c50	4	2497.00	280.30	4	3.18%	250.7028
Instance 10c50	5	2574.00	255.43	5	-2.59%	189.265
Instance 11c60	6	2969.00	406.67	6	-3.74%	304.6982
Instance 12c60	5	2657.00	242.50	5	1.56%	277.7513
Instance 13c60	4	2635.00	343.69	6	9.64%	260.6532
Instance 14c60	5	2702.00	421.95	6	6.63%	324.5281
Instance 15c60	5	2667.00	371.78	5	-5.21%	323.5711
Average	*4.40*	*2490.27*	*253.71*	*4.73*	*-3.99%*	*228.08*

Experiment 9.2: Charging speed

Table 33 shows the results of Experiment 2.2 for medium size real-life instances. Section 5.3.1 indicates that having a faster charging speed should result in better solutions. However, that behavior is not completely visible here. A lot of solutions lay close to each other in terms of performance. Needless to say, having a faster charging speed does have a slightly better performance, however, it is not as convincing as for the other instances where Experiment 2.2 is considered.

Table 33. Medium size real-life instances Experiment 2.2: Charging speed

Charging speed:	**Charging speed-0.1 (VNS/VND-CIH)**			**Charging speed-0.2 (VNS/VND-CIH)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c40	4	1997.00	142.70	4	-3.47%	152.95
Instance 2c40	3	1990.00	153.46	5	5.46%	138.17
Instance 3c40	3	1969.00	107.32	4	7.08%	132.72
Instance 4c40	4	2116.00	180.92	3	1.12%	89.00
Instance 5c40	4	2270.00	148.85	4	1.05%	91.93
Instance 6c50	4	2333.00	202.98	5	1.85%	205.20
Instance 7c50	4	2683.00	139.63	4	2.75%	127.01
Instance 8c50	5	2493.00	145.69	5	-9.63%	223.89
Instance 9c50	4	2547.00	141.78	4	-19.19%	116.41
Instance 10c50	4	2507.00	111.51	5	0.91%	140.27
Instance 11c60	5	3042.00	186.38	5	-4.54%	213.27
Instance 12c60	5	2485.00	220.42	5	6.79%	229.81
Instance 13c60	6	2746.00	186.81	5	2.45%	145.97
Instance 14c60	5	2740.00	221.56	5	-6.86%	223.48
Instance 15c60	5	2484.00	235.88	5	14.93%	169.66
Average	*4.33*	*2426.80*	*168.39*	*4.53*	*0.05%*	*159.98*

Experiment 9.3: Battery usage

Table 34 shows the results of Experiment 2.3 for medium size real-life instances. The general expectation of having more driving range resulting in better solutions is not confirmed here. The extra driving range is here created due to having a more efficient battery does not seem to lead to better solutions. It is unclear what the cause of this situation is. The solving times for considering a worse battery efficiency are not that much lower, which does not indicate that the algorithm got stuck in a local optimum. However, the performance of the two battery usages does lay close for many instances. Still, the expected outcome is not visible in these results.

Table 34. Medium size real-life instances Experiment 2.3: Battery usage

Battery usage:	**Battery usage-0.9 (VNS/VND-CIH)**			**Battery usage-1.1 (VNS/VND-CIH)**		
Instances c#locations	*#vehicles*	*Obj.(Distance)*	*Time (s)*	*#vehicles*	*ΔObj% (Distance)*	*Time (s)*
Instance 1c40	4	1923.00	87.45	4	-2.67%	85.00
Instance 2c40	4	1829.00	109.28	4	21.54%	121.28
Instance 3c40	4	2165.00	113.38	4	-8.47%	106.98
Instance 4c40	4	2338.00	115.65	3	-14.50%	76.33
Instance 5c40	4	2215.00	128.41	4	-1.61%	93.88
Instance 6c50	5	2651.00	227.62	4	-17.51%	180.18
Instance 7c50	4	2965.00	235.56	4	-8.81%	216.65
Instance 8c50	4	2237.00	262.45	4	2.23%	152.10
Instance 9c50	4	2440.00	187.24	4	3.25%	187.02
Instance 10c50	5	2905.00	284.61	4	-11.52%	322.38
Instance 11c60	5	2937.00	231.60	6	-5.19%	433.53
Instance 12c60	5	2622.00	230.25	5	-5.43%	326.68
Instance 13c60	5	2813.00	244.21	6	1.99%	294.55
Instance 14c60	5	2632.00	313.91	5	-2.02%	380.97
Instance 15c60	5	2400.00	393.32	5	1.80%	355.31
Average	*4.47*	*2471.47*	*211.00*	*4.40*	*-3.13%*	*222.19*

Having performed the experiments and comparisons for the real-life instances, the following can be concluded. As with the benchmark instances, it shows that increasing driving range establishes better solutions. Next to that, *Experiment 3: Charging strategy* shows that both charging strategies perform quite the same.

Table 23 shows that the performance of the heuristic is quite good with an average gap to the solution by Gurobi of around 4% with a lot faster solving time.

5.4 Results of the company (VSP)

The goal of this research is to solve the EVRPTW for the new situation of VSPs routing logistics. Section 5.4 displays the best-found solution using the proposed solution approach. The solutions given are solutions for the EVRTPW problem on Tuesday-Friday (Section 5.4.2) and Wednesday-Saturday (Section 5.4.2).

5.4.1 Tuesday-Friday

Error! Reference source not found. displays the routes for Tuesday-Friday of VSP routing logistics. The blue triangles depict a charging location. The black dots are normal locations. As can be seen, not a lot of routes need charging during the route. This is caused by the total distance of the routes. When the length of routes does not exceed the capacity of the vehicles, which is 380 kilometers for the Mercedes eSprinter, then charging during the routes is not needed. The route can remain the same as what it was or in this case the optimized route from Phase 1.

Table 35 gives a comparison between the current situation and the new routes. Chapter 2 gives a full analysis of the current situation. Here, the times are compared and as we can see is that the total time can be reduced by quite an amount, but to be able to do that some routes are getting longer. These longer routes are still within the desired amounts. Furthermore, Table 35 also gives the distances of the new routes. This is given because it gives a quick overview of how many charges are needed and the number of minutes that need to be charged based on the data from Mercedes. Combining the total charge times with the total time of the routes shows that this total time is still less than the route times of the current routes.

Table 35. Comparison and evaluation new routes Tuesday-Friday

	Tuesday-Friday (Current)	**Tuesday-Friday (New)**			
	*Times**	*Phase 1**	*Phase 2**	*Distances*	*Charge times*
Route 1	570 minutes	228 minutes	228 minutes	170 km (0 charges)	0 minutes
Route 2	569 minutes	611 minutes	612 minutes	383 km (1 charge)	0.5 minutes
Route 3	536 minutes	504 minutes	504 minutes	260 km (0 charges)	0 minutes
Route 4	549 minutes	410 minutes	410 minutes	284 km (0 charges)	0 minutes
Route 5	489 minutes	569 minutes	569 minutes	361 km (0 charges)	0 minutes
Route 6	563 minutes	634 minutes	640 minutes	454 km (1 charge)	11 minutes
Route 7	680 minutes	700 minutes	707 minutes	478 km (1 charge)	15 minutes
Route 8	615 minutes	693 minutes	698 minutes	521 km (1 charge)	22 minutes
Route 9	590 minutes	665 minutes	681 minutes	512 km (1 charge)	20 minutes
Route 10	590 minutes	534 minutes	534 minutes	369 km (0 charges)	0 minutes
Route 11	540 minutes	385 minutes	405 minutes	417 km (1 charge)	6 minutes
Route 12	489 minutes	519 minutes	519 minutes	341 km (0 charges)	0 minutes
Total Time	**6780 minutes**	**6452 minutes**	**6507 minutes (Δ-273 min)**	**4550 km**	**74.5 minutes**

**Times are driving times + handling times*

The layout of the routes with names of the locations can be found in Appendix E. Here, the complete overview with the cities' names is given.

5.4.2 Wednesday-Saturday

Section 5.4.2 evaluates the solution for the routes on Wednesday-Saturday. The routes are shown in the same way as in Section 5.4.1, where the blue triangles depict charging locations and black dots are the normal locations. In this set of routes, the biggest problem is the orange route, because these locations lay far away from the depot. It is hard to improve route time as the time of getting to those locations is already longer than some other routes.

Table 36 gives a comparison of route times for the current and new routes for the Wednesday-Saturday routes. Here, we can see that the same is happening with the routes for Tuesday-Friday. To be able to shorten the times for most routes, other routes become longer within the desired times. Table 36 also gives the distances and the charge times for the new routes. Here, it is visible that the distances in this set of routes are much higher than with the routes of Tuesday-Friday. It has an impact on the charge times. With these routes, a lot more charging is needed as the distances are larger. This results in more charging time. In this case, the amount of saved time with the Phase 1 solution is not enough to compensate for the amount of charging that needs to be done. However, the total time of the routes including charging does not become much longer because of the improvement done in Phase 1. The layout of the routes for Wednesday-Saturday with names of the locations are given in Appendix F.

Table 36. Comparison and evaluation new routes Wednesday-Saturday

	Wednesday-Saturday (Current)	**Wednesday-Saturday (New)**			
	*Times**	*Phase 1**	*Phase 2**	*Distances*	*Charge times*
Route 1	644 minutes	652 minutes	654 minutes	482 km (1 charge)	16 minutes
Route 2	732 minutes	746 minutes	863 minutes	833 km (2 charge)	2x35 minutes
Route 3	515 minutes	503 minutes	527 minutes	445 km (1 charge)	10 minutes
Route 4	504 minutes	569 minutes	555 minutes	570 km (1 charge)	30 minutes
Route 5	534 minutes	597 minutes	602 minutes	476 km (1 charge)	15 minutes
Route 6	562 minutes	476 minutes	513 minutes	432 km (1 charge)	8 minutes
Route 7	545 minutes	439 minutes	439 minutes	340 km (0 charge)	0 minutes
Route 8	640 minutes	582 minutes	603 minutes	505 km (1 charge)	20 minutes
Route 9	610 minutes	615 minutes	662 minutes	550 km (1 charge)	26 minutes
Route 10	594 minutes	570 minutes	565 minutes	468 km (1 charge)	14 minutes
Route 11	658 minutes	638 minutes	564 minutes	520 km (1 charge)	22 minutes
Route 12	425 minutes	232 minutes	232 minutes	349 km (0 charge)	0 minutes
Total Time	**6963 minutes**	**6619 minutes**	**6779 minutes (Δ-184 min)**	**5970 km**	**231 minutes**

**Times are driving times + handling times*

5.5 Conclusion on experiments

Chapter 5 considered multiple comparisons and experiments to identify the performance of the proposed solution approach and identify the influence of various input settings on the solution outcome. With different types of instances for each of the presented experiments (Section 5.1), the following conclusions can be made.

First off, it is clear that enlarging the driving range, by either increasing battery capacity or improving battery efficiency, leads to better performance in general. The logical explanation for that is that with more capacity less charging is needed. When less charging is needed, the routes have to visit fewer charging stations which saves time and distance. Furthermore, time at a charging station is additionally saved when less charging is needed. This conclusion is also what lies within the expectation and is confirmed by the performed experiments.

Looking into the performance of the model and the proposed solutions approach, it is visible that Gurobi shows good performance. Within the given time limit, solutions are found with multiple optimal solutions as well. Looking into the performance of the VNS/VND-CIH, this is not as good as some of the approaches found in the literature (Chapter 3). However, it is a simple to apply approach that still can generate reasonable solutions. Looking at Section 5.3.1, it is visible that the performance on the small real-life instances is quite good. This correlates with Section 5.4. which shows that the current situation of VSP can be improved using Phase 1 and EVRPTW for their new situation can be solved with Phase 2.

The main conclusions based on the experiments are the following:

1. In general, creating more driving range by increasing battery capacity or improving battery efficiency, ensures better solutions.
2. The performance of the heuristic is good for smaller real-life instances, however, for medium real-life, small benchmark, or large benchmark instances, the performance is acceptable for an easy-to-use approach like the proposed method.
3. The heuristic is a relatively simple-to-use approach that can help improve a VRPTW situation and solve the EVPRTW in a new situation for real-life cases.

6 Conclusions and recommendations

Chapter 6 concludes the research that is performed. Section 6.1 discusses whether the main research question is answered. Furthermore, we will evaluate the proposed solution method, indicate its flaws, and see if VSP can use it for its EV routing logistics. Section 6.2 states recommendations for the company with regard to the implementation of the found solution, but also what can be done to get even better results. Section 6.3 gives recommendations for future work and Section 6.4 shows what the contribution is to literature and practice.

6.1 Conclusion

The goal of this research is to gain insight into the current performance of vehicle routing at VSP to be able to create new routes for the introduction of EVs. When gaining these insights, VSP is also able to improve its current routing logistics. Therefore, this research wants to answer the main research question:

How can VSP introduce Electric Vehicles to their routing logistics, while also improving the existing routes simultaneously?

The proposed solution method improves VSP's current routing in phase 1, where EVs are introduced into the phase 1 solution to be able to establish routes for the new situation in phase 2. Therefore, we can conclude that the research question for VSP is answered.

Furthermore, concluding on the generalizability of the proposed solution approach, it is shown in Chapter 5 that it can find good solutions for smaller instances and okay solutions for medium or larger instances. Which for a relatively simple approach is acceptable. The approach uses a construction heuristic that can find initial solutions for problems containing time windows, which in Chapter 5 shows it can find solutions for every instance if the problem is not infeasible.

However, a limitation is that the algorithm does perform better in a situation, where not all locations have time windows. Section 5.3 shows that in the real-life instances, the performance is relatively better than in the performance on the benchmark instances in Section 5.2. The benchmark instances have more time windows compared to real-life instances. As the solution approach is designed for the situation of VSP, in case more time windows are considered, the performance becomes less good. However, the solution approach is still able to solve the situations with more time windows.

6.2 Recommendations

This section gives some recommendations for VSP where to focus when further investigating the new situation of implementing EVs. As said, this research shows the potential and do-ability of the implementation when considering the current customer base.

The first recommendation is directly related to the customer base. As the new situation is 2 years away, it is a good idea to investigate whether or not the customer base is going to grow. This has a significant impact on the difficulty of the EVRPTW problem at hand. And if it is a changing factor, determine how to systematically handle adding new customers. This is related to the next recommendation. Which is developing a tool, to help with the determination of the routes. An agile tool that can be used in multiple settings and updated with the new locations to make decisions on the routes. The proposed method can provide a base for such a tool.

The third recommendation is to consider different hubs or depots throughout the country. This was also discussed before this research but intentionally left out to see what is possible without considering hubs. When evaluating the process and inputs, it becomes clear that some routes just cannot be shortened as the

locations in these routes are far away from the depot. This is mainly due to the depot being at a place that is not that centered in the country. Therefore, most of the driving time in routes is going from and to be depot. After showing that routes are possible in this situation, it is not directly clear what happens when the customer base is growing. The chance of having even more time problems when the customer base is reasonably large. Possibly the use of a new depot location that is closer to these locations can help greatly.

The fourth recommendation is related to the proposed solution approach and limitations discussed in Section 6.1. As the performance of the solution approach can fluctuate, the recommendation is to run enough iterations when using the approach to ensure that the heuristic can escape local optima and ensure the best performance the algorithm is capable of.

6.3 Future work

This section contains the recommendations for future work. Trying to identify the gaps that are left after the completion of this research.

The first recommendation for future work is related to the third recommendation in Section 6.2. The biggest problem that VSP is facing is the location of the current depot. That location causes some routes to be impossible to improve as initial driving times to these locations cannot be shortened in the current situation. Therefore, as the third recommendation suggests, the consideration of a second depot or hub is a suggestion for future work. A lot of time can be saved when the initial drive to these locations can be saved. However, future research needs to be done to see where this needs to be, but also how VSP can still ensure that these clothes can go to Poland in the truck that leaves from the existing depot. This combination of the determination of the location as well as the layout of the process can be very beneficial for VSP.

The second recommendation for future work is to see if the proposed solution approach can be improved in relation to problems that consider more time windows. As Chapter 5 shows the performance of the heuristic is better when the problem considers fewer time windows. This makes sense, however, within the found literature in Chapter 3, other approaches can find better solutions that are closer to the found solutions by a solver. Therefore, it could interesting to see if the proposed solution method can also achieve better performance. The generalizability of the proposed solution method would then be better if a company or VSP faces more time windows.

The third recommendation is to investigate in what kind of situation the EVRPTW can be solved as the VRPTW again. At some point in time, electric vehicles can be charged as fast as fueling is done now or the driving range is enlarged to a point it can go as far as a normal car. Nevertheless, it is interesting to keep this development in check, as this era of solving the EVRPTW could be a transition phase, where eventually the problem returns to the VRPTW.

6.4 Contribution to literature and practice

This section includes an overview of the contribution this research has to the literature. Next to that, the contributions to practice are stated.

As far as we know, within the current literature, there is no other two phase solution approach that first optimizes the VRPTW problem and then uses this solution as an initial solution for solving the EVRPTW problem. Within the literature, there are multiple two-phase approaches related to the EVRPTW problem, however these approaches, for example, focus on first clustering the customers and then determining the routes (Li et al., 2023). Furthermore, the paper by Li et al. (2023) was published during the development of the proposed solution approach in this research.

Furthermore, as far as we know, not much research can be found that test their approaches in real-life instances. There are researches considering large instances, however, the consideration of real-life instances of 100 locations is not found frequently. Therefore, this research adds insights into how a solution approach as the proposed one would perform in a real-life situation.

In addition to that, this method is a relatively easy-to-use approach for solving an EVRPTW problem. Therefore, it can lead to improvements with regard to routing performance without putting too much effort in. Relative to other approaches used in literature, this can be used in practice. That is shown by the experiments performed on real-life instances and the results for VSP.

References

Afifi, S., Dang, D. C., & Moukrim, A. (2013). A simulated annealing algorithm for the vehicle routing problem with time windows and synchronization constraints. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7997 LNCS*, 259–265. https://doi.org/10.1007/978-3-642-44973-4_27/TABLES/1

Bac, U., & Erdem, M. (2021). Optimization of electric vehicle recharge schedule and routing problem with time windows and partial recharge: A comparative study for an urban logistics fleet. *Sustainable Cities and Society*, *70*, 102883. https://doi.org/10.1016/J.SCS.2021.102883

Barbarosoglu, G., & Ozgur, D. (1999). A tabu search algorithm for the vehicle routing problem. *Computers & Operations Research*, *26*(3), 255–270. https://doi.org/10.1016/S0305-0548(98)00047-1

Bezerra, S. N., Souza, M. J. F., & de Souza, S. R. (2023). A variable neighborhood search-based algorithm with adaptive local search for the Vehicle Routing Problem with Time Windows and multi-depots aiming for vehicle fleet reduction. *Computers & Operations Research*, *149*, 106016. https://doi.org/10.1016/J.COR.2022.106016

Braekers, K., Ramaekers, K., & Van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, *99*, 300–313. https://doi.org/10.1016/J.CIE.2015.12.007

Bräysy, O. (2003). A Reactive Variable Neighborhood Search for the Vehicle-Routing Problem with Time Windows. *Https://Doi.Org/10.1287/Ijoc.15.4.347.24896*, *15*(4), 347–368. https://doi.org/10.1287/IJOC.15.4.347.24896

Bräysy, O., & Gendreau, M. (2005). Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. *Https://Doi.Org/10.1287/Trsc.1030.0056*, *39*(1), 104–118. https://doi.org/10.1287/TRSC.1030.0056

Breedam, A. Van. (2001). Comparing descent heuristics and metaheuristics for the vehicle routing problem. *Computers & Operations Research*, *28*(4), 289–315. https://doi.org/10.1016/S0305-0548(99)00101-X

Bruglieri, M., Pezzella, F., Pisacane, O., & Suraci, S. (2015). A Variable Neighborhood Search Branching for the Electric Vehicle Routing Problem with Time Windows. *Electronic Notes in Discrete Mathematics*, *47*, 221–228. https://doi.org/10.1016/J.ENDM.2014.11.029

Chen, J., Huang, H., Zhang, Z., & Wang, J. (2022). Deep Reinforcement Learning with Two-Stage Training Strategy for Practical Electric Vehicle Routing Problem with Time Windows. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *13398 LNCS*, 356–370. https://doi.org/10.1007/978-3-031-14714-2_25/FIGURES/3

Chiang, W. C., & Russell, R. A. (1996). Simulated annealing metaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research*, *63*, 3–27. https://doi.org/10.1007/BF02601637

Christofides, N., & Beasley, J. E. (1984). The period routing problem. *Networks*, *14*(2), 237–256. https://doi.org/10.1002/NET.3230140205

Clarke, G., & Wright, J. W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, *12*(4), 568–581. https://doi.org/10.1287/OPRE.12.4.568

Cordeau, J. F., Laporte, G., Savelsbergh, M. W. P., & Vigo, D. (2007). Chapter 6 Vehicle Routing. *Handbooks in Operations Research and Management Science*, *14*(C), 367–428.

https://doi.org/10.1016/S0927-0507(06)14006-2

Corona-Gutiérrez, K., Nucamendi-Guillén, S., & Lalla-Ruiz, E. (2022). Vehicle routing with cumulative objectives: A state of the art and analysis. *Computers & Industrial Engineering*, *169*, 108054. https://doi.org/10.1016/J.CIE.2022.108054

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, *6*(1), 80–91. https://doi.org/10.1287/MNSC.6.1.80

Dror, M., & Levy, L. (1986). A vehicle routing improvement algorithm comparison of a "greedy" and a matching implementation for inventory routing. *Computers & Operations Research*, *13*(1), 33–45. https://doi.org/10.1016/0305-0548(86)90062-6

Erdelić, T., Carić, T., Erdelić, M., & Tišljarić, L. (2019). Electric vehicle routing problem with single or multiple recharges. *Transportation Research Procedia*, *40*, 217–224. https://doi.org/10.1016/J.TRPRO.2019.07.033

Erdoğan, S., & Miller-Hooks, E. (2012). A Green Vehicle Routing Problem. *Transportation Research Part E: Logistics and Transportation Review*, *48*(1), 100–114. https://doi.org/10.1016/j.tre.2011.08.001

Euchi, J., & Yassine, A. (2022). A hybrid metaheuristic algorithm to solve the electric vehicle routing problem with battery recharging stations for sustainable environmental and energy optimization. *Energy Systems*, *14*(1), 243–267. https://doi.org/10.1007/S12667-022-00501-Y/FIGURES/9

Fernández Gil, A., Lalla-Ruiz, E., Gómez Sánchez, M., & Castro, C. (2022). A Review of Heuristics and Hybrid Methods for Green Vehicle Routing Problems considering Emissions. *Journal of Advanced Transportation*, *2022*, 1–38. https://doi.org/10.1155/2022/5714991

García-López, F., Melián-Batista, B., Moreno-Pérez, J. A., & Moreno-Vega, J. M. (2002). The parallel variable neighborhood search for the p-median problem. *Journal of Heuristics*, *8*(3), 375–388. https://doi.org/10.1023/A:1015013919497/METRICS

Goel, A., & Gruhn, V. (2008). A General Vehicle Routing Problem. *European Journal of Operational Research*, *191*(3), 650–660. https://doi.org/10.1016/J.EJOR.2006.12.065

Joshi, S., & Kaur, S. (2015). Nearest Neighbor Insertion Algorithm for solving capacitated vehicle routing problem. *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*.

Kallehauge, B., Larsen, J., Madsen, O. B. G., & Solomon, M. M. (2005). Vehicle routing problem with time windows. *Column Generation*, 67–98. https://doi.org/10.1007/0-387-25486-2_3/COVER

Keskin, M., & Çatay, B. (2016). Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, *65*, 111–127. https://doi.org/10.1016/J.TRC.2016.01.013

Keskin, M., & Çatay, B. (2018). A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. *Computers & Operations Research*, *100*, 172–188. https://doi.org/10.1016/J.COR.2018.06.019

Keskin, M., Çatay, B., & Laporte, G. (2021). A simulation-based heuristic for the electric vehicle routing problem with time windows and stochastic waiting times at recharging stations. *Computers & Operations Research*, *125*, 105060. https://doi.org/10.1016/J.COR.2020.105060

KVK. (2022). *Elektrisch rijden: deze maatregelen komen op je af*. https://www.kvk.nl/advies-en-

informatie/innovatie/duurzaam-ondernemen/elektrisch-rijden-deze-maatregelen-komen-op-je-af/#:~:text=Vanaf 2025 mogen gemeenten zero,emissie zones niet meer binnen

Kytöjoki, J., Nuortio, T., Bräysy, O., & Gendreau, M. (2007). An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, *34*(9), 2743–2757. https://doi.org/10.1016/J.COR.2005.10.010

Lalla-Ruiz, E., & Voß, S. (2020). A POPMUSIC approach for the Multi-Depot Cumulative Capacitated Vehicle Routing Problem. *Optimization Letters*, *14*(3), 671–691. https://doi.org/10.1007/S11590-018-1376-1

Lam, E., Desaulniers, G., & Stuckey, P. J. (2022). Branch-and-cut-and-price for the Electric Vehicle Routing Problem with Time Windows, Piecewise-Linear Recharging and Capacitated Recharging Stations. *Computers & Operations Research*, *145*, 105870. https://doi.org/10.1016/J.COR.2022.105870

Laporte, G. (2007). What you should know about the vehicle routing problem. *Naval Research Logistics (NRL)*, *54*(8), 811–819. https://doi.org/10.1002/NAV.20261

Lenstra, J. K., & Kan, A. H. G. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, *11*(2), 221–227. https://doi.org/10.1002/NET.3230110211

Li, M. ; ;, Hao, J. A., Lai, K. K., Yu, L., Chai, J., Ding, N., Li, M., & Hao, J. (2023). A Two-Phase Approach to Routing a Mixed Fleet with Intermediate Depots. *Mathematics 2023, Vol. 11, Page 1924*, *11*(8), 1924. https://doi.org/10.3390/MATH11081924

Lin, B., Ghaddar, B., & Nathwani, J. (2022). Deep Reinforcement Learning for the Electric Vehicle Routing Problem With Time Windows. *IEEE Transactions on Intelligent Transportation Systems*, *23*(8), 11528–11538. https://doi.org/10.1109/TITS.2021.3105232

Lin, J., Zhou, W., & Wolfson, O. (2016). Electric Vehicle Routing Problem. *Transportation Research Procedia*, *12*, 508–521. https://doi.org/10.1016/J.TRPRO.2016.02.007

Maniezzo, V., Stützle, T., & Voß, S. (2010). Matheuristics: Hybridizing Metaheuristics and Mathematical Programming. *Springer Verlag*, *10*, 1–282. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Matheuristics#1

Mao, H., Shi, J., Zhou, Y., & Zhang, G. (2020). The Electric Vehicle Routing Problem with Time Windows and Multiple Recharging Options. *IEEE Access*, *8*, 114864–114875. https://doi.org/10.1109/ACCESS.2020.3003000

Marinho Diana, R. O., & de Souza, S. R. (2020). Analysis of variable neighborhood descent as a local search operator for total weighted tardiness problem on unrelated parallel machines. *Computers & Operations Research*, *117*, 104886. https://doi.org/10.1016/J.COR.2020.104886

Miltenberger, M. (2023). *What is the MIPGap?* https://support.gurobi.com/hc/en-us/articles/8265539575953-What-is-the-MIPGap-

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*(11), 1097–1100. https://doi.org/10.1016/S0305-0548(97)00031-2

Montané, F. A. T., & Galvão, R. D. (2006). A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers & Operations Research*, *33*(3), 595–619. https://doi.org/10.1016/J.COR.2004.07.009

Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, *41*(4), 421–451. https://www.academia.edu/4688179/Metastrategy_simulated_annealing_and_tabu_search_algorithms_for_the_vehicle_routing_problem

Qin, H., Su, X., Ren, T., & Luo, Z. (2021). A review on the electric vehicle routing problems: Variants and algorithms. *Frontiers of Engineering Management 2021 8:3*, *8*(3), 370–389. https://doi.org/10.1007/S42524-021-0157-1

Qiu, F., Geng, N., & Wang, H. (2023). An improved memetic algorithm for integrated production scheduling and vehicle routing decisions. *Computers & Operations Research*, *152*, 106127. https://doi.org/10.1016/J.COR.2022.106127

Raza, S. M., Sajid, M., & Singh, J. (2022). Vehicle Routing Problem Using Reinforcement Learning: Recent Advancements. *Lecture Notes in Electrical Engineering*, *858*, 269–280. https://doi.org/10.1007/978-981-19-0840-8_20/FIGURES/4

Schneider, M., Stenger, A., & Goeke, D. (2014). The Electric Vehicle-Routing Problem with Time Windows and Recharging Stations. *Https://Doi.Org/10.1287/Trsc.2013.0490*, *48*(4), 500–520. https://doi.org/10.1287/TRSC.2013.0490

Solomon, M. M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Https://Doi.Org/10.1287/Opre.35.2.254*, *35*(2), 254–265. https://doi.org/10.1287/OPRE.35.2.254

Tavakkoli-Moghaddam, R., Gazanfari, M., Alinaghian, M., Salamatbakhsh, A., & Norouzi, N. (2011). A new mathematical model for a competitive vehicle routing problem with time windows solved by simulated annealing. *Journal of Manufacturing Systems*, *30*(2), 83–92. https://doi.org/10.1016/J.JMSY.2011.04.005

Ursani, Z., Essam, D., Cornforth, D., & Stocker, R. (2011). Localized genetic algorithm for vehicle routing problem with time windows. *Applied Soft Computing*, *11*(8), 5375–5390. https://doi.org/10.1016/J.ASOC.2011.05.021

Wang, W., & Zhao, J. (2023). Partial linear recharging strategy for the electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research*, *308*(2), 929–948. https://doi.org/10.1016/J.EJOR.2022.12.011

Wingerden, J. van. (2022). *"KABINET VERPLICHT ELEKTRISCHE LEASEAUTO PER 2025."* AutoWeek. https://www.autoweek.nl/autonieuws/artikel/vanaf-2025-moet-nieuwe-leaseauto-volledig-elektrisch-zijn/?referrer=https%3A%2F%2Fwww.google.com%2F

Woch, M., & Łebkowski, P. (2009). Sequential Simulated Annealing for the Vehicle Routing Problem with Time Windows. *Decision Making in Manufacturing and Services*, *3*(2), 87–100. https://doi.org/10.7494/DMMS.2009.3.2.87

Appendix

Appendix A

The Excel file ‘traveltimedata’ contains the matrix with all the driving times between all combinations of locations. If the reader wants to see the file, it can be requested by the author.

Appendix B

For the service times, there are two files containing these times. The same as for the travel time matrix, it can be requested by the author, if the reader wants to see the file. These files are ‘Locations coordinates2 tuefri’ and ‘Locations coordinates2 wedsat’.

Appendix C

An Excel file including a matrix with all travel times between all locations including charging locations can be requested by the author. These are such large files, that cannot be presented in the report itself.

Appendix D

An Excel file including a matrix with all distances between all locations including charging locations can be requested by the author. These are such large files, that cannot be presented in the report itself.

Appendix E

['Depot', 'Hengelo', 'Lichtenvoorde', 'Aalten', 'Winterswijk', 'Sudlohn', 'Ahaus', 'Depot1']

['Depot', 'Amsterdam Centrum', 'Maarssen', 'Bilthoven', 'Zeist', 'Soesterberg (KPU)', 'Leusden', 'Lunteren', 'Barneveld', 'Kootwijkerbroek', 'De Hucht', 'Depot1']

['Depot', 'Arnhem', 'Duiven', 'Zevenaar', 'Didam', 'Ulft', 'Doetinchem', 'Zelhem', 'Vorden', 'Borculo', 'Enschede Haven', 'Enschede', 'Depot1']

['Depot', 'Amersfoort industrie', 'Amersfoort', 'Nijkerkerveen', 'Nijkerk', 'Voorthuizen', 'Rijssen', 'Almelo', 'Depot1']

['Depot', 'Utrecht', 'Utrecht Leidsche Rijn', 'Harmelen', 'Woerden', 'Ijsselstein', 'Nieuwegein', 'Houten', 'Driebergen', 'Woudenberg', 'Depot1']

['Depot', 'Wijchen', 'Grave', 'Rosmalen', 'Den Bosch', 'Waalwijk', 'Kaatsheuvel', 'Dussen', 'Werkendam', 'Gorinchem', 'Leerdam', 'Zaltbommel', 'Opheusden', 'Heteren', 'Gelredome', 'Depot1']

['Depot', 'Schiphol', 'Halfweg', 'Haarlem', 'Beverwijk', 'Wormerveer', 'Purmerend', 'Zaandam', 'Amsterdam Noord', 'Soest', 'Baarn', 'Bunschoten', 'De Slaag', 'Schalkhaar', 'Deventer', 'Het Veelsveld', 'Depot1']

['Depot', 'Alkmaar', 'Bergen', 'Den Helder', 'Schagen', 'Heerhugowaard', 'Hoorn', 'Lelystad', 'Han Stijkel', 'Emmeloord', 'Wijhe ', 'Raalte', 'Nijverdal', 'Depot1']

['Depot', 'Vught', 'Oisterwijk', 'Udenhout', 'Tilburg', 'Bavel', 'Baarle Nassau', 'Uden', 'Oss', 'Druten', 'Gelredome', 'Elst', 'Oldenzaal', 'Depot1']

['Depot', 'Apeldoorn', 'Apeldoorn Noord', 'Hattem', 'Kampen', 'Zwolle', 'Lemelerveld', 'Emmen', 'Klazienaveen', 'Emmer-Compascuum', 'Haselünne', 'Depot1']

['Depot', 'Minden', 'Stadthagen', 'Bad Oeynhausen', 'Munster', 'Munster-Nienberge', 'Het Veelsveld', 'Depot1']

['Depot', 'Zutphen', 'Doesburg', 'Nijmegen', 'Nijmegen Zuid', 'Ede', 'Veenendaal', 'Rhenen', 'Wageningen', 'Bennekom', 'Renkum', 'Oosterbeek', 'Depot1']]

Appendix F

['Depot', 'Jubbega', 'Drachten', 'Surhuisterveen', 'Leeuwarden', 'Franeker', 'Sneek', 'Balk', 'Delfstrahuizen', 'Joure', 'Heerenveen', 'Wolvega', 'Steenwijk', 'Meppel', 'Panjerd', 'Lageveen', 'Hoogeveen', 'Depot1']

['Depot', 'Krabbendijke', "'s-Gravenpolder", 'Goes', 'Vlissingen', 'Middelburg', 'Terneuzen', 'Zaamslag', 'Vliedberg', 'Vliedberg', 'Oostburg', 'Sluis', 'Heinkenszand', 'Middelharnis', 'Tolnegen', 'Depot1']

['Depot', 'Gouda', 'Rotterdam Alexandrium', 'Capelle ad Ijssel', 'Rotterdam Ijsselmonde', 'Ridderkerk', 'Dordrecht', 'Geldermalsen', 'Tiel', 'Varakker', 'Varakker', 'Depot1']

['Depot', 'Roermond', 'Maastricht', 'Meerssen', 'Beek', 'Sittard', 'Heerlen', 'Kerkrade', 'Het Veelsveld', 'Het Veelsveld', 'Depot1']

['Depot', 'Bodegraven', 'Alphen ad Rijn', 'Leiderdorp', 'Leiden', 'Katwijk', 'Leimuiden', 'Hoofddorp', 'Amstelveen', 'Amsterdam Bijlmer', 'Weesp', 'Almere', 'Laren ', 'Tolnegen', 'Tolnegen', 'Depot1']

['Depot', 'Onstwedde', 'Veendam', 'Assen', 'Haren', 'Groningen', 'Leek', 'Appingedam', 'Winschoten', 'Het Veelsveld', 'Het Veelsveld', 'Depot1']

['Depot', 'Hilversum', 'Bussum', 'Huizen', 'Putten', 'Ermelo', 'Harderwijk', 'Nunspeet', 'Epe', 'Oene', 'Depot1']

['Depot', 'Rotterdam Hesseplaats', 'Rotterdam Hillegersberg', 'Rotterdam Centrum', 'Schiedam', 'Maassluis', 'Vlaardingen', 'Hoogvliet', 'Hellevoetsluis', 'Spijkernisse', 'Rotterdam Zuidplein', 'Waddinxveen', 'Tolnegen', 'Tolnegen', 'Depot1']

['Depot', 'Weert', 'Helmond', 'Eindhoven', 'Eindhoven Wildenberg', 'Eindhoven Woensel', 'Veldhoven', 'Waalre', 'Valkenswaard', 'Bladel', 'Reusel', 'Sevenum', 'Horst', 'Geulenkamp', 'Geulenkamp', 'Depot1']

['Depot', 'Pijnacker', 'Delft ', "'s-Gravenzande", 'Naaldwijk', 'Rijswijk', 'Den Haag', 'Voorburg', 'Leidschendam', 'Zoetermeer', 'Tolnegen', 'Tolnegen', 'Depot1']

['Depot', 'Oosterhout', 'Bavel', 'Breda', 'Bergen op Zoom', 'Roosendaal', 'Etten Leur', 'Made', 'Tolnegen', 'Tolnegen', 'Depot1']

['Depot', 'Venlo', 'Depot1']]

Appendix G

#instance 1

Customer locations=[3, 7, 72, 78, 90]

Charging locations=[132, 179, 182, 234, 245]

#instance 2

Customer locations=[6, 44, 81, 92, 93]

Charging locations=[140, 199, 210, 229, 245]

#instance 3

Customer locations=[7, 51, 64, 98, 106]

Charging locations=[117, 159, 169, 176, 194]

#instance 4

Customer locations=[21, 28, 42, 50, 94]

Charging locations=[130, 154, 224, 240, 255]

#instance 5

Customer locations=[4, 20, 32, 40, 89]

Charging locations=[132, 184, 224, 236, 239]

#instance 6

Customer locations=[1, 9, 21, 39, 51, 55, 78, 79, 90, 91]

Charging locations=[121, 125, 131, 138, 153, 168, 188, 217, 233, 248]

#instance 7

Customer locations=[16, 32, 38, 43, 48, 61, 76, 81, 91, 110]

Charging locations=[131, 148, 164, 167, 168, 185, 226, 231, 234, 238]

#instance 8

Customer locations=[2, 23, 44, 45, 49, 59, 71, 89, 110, 114]

Charging locations=[117, 122, 126, 129, 131, 150, 212, 219, 245, 251]

#instance 9

Customer locations=[8, 17, 23, 27, 31, 66, 100, 106, 108, 111]

Charging locations=[128, 153, 166, 189, 200, 219, 227, 239, 248, 254]

#instance 10

Customer locations=[2, 16, 34, 48, 57, 64, 82, 105, 110, 112]

Charging locations=[120, 143, 154, 157, 164, 169, 207, 221, 225, 244]

#instance 11

Customer locations=[2, 4, 7, 13, 27, 32, 33, 34, 48, 52, 55, 57, 77, 113, 115]

Charging locations=[120, 122, 127, 135, 145, 151, 175, 176, 180, 206, 208, 220, 226, 244, 248]

#instance 12

Customer locations=[11, 18, 21, 26, 34, 41, 46, 53, 73, 80, 84, 88, 90, 99, 113]

Charging locations=[121, 127, 146, 147, 154, 155, 168, 175, 187, 199, 216, 219, 220, 234, 238]

#instance 13

Customer locations=[2, 16, 25, 28, 36, 43, 46, 50, 55, 63, 79, 82, 89, 92, 103]

Charging locations=[120, 121, 122, 151, 159, 165, 168, 175, 181, 212, 221, 223, 229, 252, 253]

#instance 14

Customer locations=[11, 14, 28, 32, 38, 44, 51, 65, 75, 77, 83, 85, 90, 100, 106]

Charging locations=[130, 137, 150, 168, 174, 193, 196, 199, 209, 213, 214, 223, 228, 240, 242]

#instance 15

Customer locations=[2, 13, 16, 31, 33, 34, 43, 44, 50, 58, 62, 66, 74, 104, 112]

Charging locations=[119, 121, 132, 139, 152, 155, 168, 182, 204, 221, 224, 225, 227, 229, 237]

Appendix H

#instance 1

#Customer locations=[2, 3, 6, 16, 18, 19, 21, 22, 27, 33, 34, 35, 38, 41, 43, 44, 48, 49, 50, 52, 57, 62, 65, 69, 70, 75, 78, 81, 82, 83, 86, 88, 97, 98, 107, 110, 112, 113, 114, 115]

#Charging locations=[117, 127, 133, 134, 136, 137, 145, 146, 148, 152, 156, 160, 162, 165, 173, 176, 177, 182, 190, 191, 196, 202, 212, 219, 222, 237, 243, 249, 250, 251]

#instance 2

#Customer locations=[6, 9, 10, 11, 12, 16, 21, 24, 25, 33, 37, 40, 44, 45, 50, 59, 62, 68, 74, 76, 77, 79, 80, 83, 86, 90, 94, 95, 98, 103, 104, 106, 107, 108, 109, 112, 113, 114, 115, 116]

#Charging locations=[125, 130, 133, 134, 135, 139, 142, 145, 149, 157, 158, 159, 162, 165, 180, 182, 183, 192, 195, 199, 204, 208, 220, 233, 240, 241, 243, 244, 252, 253]

#instance 3

#Customer locations=[1, 2, 3, 4, 6, 10, 13, 18, 25, 26, 28, 31, 32, 38, 43, 45, 48, 59, 61, 64, 65, 68, 69, 71, 74, 77, 80, 82, 83, 87, 90, 91, 92, 94, 100, 102, 107, 109, 110, 114]

#Charging locations=[119, 125, 131, 133, 135, 136, 148, 149, 150, 153, 156, 157, 159, 162, 169, 172, 173, 175, 185, 205, 210, 211, 212, 215, 222, 223, 236, 237, 247, 250]

#instance 4

#Customer locations=[7, 10, 12, 15, 16, 18, 19, 21, 22, 25, 26, 36, 38, 40, 42, 43, 46, 47, 48, 50, 53, 56, 57, 62, 63, 67, 68, 70, 72, 84, 89, 90, 91, 98, 99, 105, 109, 111, 112, 116]

#Charging locations=[126, 127, 129, 135, 139, 143, 146, 147, 148, 150, 154, 155, 159, 162, 163, 172, 173, 179, 193, 196, 197, 200, 204, 206, 214, 215, 229, 236, 247, 248]

#instance 5

#Customer locations=[4, 5, 7, 9, 11, 13, 15, 18, 19, 22, 27, 29, 34, 35, 36, 37, 39, 45, 46, 47, 49, 53, 62, 66, 68, 74, 75, 76, 77, 80, 84, 95, 100, 102, 103, 104, 109, 110, 112, 115]

#Charging locations=[118, 122, 126, 128, 130, 133, 136, 137, 140, 142, 145, 147, 153, 155, 156, 164, 171, 172, 173, 187, 193, 212, 216, 217, 218, 219, 221, 244, 247, 251]

#instance 6

#Customer locations=[1, 2, 3, 4, 5, 6, 12, 13, 16, 18, 19, 22, 23, 25, 26, 28, 33, 34, 35, 38, 39, 40, 46, 50, 51, 53, 56, 57, 62, 66, 68, 73, 75, 76, 78, 82, 88, 89, 93, 94, 98, 100, 102, 103, 104, 106, 107, 108, 113, 116]

#Charging locations=[117, 120, 121, 123, 129, 130, 131, 141, 147, 151, 152, 153, 156, 162, 168, 170, 173, 175, 178, 179, 195, 198, 199, 200, 202, 204, 206, 208, 209, 210, 214, 219, 229, 232, 235, 240, 251, 252, 253, 254]

#instance 7

#Customer locations=[3, 5, 7, 8, 11, 12, 13, 17, 20, 27, 28, 29, 31, 33, 34, 35, 36, 37, 40, 41, 42, 44, 45, 48, 50, 53, 54, 59, 62, 64, 70, 71, 72, 73, 74, 75, 76, 78, 79, 83, 85, 87, 90, 93, 96, 98, 100, 104, 107, 116]

#Charging locations=[119, 124, 131, 133, 135, 141, 149, 152, 153, 154, 157, 158, 159, 165, 168, 171, 176, 177, 179, 182, 185, 187, 191, 192, 194, 195, 198, 207, 214, 215, 224, 227, 231, 233, 235, 237, 241, 242, 243, 252]

#instance 8

#Customer locations=[5, 6, 12, 13, 16, 17, 18, 19, 20, 21, 23, 25, 26, 27, 29, 31, 32, 33, 39, 40, 41, 42, 44, 54, 57, 59, 61, 62, 64, 65, 70, 74, 75, 81, 83, 85, 86, 88, 89, 94, 95, 96, 97, 102, 103, 104, 105, 108, 112, 113]

#Charging locations=[121, 122, 125, 126, 133, 137, 140, 141, 143, 145, 151, 156, 158, 160, 163, 169, 171, 173, 176, 189, 195, 200, 204, 206, 207, 211, 212, 213, 221, 222, 223, 226, 229, 233, 238, 239, 241, 242, 249, 253]

#instance 9

#Customer locations=[2, 7, 10, 11, 18, 25, 28, 29, 32, 35, 37, 40, 41, 43, 44, 47, 48, 52, 53, 54, 55, 56, 57, 58, 59, 62, 64, 71, 74, 77, 80, 83, 84, 89, 91, 93, 97, 98, 99, 101, 102, 103, 104, 105, 106, 107, 108, 111, 114, 115]

#Charging locations=[124, 128, 135, 136, 142, 143, 144, 146, 153, 157, 172, 173, 176, 177, 178, 179, 181, 186, 193, 194, 199, 200, 202, 204, 205, 210, 214, 217, 219, 225, 226, 228, 233, 236, 243, 246, 247, 249, 250, 253]

#instance 10

#Customer locations=[2, 3, 5, 6, 14, 18, 20, 28, 30, 31, 35, 36, 39, 42, 43, 46, 47, 49, 50, 53, 55, 56, 57, 60, 61, 63, 68, 72, 73, 78, 80, 85, 86, 88, 89, 90, 93, 94, 97, 98, 100, 101, 103, 106, 107, 108, 110, 112, 113, 114]

#Charging locations=[126, 127, 129, 131, 134, 137, 139, 141, 143, 144, 145, 147, 153, 154, 155, 156, 157, 158, 159, 162, 165, 166, 178, 179, 182, 193, 194, 197, 201, 208, 216, 218, 230, 234, 239, 245, 246, 248, 250, 253]

#instance 11

#Customer locations=[3, 4, 5, 11, 12, 13, 14, 15, 16, 19, 20, 21, 22, 23, 24, 26, 30, 31, 32, 33, 35, 37, 40, 41, 44, 46, 49, 56, 57, 60, 61, 63, 64, 68, 69, 70, 71, 72, 74, 75, 76, 77, 79, 80, 82, 83, 84, 86, 90, 93, 94, 96, 101, 102, 103, 106, 108, 110, 112, 113]

#Charging locations=[120, 121, 123, 127, 129, 132, 135, 136, 140, 142, 143, 147, 149, 154, 159, 164, 167, 170, 173, 176, 178, 182, 184, 185, 191, 193, 195, 196, 197, 198, 212, 213, 217, 218, 219, 221, 224, 227, 228, 236, 237, 238, 239, 242, 243, 244, 245, 246, 251, 254]

#instance 12

#Customer locations=[1, 2, 4, 6, 8, 9, 10, 13, 19, 25, 27, 29, 30, 31, 32, 35, 36, 37, 38, 39, 40, 42, 44, 45, 46, 47, 50, 52, 53, 54, 55, 56, 59, 60, 61, 62, 65, 72, 74, 75, 77, 78, 79, 80, 81, 83, 85, 87, 90, 91, 92, 94, 96, 100, 101, 103, 106, 108, 110, 116]

#Charging locations=[117, 118, 120, 121, 122, 130, 131, 139, 140, 141, 142, 144, 146, 149, 153, 156, 159, 163, 165, 168, 176, 181, 186, 190, 191, 197, 203, 204, 205, 206, 208, 211, 215, 219, 226, 228, 229, 230, 231, 232, 233, 235, 236, 239, 242, 244, 245, 251, 254, 255]

#instance 13

#Customer locations=[1, 4, 5, 6, 9, 11, 13, 14, 17, 22, 23, 24, 26, 27, 28, 29, 30, 32, 33, 34, 36, 37, 38, 39, 41, 42, 43, 44, 46, 48, 49, 51, 52, 58, 59, 61, 64, 69, 70, 71, 75, 77, 78, 80, 81, 82, 84, 85, 88, 90, 92, 93, 98, 102, 104, 105, 106, 108, 115, 116]

#Charging locations=[118, 122, 128, 129, 131, 132, 133, 136, 140, 145, 149, 151, 154, 155, 156, 157, 158, 159, 165, 167, 168, 169, 170, 171, 176, 177, 179, 187, 193, 194, 198, 200, 202, 208, 212, 215, 225, 226, 228, 231, 234, 237, 239, 241, 243, 246, 249, 250, 251, 252]

#instance 14

#Customer locations=[1, 2, 5, 7, 8, 15, 16, 17, 18, 19, 20, 21, 22, 23, 26, 27, 28, 30, 32, 35, 37, 38, 40, 41, 42, 44, 45, 46, 48, 53, 54, 55, 56, 59, 63, 66, 68, 71, 74, 76, 78, 81, 82, 83, 84, 85, 86, 87, 88, 89, 92, 97, 98, 99, 103, 104, 105, 106, 110, 116]

#Charging locations=[118, 124, 128, 134, 135, 136, 137, 139, 145, 149, 156, 159, 160, 164, 166, 172, 174, 178, 179, 180, 184, 187, 191, 193, 194, 195, 200, 202, 204, 205, 206, 207, 208, 210, 212, 213, 215, 216, 219, 229, 230, 238, 241, 242, 244, 246, 249, 250, 254, 255]

#instance 15

#Customer locations=[1, 2, 3, 6, 8, 9, 10, 11, 16, 19, 20, 21, 22, 25, 26, 27, 31, 32, 34, 35, 36, 37, 39, 42, 45, 46, 47, 48, 50, 54, 55, 56, 57, 67, 69, 71, 73, 75, 76, 78, 79, 82, 84, 85, 86, 91, 92, 93, 94, 98, 99, 101, 103, 106, 107, 109, 112, 113, 114, 116]

#Charging locations=[121, 126, 127, 135, 136, 137, 139, 140, 142, 144, 146, 147, 149, 151, 153, 156, 157, 158, 161, 164, 166, 171, 172, 174, 179, 180, 181, 188, 193, 195, 198, 201, 203, 206, 209, 212, 217, 218, 220, 224, 225, 237, 238, 239, 240, 246, 248, 250, 252, 255]